***Make Paired samples***

SPSS macros by Kirill Orlov

kior@akado.ru, ttnphns@gmail.com

<https://www.spsstools.net/en/KO-spssmacros>

All rights reserved

*Спаривание наблюдений двух выборок.* Между двумя выборками или множествами делается оптимальное спаривание наблюдений, такое, что сумма внутрипарных различий минимизируется. Используется «венгерский алгоритм» сопряжения элементов от двух массивов в пары.

# МАКРОС !KO\_HUNMATCH: ОПТИМАЛЬНОЕ СПАРИВАНИЕ ВЕНГЕРСКИМ АЛГОРИТМОМ

Version 2, Jun 2012 (Version 1, Apr 2010). Tested on SPSS 15, 17, 20.

!KO\_hunmatch prox= DISSIM /\*Значения в матрице - различия (DISSIM) или сходства (SIMIL)

/k= /\*Сколько создать пар; если не задать, то k = число столбцов входящей матрицы

/licmat= 'd:\exercise\licmx.sav' /\*Опционально: двоичная матрица размером с входящую

/\*запрещающая спаривать какие-л ряды с какими-л столбцами - путь/имя

/\*внешнего sav-файла в кавычках/апострофах

/rand= NO /\*Опционально: рандомизовать выбор в случае одинаковых значений:

/\*YES или NO (тж п/у).

Минимум надо задать PROX.

Венгерский алгоритм, известный еще как Kuhn-Munkres algorithm, вероятно, самый известный метод спаривания объектов из двух наборов, гарантирующий минимальное суммарное различие между спаренными объектами. Следовательно, этот метод годен в опросных и экспериментных исследованиях для конструирования массива, состоящего из парно сопряженных максимально похожих индивидов, массива, к какому можно применить статистические критерии для «парных» («сцепленных», «зависимых») выборок, вроде парного критерия Стьюдента.

Вообще, этот алгоритм является средством оптимального спаривания элементов двух множеств – что бы множества ни представляли. Одно множество может быть, например, работниками, а второе, скажем, видами работ. И имеется матрица значений, каждое из которых есть «цена» (что бы она ни значила) назначения данной работы данному работнику. Задача – распределить работы по работникам – один человек одна работа – так, чтобы сумма «цен» сделанных назначений была абсолютно минимальна.

Алгоритм дает это глобальное оптимальное решение, однако не является особо быстрым и потому не рассчитан на большие объемы данных, исчисляемые многими тысячами объектов в каждом из двух наборов. Но он несравненно быстрее, чем попытка решить ту же задачу комбинаторным перебором.

Макрос берет в качестве входящих прямоугольную матрицу, являющуюся рабочим массивом данных, в которой ряды – это одна выборка респондентов (или одно множество элементов), а столбцы это вторая выборка (или множество). Значения в матрице это различия или сходства между теми и другими (или же «цена» или «выигрыш» спаривания этих с теми). Опционально можно указать вторую матрицу, идентичную по размеру первой, которая играет роль «лицензирующей» по отношению к первой, в смысле – разрешает или запрещает делать те или иные спаривания. Макрос выдает новый рабочий массив с номерами рядов и столбцов, которые оказались алгоритмом спарены.

Внимание, если матрица не квадратная, расположите ее так, *чтобы число столбцов было меньше чем число рядов*. Аналогичное расположение должно быть и у опционально привлекаемой лицензирующей матрицы. Пропущенные данные где-либо не разрешены.

Классический венгерский алгоритм спаривает все столбцы с какими-либо рядами. Данный же макрос содержит модификацию, позволяющую заказать любое число пар *k* (я очень благодарен A. G. McDowell, <http://www.mcdowella.demon.co.uk/>, который предложил ее мне). Парадоксально, но чем меньше *k*, тем больше требуется ресурсов/времени; это объясняется особенностями модификации. Поэтому если у вас очень большая входящая матрица, а пар вам требуется, напротив, мало, алгоритм становится нецелесообразным. Тогда лучше применить иной алгоритм спаривания, например просто отобрать *k* таких наименьших значений в матрице различий, которые дают *k* полностью разных пар; в условиях, когда *k* мало по сравнению с размером матрицы, такой подход вполне вероятно даст глобальное, ане локальное, оптимальное спаривание (которое венгерский алгоритм дает всегда).

ПРИМЕР 1. Спаривание похожих респондентов двух подвыборок. Участвуют качественные и количественные признаки. Качественные признаки должны полностью совпадать у двух респондентов, чтобы разрешено было их спарить; количественные признаки должны различаться как можно меньше, чтобы двух респондентов спарить.

proximities sex marriage /view=case /measure= euclid /print= none /matrix= out(\*).

select if caseno\_>50.

execute.

delete variables rowtype\_ to varname\_ var51 to var120.

recode all (0=1) (else=0).

save outfile= 'd:\exercise\licmat.sav'.

dataset activate data.

proximities item1 item2 item3 /view=case /measure= euclid /print= none /matrix= out(\*).

select if caseno\_>50.

execute.

delete variables rowtype\_ to varname\_ var51 to var120.

dataset name prox.

!KO\_hunmatch prox= DISSIM /licmat= 'd:\exercise\licmat.sav'.

* В рабочем массиве данных (он назван *DATA*) всего 120 респондентов. Первую подвыборку составляют первые 50, вторую – остальные 70. Качественные переменные: *SEX, MARRIAGE* (пол – две категории, брачный статус 0 две категории); количественные: *ITEM1, ITEM2, ITEM3* (ответы на балльные пункты опросника).
* Команда PROXIMITIES выводит в новый безымянный рабочий массив матрицу евклидовых расстояний между всеми респондентами по переменным *SEX, MARRIAGE*. SELECT IF удаляет из нее первые 50 строк, а DELETE VARIABLES удаляет последние 70 столбцов, заодно с оформительскими переменными от *ROWTYPE\_* до *VARNAME\_*. В итоге имеем матрицу 70 респондентов 2-й подвыборки (строки) × 50 респондентов 1-й подвыборки (столбцы). Перекодируем все ровно нулевые расстояния в 1, а прочие в 0. Получили «лицензирующую матрицу», которую сохраняем как внешний файл.
* Возвращаемся к исходным данным *DATA* (DATASET ACTIVATE). PROXIMITIES выводит в новый безымянный рабочий массив матрицу евклидовых расстояний между всеми респондентами по переменным *ITEM1, ITEM2, ITEM3*. Далее превращаем эту квадратную матрицу в прямоугольную 70 × 50 точно так как делали выше. Это матрица близостей, основная входящая в макрос. Называем ее *PROX* (DATASET NAME).
* Пускаем макрос, указав, что значения в матрице есть различия и указав путь к внешней «лицензирующей» матрице. Макрос сделает оптимальное спаривание всех 50 респондентов-«столбцов» с самыми похожими на них 50 респондентами из числа 70 респондентов-«строк».

***Подкоманды***

**PROX**

Укажите PROX=DISSIM, если входящая матрица подобия – рабочий массив – содержит различия (т.е. «цену спаривания»). Укажите PROX=SIMIL, если она содержит сходства (т.е. «выигрыш спаривания»). Значения в матрице должны быть *положительные*. Если у вас есть неположительные значения, прибавьте положительное число, чтобы сделать их положительными (результат спаривания не зависит от прибавления константы).

**K**

Сколько пар «ряд-столбец» с минимальным суммарным внутрипарным различием создать. Укажите целое число от 1 до число столбцов. По умолчанию/незаданию K принимается как число столбцов.

**LICMAT**

Необязательная подкоманда, позволяющая ввести «лицензирующую» матрицу. Это должен быть внешний SAV-файл, представляющий собой массив того же размера, что матрица подобия. В лицензирующей матрице могут быть только значения 1 (спаривание ряда со столбцом разрешено) и 0 (спаривание ряда со столбцом запрещено).

**RAND**

Это подкоманда может понадобиться, если во входящей матрице подобия много одинаковых значений (ties), так что может статься, что на спаривание одновременно претендует более двух одинаково похожих объекта. При RAND=YES в такой ситуации выбор пары из претендующих будет случаен (и может меняться от пуска к пуску макроса с теми же входящими данными). При RAND=NO выбор пары из претендующих не случаен и зависит от порядка рядов и столбцов в матрице.

***Особые режимы***

Макрос не слушается взвешивания и не рассчитан на расщепленное состояние массива данных.