**Matching Tool Logics** (06/16/2015)

The matching tool is a software that produces recommendations about matching new component records with existing persons in the database. It takes input from the component table in the database (the database file needs to be configured in the code). It also produces the results as a table. The prototype of the matching tool was developed to match records from the 1458 catasto table.

The matching tool is built modularly, and the algorithm includes several steps that are expected to be generalizable to other component tables. However, the logics of each step is adjustable for the implementation of various algorithms. I have developed two versions of the prototype that implement two basic algorithms. These algorithms should be further developed and refined through iterative experimentation.

For each component record, the tool retrieves from the database an initial set of plausible matches, and produces a *fitness score* for each one of them. Fitness scores range from 0 to 1 (and 1.1 in some cases), with higher score representing better fit. The overall fitness score is calculated based on the fitness scores for the names, the years and the neighborhood. In some future component tables, a guild fitness score should be added. Each of the specific fitness scores also gets a value between 0 and 1, and the overall fitness score is calculate using the following formula:

* Fit Score = Name Score x Neighborhood Score x Year Score

Fitness scores can be interpreted in different ways to produce recommendations, and the interpretation should be tailored to the needs of the users and improved through experimentation. For the two versions I developed I used two different logics of recommendation.

**Version 1** was developed to produce recommendations for the entire population of the 1458 catasto records. For each records one of three recommendations was produced:

* Definite identification – if there was exactly one match with 0.5 or greater fitness score.
* Multiple possibilities – if there were multiple matches with 0.5 or greater fitness score.
* No Match – if no match had 0.5 or greater fitness score.

86.1% of the catasto records were correctly identified by Version 1.

**Version 2** was developed to focus on records that can be positively identified with high certainty.

42.3% of the component records were identified with high certainty (fitness scores of 1 or 1.1), of which 98.1% were identified correctly. Many incorrect identifications had two matches with 1 or 1.1 fitness score. The performance of the tool can be improved to around 99% by disqualifying such cases.

**Last name standardization and fit**

* If the last name has a “,” in it, everything from the “,” and on is truncated.
* If the standardized last name is contained in a last name in Master (case insensitive), it matches. Otherwise, it doesn’t match.
* For example, “Ricasoli, da” in the Decima would match “Panzano, da/Ricasoli” in Master.

**Given name standardization and fit**

* Given names are standardized based on the Name Standardizing table (see separate document). So far, the code uses a simplified version of the table that I created (attached is the Excel version). The tool needs to be changed to work with the version that Junyan developed.
* If the name in component has ever been associated with the standardized name of a record in Master, they match. Otherwise, they don’t match.

**Overall name fit**

|  |  |  |  |
| --- | --- | --- | --- |
| **Last name** | **First Name** | **Middle Name** | **Fit Score** |
| + | + | + | 1 |
| + | + | Missing | .85 |
| Missing | + | + | .7 |
| - | + | + | .5 |
| + | - | + | .5 |
| + | + | - | .5 |
| Any other combination | | | 0 |

* If middle name 2 matches, add 0.1. If middle name 2 doesn’t match, subtract .1. If middle name 2 is missing, leave as is.

**Neighborhood fit**

* If person has ever lived in the neighborhood – 1.0
* If we only have quarter data and person has lived in the quarter – 0.8
* No neighborhood data – 0.65
* Neighborhood doesn’t match – 0.5

**Years fit**

This is the trickiest part and it needs further refinement. This version is still crude.

* If the master record has both birth and death years:
  + If component record is between birth and death year, score =1, otherwise, score = 0.
* If there is birth year and no death year:
  + If component record is between birth and latest record, or between birth and age 60, score = 1.
  + If component record is between ages 60 and 80, score = .75.
  + If component record is between ages 80 and 90, score =.5
* If there is a death year and no birth year:
  + If component record is between earliest record and death, or between 50 years before death and death, score = 1.
  + If component record is between 50 and 70 years before death, score = .75.
  + If component record is between 70 and 80 years before death, score = .75.
* If there are no birth and death years, but there are other dated records:
  + If component record is between 50 years before latest record and 50 years after earliest record, score = 1.
  + Otherwise, if component record is between 65 years before latest record and 65 years after earliest record, score = .75.
  + Otherwise, if component record is between 80 years before latest record and 80 years after earliest record, score = .5.
* If there are no dated records, score = 0