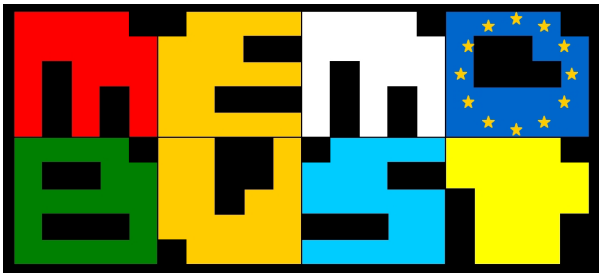

This module is part of the

Memobust Handbook

on Methodology of Modern Business Statistics

26 March 2014

Method: Object Identifier Matching

Contents

General section

1. Summary

The matching of records in two data sets is considered, on the basis of common object identifiers (key variables). The scores on these object identifiers are assumed to be of good quality in both data sets, though they need not be perfect. To understand the context of this type of matching in the handbook, the reader is referred to the theme module "Micro-Fusion – Object Matching (Record Linkage)". The present module is based on Willenborg and Heerschap (2012).

2. General description of the method

Matching based on an object identifier variable is the simplest way to match. Both matching data sets contain the same unique object identifier that is used as the matching key. The assumption is that the quality of the object identifier is sufficiently high; otherwise this matching method cannot be used effectively. Although we talk about an object identifier, it may actually exist of more than one variable, these are referred to as 'key variables'.

The basic principle is that a match is made if and only if a record from one dataset has exactly the same object identifier (key) value as another record from the second dataset. This type of matches is standard in databases, where it is called a 'join', or an 'equijoin'. See, e.g., Date (2000), Elmasri and Navathe (2004), or another book on relational databases.

In object identifier matching there are two input data sets that need to be matched. It would be perfectly acceptable if both data sets play interchangeable roles. In practice, however, often there is a primary input data set. The idea is to 'enrich' the records of this data set with values from the second input data set through matching. So the records in the primary input data set act as receptors and those of the second input data set, as donors. In this situation the roles of both input sets is not symmetric anymore.

Exact matching, or 'joining' as it is defined above, describes an ideal situation, in the sense that there are no errors in the object identifiers. In practice this ideal situation may not exist because some object identifier values are in fact erroneous, for instance because they were wrongly copied from another source. This is what makes the present method nontrivial, as the ideal case is very simple, conceptually. If the data sets are big there may be computational problems when matching the two files. In this case it may be partition the data sets into blocks that are manageable. When matching the records in a particular block of one file, only the records of a specific block in the other file is considered to find matching pairs. This blocking may result in missed matches.

2.1 Two steps

The assumption underlying the object identifier matching method in the ideal case is that the matching keys used in both data sets are error free. In practice, however, they need not be perfect. It is sufficient if they are of good quality. This allows that enough records can be matched, although there is a chance that mismatches or missed matches will occur.

First step: Records from both data sets are matched on the basis of exact equality of the object identifier scores. In this version of the method it is assumed that each record of the first data set has at most one match in the second dataset.

Second step: If some records of the first data set are not matched, this may be due to errors in the object identifier values. In a second step it is attempted to match any of the remaining records using the object identifier only.

The errors in the object identifiers may be due to typing errors: a wrong character was typed, two neighbouring characters were wrongfully interchanged, a character was wrongfully not typed (or deleted), or an extra character was wrongfully typed, etc. With this in mind it could be possible to correct for a missed match. This is attempted in this second step. The idea is to look among the missed matches and find pairs that are close in terms of the Levenshtein (or Damerau-Levenshtein) distance. See also Example 4.2 below. The distance is discussed in the method module on weighted matching of object characteristics in the handbook. If some records of the first data set do not match in the second step, they can still be part of the output data set, where the added variables are missing. Whether this is allowable depends on the variant of the method that is used.

3. Preparatory phase

The quality of the object identifier scores in both data sets should be assessed, to see if the Primary key matching method is applicable. If this seems to be the case, the first step can be attempted. Depending on the number of unmatched records one has to decide what to do next. Go ahead with the method or not. And if so, choose a suitable metric, depending on the variables in the object identifier.

4. Examples – not tool specific

Most of the examples below refer to Statistics Netherlands, but the issue at stake in each case can be generalised.

4.1 First example

The matching of enterprises from two surveys, which are both based on the General Business Register. In both data sets, the unit – the enterprise – is identified by an eight-digit business identification number (a BEID). The BEID is the object identifier on which matching takes place. If the BEIDs in both data sets are the same, then a match is made; if the BEIDs are not the same, then the units are not matched. For example, no account is taken of the fact that, during the processing procedure for the individual statistics, errors could have crept into the BEIDs. This check is also often difficult because, in many cases, there are no more object characteristics present, such as names and addresses.

4.2 Second example

Suppose that BEID is used as the object identifier, and you also have a complete list with BEIDs with at least some information about the businesses concerned. If a BEID is found that does not seem to be correct, then you could look in the neighbourhood of this number in the list. The idea here is that a mistake was made when copying the number, for example, two digits were interchanged, or a 5 was replaced by a 6 (or vice versa) or a 7 by a 1 (or vice versa), etc. If, for example, you search for all BEIDs with a Levenshtein distance of 1 or 2 from the given BEID, and also compare the associated

business attributes with the data in the dataset or register concerned, you could potentially find the correct BEID with the associated business attributes.

4.3 Third example

For privacy or data protection reasons external object identifiers can be replaced by secure internal object identifiers. The advantage is that it is impossible to link other external information to the records. This does prevent direct identification of units.

If E is the set of external keys and I the set of internally used keys, this replacement can be represented by a function $k : E \rightarrow I$, which should be injective.

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

Date, C. J. (2000), *An Introduction to Database Systems*, 7th edition. Addison-Wesley.

Elmasri, R. and Navathe, S. B. (2004), *Fundamentals of Database Systems*. Addison-Wesley.

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

Specific section

8. Purpose of the method

Enriching records in a given microdata set with information from a second microdata set.

9. Recommended use of the method

1. The method can be applied in case object identifiers (key variables) of good quality are available in both matching data sets.

10. Possible disadvantages of the method

1. If the quality of the object identifier values (key values) is not very high, the number of mismatches or missed matches may be substantial.

11. Variants of the method

1.

12. Input data

1. There are two input data sets, typically a primary input data set whose records are supposed to be 'enriched' by information from records from the second input data set through (object identifier) matching.

13. Logical preconditions

1. Missing values
    1. The object identifier values used in the matching are not supposed to be missing (too often). In case they are missing in some records the corresponding objects cannot be matched using object identifier matching (but possibly with object characteristics matching).
2. Erroneous values
    1. Errors in the object identifier (key variables) are allowed for some records in the input data files.
3. Other quality related preconditions
    1. The object identifiers used for matching are not supposed to change in time. If they do this could result in matching errors (mismatches or missed matches).
4. Other types of preconditions
    1.

14. Tuning parameters

1. In case the object identifier is (near) perfect (error-free) and the input data sets are small there is nothing to tune. Sorting on the object identifiers in both input files will yield an easy method to match.

2. In case the input files are big, blocking may be appropriate, that is partitioning the data sets into blocks. The choice of a good blocking variable is part of the tuning in this case.
3. In case the object identifier is not perfect (but good enough) matching on equality of object identifiers can still be carried out but may result in some mismatches or missed matches. In case more sophisticated matching criteria are used and metrics, one is in fact in the area of another type of matching, namely object characteristics matching. (An object identifier with quite some errors is more of an object characteristic.) See the method modules "Micro-Fusion – Unweighted Matching of Object Characteristics" and "Micro-Fusion – Weighted Matching of Object Characteristics" for the tuning parameters needed in those cases.

15. Recommended use of the individual variants of the method

1. In case of big input data sets the use of blocking may be applied, to split the data files in smaller blocks. This typically requires the use of one or more blocking variables.

16. Output data

1. A microdata set containing all variables of primary input data set, with variables added from the second input data set.
2. Optional data set containing all non-matching records from the primary input data set.
3. Optional data set containing all non-matching records from the second input data set.

17. Properties of the output data

1. There may be a set of matches (records form the primary input data set enriched with information from the second input data set) and a set of non-matches (from the both input data sets). In case the object key is not error-free the matches may contain false matches, and among the non-matches there may be missed matches.

18. Unit of input data suitable for the method

Objects are the units of input in this method. The objects are assumed to correspond with records in two data sets, conceptually not necessarily physically. The physical representation may be different, for instance when the objects are presented in normalised relational databases. Here the information about an object is physically scattered over various tables.

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 and assessment must be made of the number of mismatches and missed matches.

20. Logging indicators

1. Number of non-matching records from the primary input data set.

2. Number of non-matching records from the second input data set.
3. Time used for the matching.

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 may be a labour intensive job to carry out in case the matching files are big. First impressions may be obtained from inspecting a sample of the matched records, and of the non-matched records in the input data sets.

22. Actual use of the method

1. In case good quality object identifiers are available in the two files to be matched, object identifier matching is the preferred matching method.

Interconnections with other modules

23. Themes that refer explicitly to this module

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

24. Related methods described in other modules

1. Micro-Fusion – Unweighted Matching of Object Characteristics
2. Micro-Fusion – Weighted Matching of Object Characteristics

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-Object Identifier 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	29-10-2013	revised version (using EB review comments)	Leon Willenborg	CBS (Netherlands)
0.5.1	18-11-2013	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