### 运行方法：

1. 运行R:

library(Rserve)

Rserve(port=6000)

1. 运行Python Django:

cd D:\youge\_random\ComputeEngine\djangoR\src

python manage.py runserver

### 使用方法：

共有5种随机化方案的实现。这些方案都是通过HTTP POST方法获取计算结果。输入参数以及计算结果都是JSON格式。

JSON作为参数输入对变量名（用作key-value pair中的key）有严格的要求，特别是大小写，不一致就会出现问题。

Minimization Randomization的输入参数较为复杂，我们对markerOld（如果不是空的情况）、markerNew的输入有**特殊要求**，主要表现在第一个元素的key必须是patientID，最后一个元素的key是treatment。中间的marker，依次为F1, F2, F3, …。所以这里面的顺序是重要信息。

变量名限制在如下几种（**红色标记为修改过的，与以前不同**）：

(1) Single word, special vars: capitalized

K

N

(2) Single word, lower letter

seed

weight

treatment

F1

F2

F3

nlevel

nfactor

(3) More than 2 words, lower camelCase

blockSize

alloRatio

markerOld

markerNew

patientID

##### (1) Complete Randomization

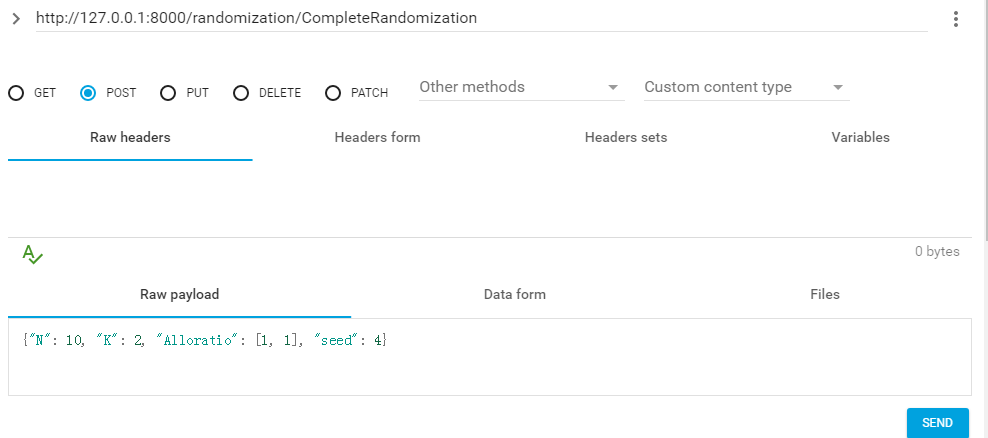
<http://127.0.0.1:8000/randomization/CompleteRandomization>

**输入参数**

|  |  |  |
| --- | --- | --- |
| **参数名** | **描述** | **备注** |
| N | 样本量 | 样本数应该是分组数K和sum(alloRatio)的倍数。 例如，我们需要分成2组,入组比例为2:1（K=2，alloRatio=(2:1)）,则N应为2和2+1=3的倍数，即6的倍数，例如6,12，24等。 |
| K | 分组数 | 例如K=2， 则表明有两个组。 |
| alloRatio | 入组人数之比例 | 长度是K的向量。假设K=2， 如果alloRatio=(1:1),则为  均衡随机。如果alloRatio=(2:1),则分配到A组和到B组的比例为2:1。 |
| seed | 随机种子编号 | 相同的随机种子编号可以得到相同的随机化结果；不同种子编号会得到不同随机化结果。这个选项可以不开放给用户。 |

POST参数示例：

{"N": 10, "K": 2, "alloRatio": [1, 1], "seed": 4}



在这个例子里，程序将10个病人均衡随机到2个组里面。随机种子编号为4.

**输出参数**

|  |  |  |
| --- | --- | --- |
| **参数名** | **描述** | **备注** |
| patientID | 病人顺序号 | 长度为N的向量，第i个元素代表第i个病人的顺序号 |
| group | 病人分组 | 长度为N的向量，第i个元素代表第i个顺序号的病人被分配到的实验组编号。实验组编号为1， 2，…,K |
| summary | 分组总结 | 长度为K的向量。第i个代表被分配到第i个实验组中病人的人数。 |
| status | 程序状况指示 | 返回“success”随机化成功。如果返回“failure”, 程序会停止运行，并显示错误提示信息。 |

结果：

{"patientID":["1","2","3","4","5","6","7","8","9","10"],"group":["1","2","2","2","1","2","1","1","1","2"],"summary":[[5,5]],"summary\_rowname":[""],"summary\_colname":["group.1","group.2"],"status":"success"}

这个例子里，status为“sucess”,说明算法成功执行。分组结果 “group“和病人顺序号“patientID”按顺序一一对应。例如，顺序号为1的病人被分配到实验组1而顺序号为2的病人被分配到实验组2,依次类推。 向量“summary”为（5,5），代表各有5个病人分配到group1和group2中。

##### （2）Block Randomization

<http://127.0.0.1:8000/randomization/BlockRandomization>

**输入参数**

|  |  |  |
| --- | --- | --- |
| **参数名** | **描述** | **备注** |
| N | 总样本量 | 样本数应该是分组数K和sum(alloRatio)的倍数,还应该为blocksize的倍数。假设2组（K=2），每组比例为2:1（alloRatio=（2,1））， 区块长度为5（blocksize=5），则N应该为2,2+1=3和5的倍数，即30的倍数，如30,60，90等。 |
| K | 分组数 | 例如K=2， 则表明有两个组。 |
| alloRatio | 入组人数之比例 | 长度是K的向量。假设K=2， 如果alloRatio=(1:1),则为  均衡随机。如果alloRatio=(2:1),则分配到A组和到B组的比例为2:1。 |
| blookSize | 区块长度。 | 目前不支持随机区组长度。 |
| seed | 随机种子编号 | 相同的随机种子编号可以得到相同的随机化结果；不同种子编号会得到不同随机化结果。这个选项可以不开放给用户。 |

POST参数示例：

{"N": 30, "K": 3，"alloRatio": [2, 2, 2], "blocksize": 3, "seed": 4}

在这个例子里，程序将60个病人均衡随机到3个组里面，区组长度为3。随机种子编号为4.

**输出参数**

|  |  |  |
| --- | --- | --- |
| **参数名** | **描述** | **备注** |
| patientID | 病人顺序号 | 长度为N的向量，第i个元素代表第i个病人的顺序号 |
| group | 病人分组 | 长度为N的向量，第i个元素代表第i个顺序号的病人被分配到的实验组编号。实验组编号为1， 2，…,K |
| summary | 分组总结 | 长度为K的向量。第i个代表被分配到第i个实验组中病人的人数。 |
| status | 程序状况指示 | 返回“success”随机化成功。如果返回“failure”, 程序会停止运行，并显示错误提示信息。 |

结果：

{"patientID":["1","2","3","4","5","6","7","8","9","10","11","12","13","14","15","16","17","18","19","20","21","22","23","24","25","26","27","28","29","30"],"group":["2","1","3","1","2","3","3","2","1","1","2","3","1","2","3","2","3","1","3","2","1","3","2","1","2","3","1","3","2","1"],"summary":[[10,10,10]],"summary\_rowname":["1"],"summary\_colname":["group.1","group.2","group.3"],"status":"success"}

这个例子里，status为“sucess”,说明算法成功执行。分组结果 “group“和病人顺序号“patientID”按顺序一一对应。。例如，顺序号为1的病人被分配到实验组2；顺序号为2的病人被分配到实验组1,依次类推。 向量“summary”为（10,10,10），代表各有10个病人分配到三个组中。

##### （3）Stratified Complete Randomization

<http://127.0.0.1:8000/randomization/StratifiedCompleteRandomization>

**输入参数**

|  |  |  |
| --- | --- | --- |
| **参数名** | **描述** | **备注** |
| N | 每层样本量 | 长度为层数的向量，每个元素对应一个层的样本数。  每层样本数应该是分组数K，区块长度和sum(alloRatio)的倍数。总样本量应该为每层样本量的和（sum(N).  例如，假设分为2组（K=2）， 每组比例为2:1 （alloRatio=（2,1））， 则N的每个元素都应该为2和2+1=3的倍数，即6的倍数，如6,12,24等。假设共有2层,每层人数均等，则一个N的范例为N=(12,12)。此时总样本量12+12=24。 |
| K | 分组数 | 例如K=2， 则表明有两个组。 |
| alloRatio | 入组人数之比例 | 长度是K的向量。假设K=2， 如果alloRatio=(1:1),则为  均衡随机。如果alloRatio=(2:1),则分配到group1和到group2的比例为2:1。 |
| seed | 随机种子编号 | 相同的随机种子编号可以得到相同的随机化结果；不同种子编号会得到不同随机化结果。这个选项可以不开放给用户。 |

POST参数示例：

{"N": [20,30,20], "K": 2, "alloRatio": [1, 1], "seed": 4}

在这个例子里，一共有三个分层，第一层有20个病人，第二层有30个病人，第三层有20个病人。我们需要将这些病人分别随即分配到2个组里，分配比例是1：1（均衡随机），随机数种子编号为4.

**输出参数**

|  |  |  |
| --- | --- | --- |
| **参数名** | **描述** | **备注** |
| patientID | 病人顺序号 | 长度为sum(N)的向量，第i个元素代表第i个病人的顺序号 |
| strata | 层编号 | 长度为N的向量，第i个元素代表第i个病人的所属的层编号。 |
| group | 病人分组 | 长度为N的向量，第i个元素代表第i个顺序号的病人被分配到的实验组编号。实验组编号为1， 2，…,K |
| summary | 分组总结 | 为K列J行的矩阵，其中K为分组数，J为层数。第i列第j行的数字代表分配到第j层分配到第i组的人数。 |
| status | 程序状况指示 | 返回“success”随机化成功。如果返回“failure”, 程序会停止运行，并显示错误提示信息。 |

结果：

{"patientID":["1","2","3","4","5","6","7","8","9","10","11","12","13","14","15","16","17","18","19","20","21","22","23","24","25","26","27","28","29","30","31","32","33","34","35","36","37","38","39","40","41","42","43","44","45","46","47","48","49","50","51","52","53","54","55","56","57","58","59","60","61","62","63","64","65","66","67","68","69","70"],"strata":["1","1","1","1","1","1","1","1","1","1","1","1","1","1","1","1","1","1","1","1","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","3","3","3","3","3","3","3","3","3","3","3","3","3","3","3","3","3","3","3","3"],"group":["2","1","1","2","2","1","1","1","1","2","2","2","2","1","2","2","2","1","1","1","1","1","2","2","1","1","1","1","1","2","1","1","2","2","1","2","1","1","2","1","1","1","2","2","1","2","2","1","2","2","1","2","2","2","2","1","2","1","2","2","2","2","1","2","2","2","1","2","1","2"],"summary":[[10,10],[17,13],[6,14]],"summary\_rowname":["strata-1","strata-2","strata-3"],"summary\_colname":["group.1","group.2"],"status":"success"}

这个例子里，status为“sucess”,说明算法成功执行。分组结果 “group“，层编号“strata”和病人顺序号“patientID”按顺序一一对应。例如，顺序号为1的病人属于第1层，被分配到实验组2。顺序号为21的病人属于第2层，被分配到实验组1,依次类推。 矩阵“summary”为分组总结。其中第二行第一列为17第二列为13，代表第二层里被分配到第1组的人数为17，分配到第2组的人数为13。

##### （4）Stratified Block Randomization

<http://127.0.0.1:8000/randomization/StratifiedBlockRandomization>

**输入参数**

|  |  |  |
| --- | --- | --- |
| **参数名** | **描述** | **备注** |
| N | 每层样本量 | 长度为层数的向量，每个元素对应一个层的样本数。每层样本数应该是分组数K，区块长度和sum(ratio)还有blocksize的倍数。总样本量应该为每层样本量的和（sum(N)）.  假设一共有2层，每层都分为2组（K=2），入组比例为2:1，区块长度为5（blocksize=5）， 则每层的样本数都应为2，2+1=3和5的倍数，即30的倍数，例如30,60,90。假设一共有两层，每层样本量一样，则一个合格的N的输入为（30,30）。而总样本量为30+30=60. |
| K | 分组数 | 例如K=2， 则表明有两个组。 |
| alloRatio | 入组人数之比例 | 长度是K的向量。假设K=2， 如果alloRatio=(1:1),则为  均衡随机。如果alloRatio=(2:1),则分配到group1和到group2的比例为2:1。 |
| blookSize | 区块长度。 | 目前不支持随机区组长度 |
| seed | 随机种子编号 | 相同的随机种子编号可以得到相同的随机化结果；不同种子编号会得到不同随机化结果。这个选项可以不开放给用户。 |

POST参数示例：

{"N": [20,30,20], "K": 2, "alloRatio": [1, 1], "blocksize": 2, "seed": 4}

在这个例子里，一共有三个分层，第一层有20个病人，第二层有30个病人，第三层有20个病人。我们需要将这些病人分别随即分配到2个组里，分配比例是1：1（均衡随机），随机数种子编号为4，区组长度为2.

**输出参数**

|  |  |  |
| --- | --- | --- |
| **参数名** | **描述** | **备注** |
| patientID | 病人顺序号 | 长度为sum(N)的向量，第i个元素代表第i个病人的顺序号 |
| strata | 层编号 | 长度为N的向量，第i个元素代表第i个病人的所属的层编号。 |
| group | 病人分组 | 长度为N的向量，第i个元素代表第i个顺序号的病人被分配到的实验组编号。实验组编号为1， 2，…,K |
| summary | 分组总结 | 为K 乘以J的矩阵，其中K为分组数，J为层数。第i列第j行的数字代表分配到第j层分配到第i组的人数。 |
| status | 程序状况指示 | 返回“success”随机化成功。如果返回“failure”, 程序会停止运行，并显示错误提示信息。 |

结果：

{"patientID":["1","2","3","4","5","6","7","8","9","10","11","12","13","14","15","16","17","18","19","20","21","22","23","24","25","26","27","28","29","30","31","32","33","34","35","36","37","38","39","40","41","42","43","44","45","46","47","48","49","50","51","52","53","54","55","56","57","58","59","60","61","62","63","64","65","66","67","68","69","70"],"strata":["1","1","1","1","1","1","1","1","1","1","1","1","1","1","1","1","1","1","1","1","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","2","3","3","3","3","3","3","3","3","3","3","3","3","3","3","3","3","3","3","3","3"],"group":["1","2","2","1","1","2","2","1","2","1","1","2","1","2","1","2","1","2","2","1","2","1","1","2","2","1","2","1","2","1","2","1","1","2","2","1","2","1","1","2","2","1","1","2","2","1","1","2","1","2","2","1","1","2","1","2","1","2","1","2","1","2","2","1","1","2","2","1","2","1"],"summary":[[10,10],[15,15],[10,10]],"summary\_rowname":["strata-1","strata-2","strata-3"],"summary\_colname":["group.1","group.2"],"status":"success"}

这个例子里，status为“sucess”,说明算法成功执行。分组结果 “group“，层编号“strata”和病人顺序号“patientID”按顺序一一对应。例如，顺序号为1的病人属于第1层，被分配到实验组1。顺序号为21的病人属于第2层，被分配到实验组2,依次类推。 矩阵“summary”为分组总结。其中第二行第一列为15第二列为15，代表第二层里被分配到第1组和第二组的人数都为15。

##### （5）Minimization Randomization

<http://127.0.0.1:8000/randomization/MinimizationRandomization>

**输入参数**

|  |  |  |
| --- | --- | --- |
| **参数名** | **描述** | **备注** |
| nlevel | 预后因素水平数 | 长度为J的向量，其中J为预后因素的个数。第j个元素代表第j个预后因素的水平数。假设我们有2个预后因素，性别和年龄组。性别有2个水平，年龄组有3个水平，则J=2，nlevel=(3,2) |
| K | 分组数 | 例如K=2， 则表明有两个组。 |
| alloRatio | 入组人数之比例 | 长度是K的向量。假设K=2， 如果alloRatio=(1:1),则为  均衡随机。如果alloRatio=(2:1),则分配到group1和到group2的比例为2:1。 |
| seed | 随机种子编号 | 相同的随机种子编号可以得到相同的随机化结果；不同种子编号会得到不同随机化结果。这个选项可以不开放给用户。 |
| weight | 预后因素权重 | 向量，长度应和nlevel 一致。向量里的每个元素对应每个预后因素的权重，权重大的代表重要。 |
| markerNew | 新病人信息 | 包含新病人信息的矩阵，列数为J+2，行数为新数入组病人数。第1列为病人顺序号（"patientID"），第2到第J+1列为预后因素（"F1"，"F2"，"F3"），第J+2列为组号（"treatment"），这里设置为null. |
| markerOld | 旧病人的信息 | 包含旧病人信息（包括实际分组）的矩阵，列数为J+2，行数为所有病人个数（包括新入组的和以前已经入组的）。第1列为病人顺序号（"patientID"），第2到第J+1列为预后因素（"F1"，"F2"，"F3"），第J+2列为实际分组号（"treatment"）。  进行初次分配（第一批病人，无历史数据时），将这个设置为null |

POST参数示例：

1. 无历史数据（初次计算）输入：

{

"alloRatio": [1,2,1],

"nlevel": [2,2,3],

"seed": 1,

"markerOld": { "patientID": ["a"], "F1": [1], "F2": [2], "F3": [2], "treatment": [ "2"] },

"markerNew": { "patientID": ["b","3", "4"], "F1": [2, 1, 1], "F2": [1,2, 1], "F3": [3, 1, 1], "treatment": [null,null,null ] }

}

这个例子中，我们考虑三个预后因素，第一个因素有2个水平，第二个因素有2个水平，第三个因素有3个水平。我们需要把病人分配到3个组，比例为1:2:1。第一批病人有1个，序列号为。他们的三个预后因素水平分别为（1,2,2），则存放新病人信息的markerNew矩阵应该为如下格式：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| patientID | F1 | F2 | F3 | treatment |
| a | 1 | 2 | 2 | null |

注意，此时markerNew实际为5列1行，灰色标注的为列名。

**输出参数**

|  |  |  |
| --- | --- | --- |
| **参数名** | **描述** | **备注** |
| patientID | 病人顺序号 | 行数为所有病人个数（包括新入组的和以前已经入组的）。 |
| group | 病人分组 | 行数为所有病人个数（包括新入组的和以前已经入组的）。第i个元素代表第i个顺序号的病人被分配到的实验组编号。实验组编号为1， 2，…,K |
| summary | 分组总结 | K列sum(nlevel) 行的矩阵，其中K为分组数，sum(nlevel)为预后因素水平数的总和。第i列第j行的数字代表分配到第j个预后因素水平分配到第i组的人数。这个表格的具体解读见这个章节最后。 |
| status | 程序状况指示 | 返回“success”随机化成功。如果返回“failure”, 程序会停止运行，并显示错误提示信息。 |

结果：

{"patientID":["a"],"F1":["1"],"F2":["2"],"F3":["2"],"group":["2"],"summary":[[0,1,0],[0,0,0],[0,0,0],[0,1,0],[0,0,0],[0,1,0],[0,0,0]],"summary\_rowname":["F1.1","F1.2","F2.1","F2.2","F3.1","F3.2","F3.3"],"summary\_colname":["group.1","group.2","group.3"],"status":"success"}

输出结果中，patientID,F1,F2,F3都和输入的markerNew矩阵的相应列一致，而treatment从null变成了2，说明这个病人被分到了第2组，

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| patientID | F1 | F2 | F3 | treatment |
| a | 1 | 2 | 2 | 2 |

除了新病人的分组结果，系统还会输出分组总结矩阵Summary。这个矩阵有3列（因为共有3个分组）7行（预后因素一共有2+2+3=7个水平），灰色标注的是行名称或者列名称。

|  |  |  |  |
| --- | --- | --- | --- |
|  | Group.1 | Group.2 | Group.2 |
| F1.1 | 0 | 1 | 0 |
| F1.2 | 0 | 0 | 0 |
| F2.1 | 0 | 0 | 0 |
| F2.2 | 0 | 1 | 0 |
| F3.1 | 0 | 0 | 0 |
| F3.2 | 0 | 1 | 0 |
| F3.3 | 0 | 0 | 0 |

这个表格说明，预后因素F1的第一个水平F1.1，预后因素F2的第1个水平（F2.2），预后因素F3的后两个水平（F3.2），均有一个病人被分配到第二个实验组。

B) 有历史数据输入

{

"alloRatio": [1,2,1],

"nlevel": [2,2,3],

"seed": 1,

"markerOld": { "patientID": ["a"], "F1": [1], "F2": [2], "F3": [2], "treatment": [ "2"] },

"markerNew": { "patientID": ["b","3", "4"], "F1": [2, 1, 1], "F2": [1,2, 1], "F3": [3, 1, 1], "treatment": [null,null,null ] }

}

和无历史数据输入类似，我们同样要输入分配比alloRatio，预后因素水平数nlevel，随机数种子号seed及新病人信息矩阵markerNew。假设新进组3个病人，顺序号分别为b,3和4，对应三个预后信息为(2,1,3),（1,2,1）和（1,1,1），则markerNew矩阵为

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| patientID | F1 | F2 | F3 | treatment |
| b | 2 | 1 | 3 | Null |
| 3 | 1 | 2 | 1 | Null |
| 4 | 1 | 1 | 1 | Null |

除了新进组病人的信息，我们还需要输入之前入组的所有病人的信息，放在markerOld矩阵里.假设之前只有一个病人a入组，分组结果为2,则markerOld矩阵如下：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| patientID | F1 | F2 | F3 | treatment |
| a | 1 | 2 | 2 | 2 |

结果：

{"patientID":["a","b","3","4"],"F1":["1","2","1","1"],"F2":["2","1","2","1"],"F3":["2","3","1","1"],"group":["2","2","2","1"],"summary":[[1,2,0],[0,1,0],[1,1,0],[0,2,0],[0,1,0],[ 0,1,0],[ 0,1,0]],"summary\_rowname":["F1.1","F1.2","F2.1","F2.2","F3.1","F3.2","F3.3"],"summary\_colname":["group.1","group.2","group.3"],"status":"success"}

输出结果中包含了所有病人的信息,包括之前已经入组的病人的信息。这个例子里面有4个人，病人a,b,3和4。

新进入个病人3的分组看下面矩阵的后3行。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| patientID | F1 | F2 | F3 | treatment |
| a | 1 | 2 | 2 | 2 |
| b | 2 | 1 | 3 | 2 |
| 3 | 1 | 2 | 1 | 2 |
| 4 | 1 | 1 | 1 | 1 |

除了新病人的分组结果，系统还会输出所有病人的分组总结矩阵Summary。这个矩阵有3列（因为共有3个分组）7行（预后因素一共有2+2+3=7个水平），灰色标注的是行名称或者列名称。

|  |  |  |  |
| --- | --- | --- | --- |
|  | Group.1 | Group.2 | Group.2 |
| F1.1 | 1 | 2 | 0 |
| F1.2 | 0 | 1 | 0 |
| F2.1 | 1 | 1 | 0 |
| F2.2 | 0 | 2 | 0 |
| F3.1 | 0 | 1 | 0 |
| F3.2 | 0 | 1 | 0 |
| F3.3 | 0 | 1 | 0 |

这个表格说明，预后因素F1的第一个因素（F1.1）有一个病人分配到第一组（group.1），有2个病人分配到第二组（group.2）。预后因素F2的第一个因素（F2.1）有一个病人分配到第一组（group.1），有一个病人分配到第二组（group.2）。其他以此类推。

如何知道分配得是否均衡？

我们看一个大样本的例子。假设样本量为1000，需要均衡分配到3个组，同样考虑3个预后因素，三个预后因素的水平数分别为（2,2,3）。假设有3个预后因素在病人中均匀分布。

最小化随机算法最终的分配结果是

group.1 group.2 group.3

F1.1 173 174 172

F1.2 162 161 162

F2.1 174 174 172

F2.2 161 161 162

F3.1 117 118 117

F3.2 103 104 105

F3.3 115 113 112

从总结表可以看到，每个预后因素的每个水平，分配到3个实验组的人数基本均衡（每一行的三个数非常接近1：1:1）。这说明最小化随机算法在样本量较大的时候随机化效果很好。

权重如何体现在分配上？

看一个总等样本的例子，假设样本量为100，需要均衡分配到3个组，同样考虑3个预后因素，三个预后因素的水平数分别为（2,2,3）。假设有3个预后因素在病人中均匀分布。

如果等权重weight=(1,1,1)

group.1 group.2

F1.1 30 32

F1.2 21 20

F2.1 25 26

F2.2 26 26

F3.1 17 25

F3.2 17 16

F3.3 17 11

每个因素每个水平都略微不均衡。

如果我们设置权重为weight=(5,1,1),即第一个预后因素的权重最大，最优先平衡第一个预后因素，则结果为

group.1 group.2

F1.1 24 24

F1.2 27 28

F2.1 31 35

F2.2 20 17

F3.1 17 14

F3.2 18 16

F3.3 16 22

和上面的例子相比，明显第一个预后因素分布得较为均衡。