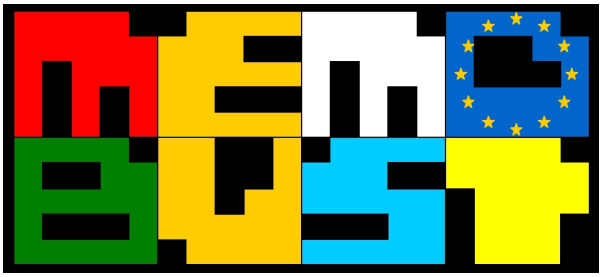

This module is part of the

Memobust Handbook

on Methodology of Modern Business Statistics

26 March 2014

Method: Unweighted Matching of Object Characteristics

Contents

General section

1. Summary

The method discussed in the present module is intended for matching two data sets on the basis of object characteristics. It is applied in case no object identifiers (of good quality) are available from both datasets. First the potentially matching records in the two data sets are identified. This requires a suitable metric and a cut-off value so that records that are too different are not considered as candidate matches. In the next step from these potential matches, a subset is computed that maximises the number of matches, under suitable constraints. The present module is based on Willenborg and Heerschap (2012). The reader is advised to read the theme module "Micro-Fusion – Object Matching (Record Linkage)" first before reading the present one. The method module "Micro-Fusion – Weighted Matching of Object Characteristics" should also be consulted, as the method described in the present module is a special case of the method discussed there. In particular it contains relevant information on metrics and graphs that are used in the present module.

2. General description of the method

The method consists of several steps. Here we discuss only the most important ones, leaving aside the preparatory steps. Part of the preparation is finding a cut-off value for the metric. This may actually take several iterations, as the cut-off value needs to be set in such a way that not too many candidate matches are generated, and not too few.

First step: The computation of matching candidates.

Using a metric and a cut-off value the set of records that can be matched is computed. To be a candidate match the distance of the two records involved must be smaller than the cut-off value.

Second step. The computation of the final matches from the candidate matches, under constraints. The constraint is that each record from both data sets can be at most in one match. The objective is to find as many matches as possible, obeying the constraint.

Both steps are typically done by computer. The second step formally requires the solution of an optimisation model. See the next subsection.

First select all records that are in a single match, and remove these from the candidate matches.

We can apply the optimisation to the remaining candidate matches, which is ambiguous in the sense that at least one of the records involved has two or more matches. In contrast to the weighted matching method, these candidate matches are of equal value, if they lead to the same total number of matches. A (random) choice must be made from the remaining candidates, or additional object characteristics are necessary. Depending on the number of remaining candidates, there is a risk of a false match, as for this method both matching records are intended to belong to the same object. This results in a number of mismatches.

We now give some comments on the approach taken in the method in the first and in the second step. If we denote a Hamming distance by d_H and we interpret the scores on the matching key as vectors of length n, then the matching criterion used here is in fact:

First step *(for Hamming metric* d_H*)*: for $\alpha \in A$, $\beta \in B$, α and β are matching candidates if and only if $d_H(\alpha, \beta) \leq k$.

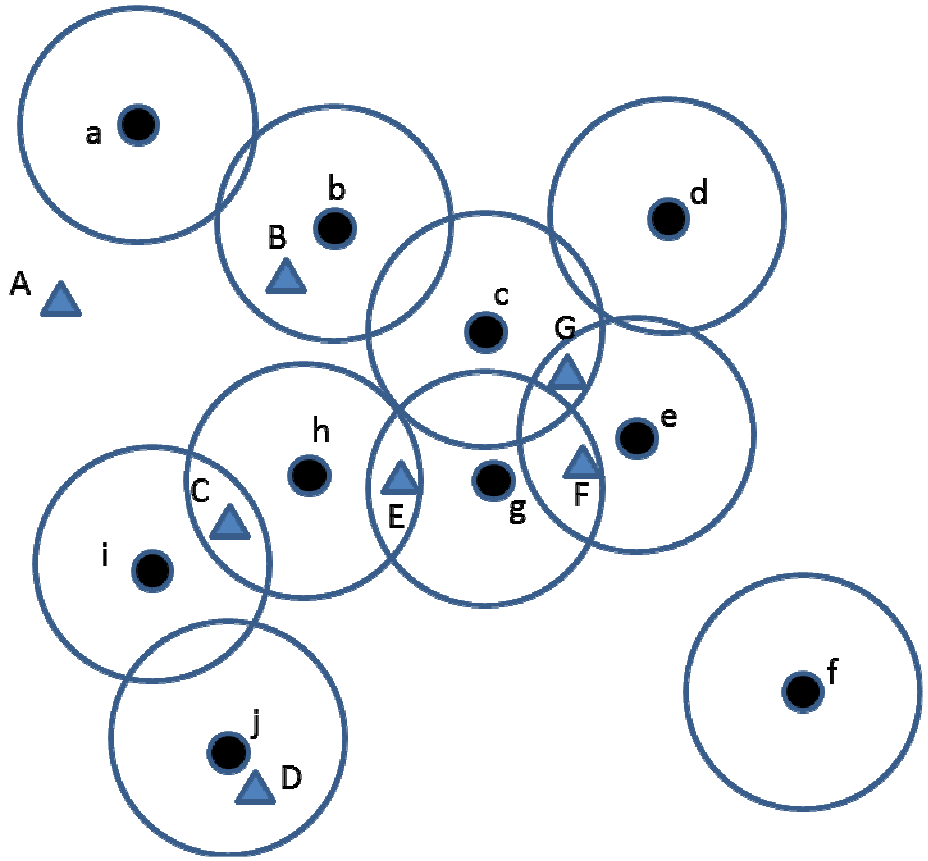

Figure 1. Records from two different files represented as points and neighbourhoods of the records from one of the files.

We can formulate this in such a way that all β from B that are inside a ball with radius k (using d_H as a metric) around α provide all matching candidates for α. See also Figure 1, illustrating the idea in case the neighbourhoods are in fact circles.

For each α in A, we can ascertain this in B. (Or vice versa, for each β in B, we can figure out which α in A are inside a circle with radius k around β. That produces the same result.) Note that, here, we only use whether a record is present in a circle around a 'point', not at what exact distance it is from that point. We could, in fact, use this distance as a matching weight: the smaller the distance the higher this weight. In the module "Micro-Fusion – Weighted Matching of Object Characteristics" this approach is described.

If we look at the above section critically, we can conclude that the selection of a Hamming distance is not essential for the approach taken; we could just as well have chosen another metric to arrive at a similar matching criterion. Therefore, based on a metric d, it is possible to formulate a matching criterion:

First step *(for general metric d)*: for $\alpha \in A$, $\beta \in B$, α and β are matching candidates if and only if $d(\alpha, \beta) \leq k$.

Once again, we can formulate this in such a way that all β from B that are inside a ball with radius k around α (measured using d), provides all matching candidates for α. For each α in A, we can ascertain this in B. (Or vice versa, for each β in B, we can figure out which α in A are inside a circle with radius k around β.) All of this produces a matching candidate graph (MC graph), where records from A and B are represented, and those which are potential matches are connected by an edge. See Figure 2.

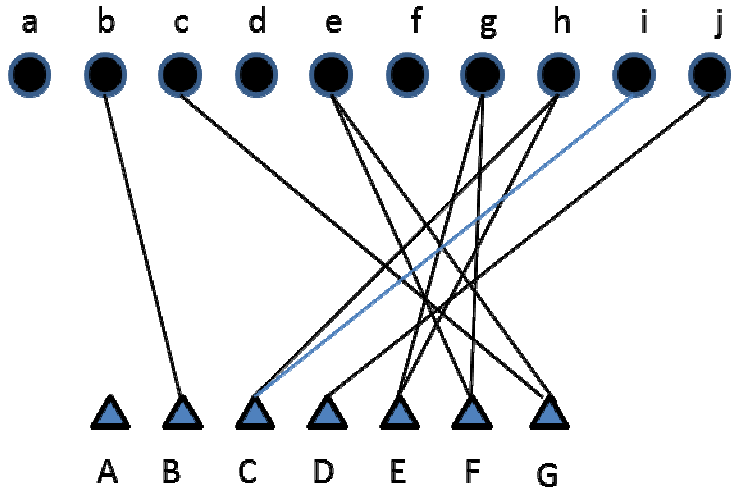

Figure 2. MC graph, representing the situation in Figure 1.

Second step: This involves the solution of the matching problem, in that matches have to be selected from candidate matches, under constraints. A typical constraint in this situation is that each record can be matched with at most one record in the other file. This is a well-known problem in combinatorial optimisation. It is discussed in books like Lawler (1976, Ch.5), Papadimitriou and Steiglitz (1998, Ch.10) or Nemhauser and Wolsey (1988, Ch. III.2), to which the interested reader is kindly referred.

3. Preparatory phase

The object characteristics common to both data sets to be matched are identified. It has to be decided if they are suitable for this type of matching; the number of potential matches should not be too big. A suitable metric for these variables should be found, as well as a suitable cut-off value. This requires some experimenting: with the cut-off value the number of potential matches can be controlled. In case the matching data sets are big special measures should be taken, such as blocking to create a manageable matching problem.

Now candidate matches can be found, from which the matches are to be calculated.

4. Examples – not tool specific

4.1 First example

An MC graph is formally a bipartite graph and is defined as follows for a matching problem where there are two data sets A and B, and a matching criterion K used on a matching key S. For an example of an MC graph without matching weights, see Figure 3.

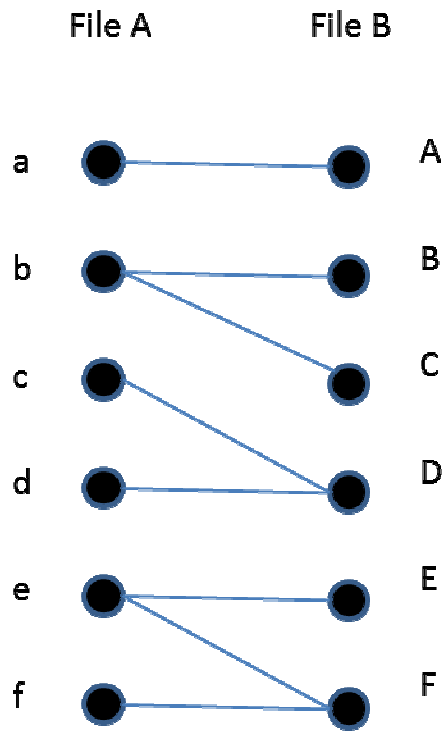

Figure 3. Example of MC-graph without matching weights.

We take data sets A and B to be sets of records. $G = (V, E)$ is the MC graph for this matching problem. The node set V is given by $V = A \cup B$ and the edge set E consist of the pairs $\{a, b\}$ where $a \in A, b \in B$ which furthermore satisfy the matching criterion K.

4.2 Second example

An MC graph is depicted in Figure 4. The edges indicate the matching candidates. The match $\{d, h\}$ is the only one that can be made unambiguously, separate from the matching criterion used. Depending on the matching criterion, more matches can be made. If this concerns a 1:1-matching, two additional matches are possible:

1. $\{a, g\}$ or $\{a, i\}$ (one of the two)
2. $\{c, f\}$ or $\{e, f\}$ (one of the two).

The choices in 1. and 2. can be made independently of each other.

In the case of an MC graph without weights, the candidate matches all count the same.

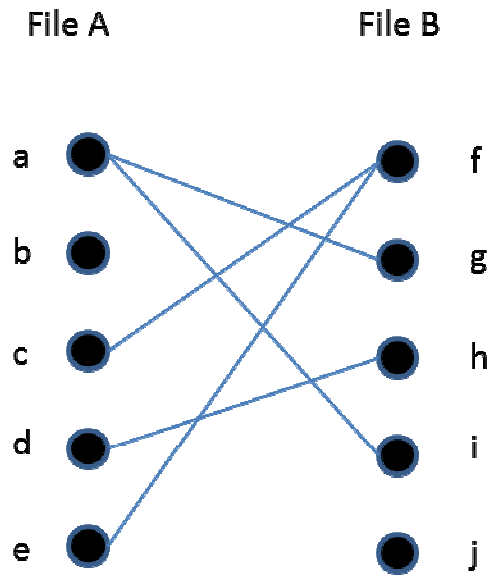

Figure 4. An MC graph

4.3 Third example

We consider two matching variables that are similar but not exactly the same. Specifically, we are talking about two age variables. One of the age variables, which specifies age in two-year classes, occurs in matching file A, and the other, which represents age in three-year classes, is found in matching file B. Depending on the reference times for each file (the time to which the data relate), we can make a connection between the age categories.

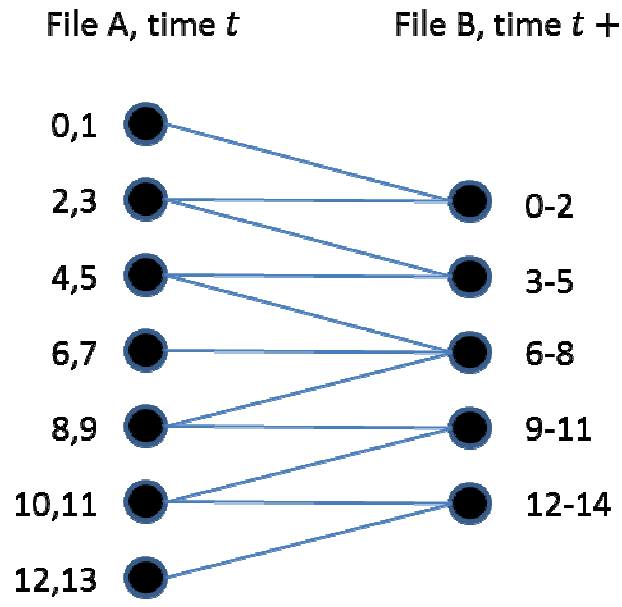

Figure 5. Two age variables and their relationship. One variable specifies age in two-year classes (file A) and the other specifies age in three (or four)-year classes (file B). The timestamps of the files differ by half a year.

Figure 5 shows a digraph that relates the age categories from the two data sets if the reference times are the same.

In practice, the reference times of two matching data sets do not have to be exactly the same. Indeed, it is more likely that they will differ. Moreover the data tend to relate to an interval rather than to a specific point in time. In Figure 5, a connection is made between the age categories if the reference times for the two data sets differ by a half a year. In this case, some people may have turned a year older in the interim period.

5. Examples – tool specific

6. Glossary

For definitions of terms used in this module, please refer to the separate "Glossary" provided as part of the handbook.

7. References

Lawler, E. L. (1976), *Combinatorial Optimization: Networks and Matroids*. Holt, Rinehart, Winston.

Nemhauser, G. L. and Wolsey, L. A. (1988), *Integer and Combinatorial Optimization*. Wiley, New York.

Papadimitriou, C. H. and Steiglitz, K. (1998), *Combinatorial Optimization: Algorithms and Complexity*. Dover, Mineola (NY).

Willenborg, L. and Heerschap, N. (2012), *Matching*. Contribution to the Methods Series, Statistics Netherlands, The Hague.

Specific section

8. Purpose of the method

The purpose is adding variables to a microdata set Ds-input1 from a second microdata set Ds-input2 for the same objects in both data sets. Records from two microdata sets are combined using a set of common object characteristics. It can be viewed as a special case of the weighted matching method (described in the method module "Micro-Fusion – Weighted Matching of Object Characteristics"), with equal matching weights (e.g., all equal to 1).

9. Recommended use of the method

1. In case common object identifiers of good quality are available in both input data sets object identifiers matching should be used. If this is not the case, the method in the current module could be an option, provided the next point holds.
2. Common object characteristic values of good quality are present in both matching data sets. Also, if similar variables are present in both data sets (with a slightly different domain) this method can be considered, depending on how much the domains differ. Observation errors can occur in the scores of these variables.
3. The data in both data sets should have (approximately) the same reference period. Otherwise there may be too many differences in the object characteristics common to both files to be matched, which has a negative effect on the matching quality.

10. Possible disadvantages of the method

1. In case both matching data sets are big, the method may be too slow. A special variant of the method described in this module, i.e., the one using blocking variables, may be able to do the job and obtain satisfactory results.
2. In case the reference periods for both data sets differ, so may the scores of some objects, due to the dynamics in the population. This may increase the chances for matching errors.

11. Variants of the method

1. The degree of matchability.

   1.1 Two records are either matching candidates or they are not. No third option possible.

   1.2 Two records are matching candidates, they are not, or they could be. In the case of the distance function d, we may decide for two records a and b as follows:

   - $d(a,b) \leq p$ if a and b are considered matching candidates.
   - $p < d(a,b) \leq q$ if a and b are doubtful matching candidates, and should be inspected by a specialist on the subject who should determine whether or not a and b are matching candidates.
   - $d(a,b) > q$ if a and b are not considered matching candidates.

The parameters p and q, with $p < q$, can be chosen so as to control how many matching candidates have to be inspected. The choices may be guided by the available capacity of specialists who can assess the doubtful cases. The parameters p and q are examples of cut-off values.

1.3 Or we can use a distances. For matching based on an object identifier, it is required that records have exactly the same score on the matching key used. We can also use this matching criterion in case of object characteristics, but the matching method this implies is less attractive here. We can relax this requirement and consider two records as candidate matches if the scores for at least k (parameter to be established) of the maximum n (length of the matching key = number of matching variables in the matching key) are the same. In fact, a metric is used here, the so-called Hamming distance.

2. The number of records in the output dataset:

   2.1 Each record of Ds-input1 is part of the output dataset (left outer join), or only matching records occur in the output dataset.

   2.2 Each record of Ds-input1 can occur more than once in the output dataset, or can occur at most once in the output dataset.

   2.3 Optionally additional output data sets are provided containing the non-matching records of Ds-input1 and Ds-input2.

3. The optional allowance of duplicate records in Ds-input1 or in Ds-input2, as an error type. Otherwise duplicate records are not allowed, as a precondition of the matching method, and a preparatory process step has to delete duplicate records, before starting the matching method.

4. Optionally one or more blocking variables can be used to partition the datasets for matching into manageable subfiles (blocks).

12. Input data

1. Ds-input1. This is the primary input data set. It is a microdata set, to which additional variables will be added.

2. Ds-input2. This input data set that contains variables that will be added to Ds-input1.

13. Logical preconditions

1. Missing values

   1. The object characteristic values used in the matching should not contain missing values.

2. Erroneous values

   1.

3. Other quality related preconditions

   1.

4. Other types of preconditions

1. A condition for using this method is that common object characteristics are present in both matching data sets, based on which the match can be performed. We also allow the situation that two similar variables have a different domain, for example, with another category division (for example, age in five-year classes in one data set, and in ten-year classes in the other). We also accept that observation errors can occur in the scores of these variables.
2. In practice, the decision to work with a matching method that does not use matching weights is often connected with performance. If the data sets to be matched are large, these methods generally work faster than those with matching weights. However, one should expect the quality of the matches – in terms of missed or missing matches – to be lower in general.
3. Duplicate records are not allowed, unless a variant of the matching method is used that handles the duplicate records.

14. Tuning parameters

1. A metric.
2. Cut-off values for the metric.
3. In case of big files: blocking variables to partition the files into manageable sub files (blocks).
4. In case of the variant with a zone of doubt (see item 11): the parameters *p* and *q*.

15. Recommended use of the individual variants of the method

1.

16. Output data

1. Ds-output1: a microdata set containing all variables of Ds-input1, with variables added from Ds-input2.
2. Optional Ds-output2 containing all non-matching records from Ds-input1.
3. Optional Ds-output3 containing all non-matching records from Ds-input2.

17. Properties of the output data

1. The output data set contains all variables from Ds-input1, but with additional variables from Ds-input2, presumably for the same objects.

18. Unit of input data suitable for the method

Processing full data sets (internally blocking variables can divide a data set into smaller parts).

19. User interaction - not tool specific

1. Before matching the tuning parameters must be set by analysing the results for different values.
2. No user interaction during matching.

3. After matching the number of mismatches must be evaluated, and quality indicators (missing and missed matches).

20. Logging indicators

1. Number of non-matching records from Ds-input1.
2. Number of non-matching records from Ds-input2.
3. Time used.

21. Quality indicators of the output data

1. The number of mismatches or missed matches and the number of missed matches can be used as quality indicators. The quality of the matching method can be assessed based on the inspection of matches of test files. It is a labour intensive job to carry out. One must examine not only the matching candidates and the matches ultimately selected, but also any missed matches under various parameter settings. The quality indicators are influenced by the use of cut-off values and the use of blocking variables to stratify large data sets.

22. Actual use of the method

1.

Interconnections with other modules

23. Themes that refer explicitly to this module

1. Micro-Fusion – Object Matching (Record Linkage)
2. Micro-Fusion – Probabilistic Record Linkage

24. Related methods described in other modules

1. Micro-Fusion – Object Identifier Matching
2. Micro-Fusion – Weighted Matching of Object Characteristics
3. Micro-Fusion – Fellegi-Sunter and Jaro Approach to Record Linkage

25. Mathematical techniques used by the method described in this module

1.

26. GSBPM phases where the method described in this module is used

1. 5.1 Integrate data

27. Tools that implement the method described in this module

1.

28. Process step performed by the method

Adding variables to microdata set

Administrative section

29. Module code

Micro-Fusion-M-Unweighted Matching

30. Version history

Version	Date	Description of changes	Author	Institute
0.1	23-04-2012	first version	Leon Willenborg, Rob van de Laar	CBS (Netherlands)
0.2	02-07-2012	second version	Leon Willenborg, Rob van de Laar	CBS (Netherlands)
0.3	11-07-2013	third version	Leon Willenborg	CBS (Netherlands)
0.4	09-08-2013	revised version (using review comments)	Leon Willenborg	CBS (Netherlands)
0.5	18-11-2013	revised version (using EB review comments)	Leon Willenborg	CBS (Netherlands)
0.5.1	19-11-2013	preliminary release		
1.0	26-03-2014	final version within the Memobust project		

31. Template version and print date

Template version used	1.0 p 4 d.d. 22-11-2012
Print date	21-3-2014 17:57