Non-Deterministic Linkage Overview
Matching of Fields

…or, should we consider Zbignew and Zbigneu the same first name?

Phonetic Transformation
Textual Similarity/Distance Measures
Phonetic Transformations

- Recode string variables in a manner that makes it possible to identify similarly sounding words

- Carl and Karl won’t match in straight deterministic comparison, but phonetically they are identical

- Can be an effective tool for matching records that do not agree due to minor data entry errors arising from multiple variations of some words or names
Soundex

• Soundex
  – 4 character phonetic representations of fields
  – First letter is first character of Soundex code
  – Otherwise, ignores vowels
  – Straightforward rules

• Example
  – “Christine” and “Christina” are both C623
  – “Christopher” is also C623
  – But “Chris” is C620
Soundex and NYSIIS

• Fast, easy, well understood
• Widely available in most software
• NYSIIS
  – New York State Identification and Intelligence System
  – Modest improvement over Soundex
Soundex and NYSIIS

• Issues
  – Problems w/non-traditional, non-English
    • Ethnic variations of Soundex exist
  – Best when one can have many false positives (say records match when they don’t) OR false negatives (say records not match when they do)
    • Preferable in combination w/other tools, multiple iterations, or non-exact (probabilistic) techniques
Metaphone and Double Metaphone

• Metaphone
  – Reduces text to 16 consonants
  – Variable length
  – Address additional limitations of Soundex
  – More complex rules, but widely available
Metaphone and Double Metaphone

• Double Metaphone
  – Reduces text to 12 consonants
  – Returns two results for lang/ethnic variations
  – Very complex
  – Slower than Soundex/NYSIIS
Simple Relative Comparisons

• Allow for range of differences in original data
• Many approaches, including probabilistic
• Relative comparisons: If criteria are met, pairing is considered a match
  – Birthweight +/- 100g
  – Birthweight < 1500g
  – Approximate dates (“June 2008”, or +/- 1 day)
  – Note that this is different from moving window approach to blocking
Textual Similarities

• Leading Characters
  – Simplest approach
  – Do two strings agree on the first n characters
  – Can be a quick and efficient tool for long strings
    • Particularly if used with other strategies
  – Example based on first 5 leading characters
    • Johnson and Johnson match
    • Johnson and Johansen do not
Similarities Indices

• A variety of similarity indices reflecting how similar two strings are

• Not making the comparison of two strings a dichotomous “match” vs. “not match”

• Some numeric value that reflects the degree to which the two fields are similar
  – Ranging from completely unrelated to very similar to identical
  – Often scaled 0 (no similarity) to 1 (identical)
Edit Distance

• **Available in FRIL**

• A variety of similarity indices reflecting how similar two strings are

• **Edit Distance**
  – How many changes are required to make two strings identical
  – Johnson to Johansen
    • Johansen → Johnsen → Johnson
    • Edit distance of 2
  – If this sounds familiar…
Jaro Distance

• Jaro and others have proposed similarity indices based on string length, number of common characters, and transpositions

• Jaro Distance…

\[ d_J = \frac{1}{3} \left[ \frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m - t}{m} \right] \]

…where m is the number of matches within a window of \( \frac{\max(|s_1|, |s_2|)}{2} - 1 \) characters of each other.
Jaro-Winkler Distance

- Jaro and Winkler Distance
  - Available in FRIL
  - Further modifies the Jaro Distance
  - Gives extra weight to agreement in the leading characters of a string

- $l$ is the length of the leading string (max 4?)
- $p$ is the additional weighting factor that one wants to give to this adjustments

\[d_{jw} = \max(1, d_j + lp(1 - d_j))\]
## Dice Scores

<table>
<thead>
<tr>
<th>Birth Certificate</th>
<th>Medicaid Enrollment</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>First</td>
</tr>
<tr>
<td>9</td>
<td>Zbignew</td>
</tr>
</tbody>
</table>

- Create bigrams for each string
- Birth certificate record
  - “br”, “re”, “ez”, “zi”, “in”, “ns”, “sk”, “ky”
- Medicaid enrollment record
  - “br”, “re”, “ez”, “zi”, “in”, “ns”, “sk”, “ki”
- Agree on 7 of the 8 bi-grams
Dice Scores

<table>
<thead>
<tr>
<th>Birth Certificate</th>
<th>Medicaid Enrollment</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>First</td>
</tr>
<tr>
<td>9</td>
<td>Zbignew</td>
</tr>
</tbody>
</table>

- Agree on 7 of the 8 bi-grams

\[
\text{DiceCoef} = 2 \left[ \frac{A_{\text{bigrams}} \cap B_{\text{bigrams}}}{A_{\text{bigrams}} + B_{\text{bigrams}}} \right]
\]

\[
\text{DiceCoef} = 2 \left[ \frac{7}{8+8} \right] = .875
\]
Q-Grams in FRIL

• Dice Scores *with a twist*
• Includes a minimal level of agreement, below which the score is automatically “0”
• Includes a ceiling level of agreement, above which the score is automatically “1”
• Can include a linearly extrapolated score for levels of agreement that fall between these values
Numeric and Date Distance

- Accept as match values falling within a given range (as raw score or percentage) of each other

- Matches – without linear approximation
  - 0: Outside the range
  - 1: Any value within the range

- Matches – with linear approximation
  - 0: Outside the range
  - 1: A “true” exact match
  - A linear approximation between 0 and 1 proportion to how close the match is within the range
Summary of Similarity Indices

• Many of these can be relatively computationally demanding
  – Both in programming and tech resources
  – Similar to probabilistic computational demands?

• A variety of situations where valuable
  – A field is necessary but susceptible to typos
  – Adds depth to comparisons of individual fields beyond weights reflecting match/not match dichotomy
Non-Deterministic Matching of Records

…or, should we consider Zbignew Brezinski and Zbignew Brezinsky the same person?

Non-Deterministic Linkage Methodology
Weighted Matches
Probabilistic
Machine Learning
Non-Deterministic Methods

• Two records do not have to agree across all fields in order to be matched
• A record in one file is compared to multiple records in another file
• Various methods then employed to determine whether each comparison reflects a true match
Background

Consider linking two data sets

- A: Birth Certificates
- B: Medicaid Enrollments

\[
\begin{bmatrix}
a_1 \\
a_2 \\
a_3 \\
\vdots \\
a_{n_A}
\end{bmatrix}
\quad
\begin{bmatrix}
b_1 \\
b_2 \\
b_3 \\
\vdots \\
b_{n_B}
\end{bmatrix}
\]
Background

Record Set-A  Record Set-B
Birth Certificates  Enrollments

\[
\begin{bmatrix}
  a_1 \\
a_2 \\
a_3 \\
\vdots \\
a_{n_A}
\end{bmatrix} \times \begin{bmatrix}
b_1 & b_2 & b_3 & \cdots & b_{n_B}
\end{bmatrix} \Rightarrow \begin{bmatrix}
a_1b_1 & a_1b_2 & a_1b_3 & \cdots & a_1b_{n_B} \\
a_2b_1 & a_2b_2 & a_2b_3 & \cdots & a_2b_{n_B} \\
a_3b_1 & a_3b_2 & a_3b_3 & \cdots & a_3b_{n_B} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
a_{n_A}b_1 & a_{n_A}b_2 & a_{n_A}b_3 & \cdots & a_{n_A}b_{n_B}
\end{bmatrix}
\]

• Many possible matches
Background

Record Set-A  Record Set-B
Birth Certificates  Enrollments

\[
\begin{bmatrix}
a_1 \\
a_2 \\
a_3 \\
\vdots \\
a_{n_A}
\end{bmatrix} \times \begin{bmatrix}
b_1 \\
b_2 \\
b_3 \\
\vdots \\
b_{n_B}
\end{bmatrix} \Rightarrow \begin{bmatrix}
a_1b_1 & a_1b_2 & a_1b_3 & \cdots & a_1b_{n_B} \\
a_2b_1 & a_2b_2 & a_2b_3 & \cdots & a_2b_{n_B} \\
a_3b_1 & a_3b_2 & a_3b_3 & \cdots & a_3b_{n_B} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
a_{n_A}b_1 & a_{n_A}b_2 & a_{n_A}b_3 & \cdots & a_{n_A}b_{n_B}
\end{bmatrix}
\]

• The truth is out there…
## General Principles

<table>
<thead>
<tr>
<th>Possible Matches</th>
<th>Set M: “True” Correct Matches</th>
<th>Set U: “True” Incorrect Matches</th>
</tr>
</thead>
<tbody>
<tr>
<td>Matrix of Possible Matches from previous slide, described below.</td>
<td>a₁b₃</td>
<td>a₁b₁, a₂b₂, ..., a₃b₁, a₂b₂, a₂b₃, a₃b₂, a₃b₃, ⋮</td>
</tr>
<tr>
<td></td>
<td>a₂bₙₜ</td>
<td>a₁b₁, a₂b₂, a₂b₃, a₃b₂, a₃b₃, ⋮</td>
</tr>
<tr>
<td></td>
<td>aₙₐb₂</td>
<td>a₁b₁, a₂b₂, a₂b₃, a₃b₂, a₃b₃, ⋮</td>
</tr>
</tbody>
</table>

Matrix of Possible Matches:

\[
\begin{bmatrix}
  a₁b₁ & a₁b₂ & a₁b₃ & \cdots & a₁bₙₜ \\
  a₂b₁ & a₂b₂ & a₂b₃ & \cdots & a₂bₙₜ \\
  a₃b₁ & a₃b₂ & a₃b₃ & \cdots & a₃bₙₜ \\
  \vdots & \vdots & \vdots & \ddots & \vdots \\
  aₙₐb₁ & aₙₐb₂ & aₙₐb₃ & \cdots & aₙₐbₙₜ \\
\end{bmatrix}
\]
### Estimating M and U

<table>
<thead>
<tr>
<th>Predicted Matches</th>
<th>Set M: “True” Correct Matches</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a_1b_3), (a_2b_nB), (a_3b_3), ...</td>
<td>(a_1b_3) &lt;br&gt; (a_2b_nB) &lt;br&gt; (a_3b_1) &lt;br&gt; ... &lt;br&gt; (a_nA) (b_2)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Predicted Non-Matches</th>
<th>Set U: “True” Incorrect Matches</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a_1b_1), (a_1b_2), (a_2b_1), (a_3b_1), (a_2b_3), (a_3b_2), ...</td>
<td>(a_1b_1), &lt;br&gt; (a_1b_2), ... &lt;br&gt; (a_1b_1), (a_2b_2), (a_2b_3), (a_3b_2), (a_3b_3), &lt;br&gt; ... (a_nA) (b_nB)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Uncertain Matches</th>
<th>?</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a_2b_2), (a_nA) (b_2), ...</td>
<td>?</td>
</tr>
</tbody>
</table>
## False Matches

### Predicted Matches

| $a_1 b_3$, $a_2 b_{nB}$, $a_3 b_3$, ... |

### Set M: “True” Correct Matches

| $a_1 b_3$, $a_2 b_{nB}$, $a_3 b_1$, $a_4 b_2$, ... |

### Predicted Non-Matches

| $a_1 b_1$, $a_1 b_2$, $a_2 b_1$, $a_3 b_1$, $a_2 b_3$, $a_3 b_2$, ... |

### Set U: “True” Incorrect Matches

| $a_1 b_1$, $a_1 b_2$, ..., $a_1 b_{nB}$, $a_2 b_2$, $a_2 b_3$, $a_3 b_2$, $a_3 b_3$, ... $a_{nA} b_{nB}$, |

### Uncertain Matches

| $a_2 b_2$, $a_{nA} b_2$, ... |
## False Non-Matches

<table>
<thead>
<tr>
<th>Predicted Matches</th>
<th>→</th>
<th>Set M: “True” Correct Matches</th>
</tr>
</thead>
<tbody>
<tr>
<td>( a_1 b_3, a_2 b_{n_B}, a_3 b_3, \ldots )</td>
<td></td>
<td>( a_1 b_3, a_2 b_{n_B}, a_3 b_1, \ldots, a_{n_A} b_2 )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Predicted Non-Matches</th>
<th>→</th>
<th>Set U: “True” Incorrect Matches</th>
</tr>
</thead>
<tbody>
<tr>
<td>( a_1 b_1, a_1 b_2, a_2 b_1, a_3 b_1, a_2 b_3, a_3 b_2, \ldots )</td>
<td></td>
<td>( a_1 b_1, a_1 b_2, \ldots, a_1 b_2, a_2 b_2, a_2 b_3, a_3 b_2, a_3 b_3, \ldots, a_{n_A} b_{n_B}, )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Uncertain Matches</th>
<th>→</th>
<th>?</th>
</tr>
</thead>
<tbody>
<tr>
<td>( a_2 b_2, a_{n_A} b_2, \ldots )</td>
<td></td>
<td>?</td>
</tr>
</tbody>
</table>
For a given level of false matches and non-matches, how do you obtain the smallest number of uncertain matches?

| Uncertain Matches | $a_2 b_2, a_n \overline{A} b_2, \ldots$ | $\rightarrow$ | ? |
Optimization Problem

• For a given level of false matches and non-matches, how do you obtain the smallest number of uncertain matches?

• In practice…
  – Minimize the false matches
  – Minimize the false nonmatches
  – Minimize the uncertain matches

• May not be possible to do all at the same time
Correct Solution...

Record Set-A  Record Set-B
Birth Certificates  Enrollments

\[
\begin{bmatrix}
  a_1 \\
  a_2 \\
  a_3 \\
  \vdots \\
  a_{n_A}
\end{bmatrix}
\times
\begin{bmatrix}
  b_1 & b_2 & b_3 & \cdots & b_{n_B}
\end{bmatrix}
\Rightarrow
\begin{bmatrix}
  a_1 b_1 & a_1 b_2 & a_1 b_3 & \cdots & a_1 b_{n_B} \\
  a_2 b_1 & a_2 b_2 & a_2 b_3 & \cdots & a_2 b_{n_B} \\
  a_3 b_1 & a_3 b_2 & a_3 b_3 & \cdots & a_3 b_{n_B} \\
  \vdots & \vdots & \vdots & \ddots & \vdots \\
  a_{n_A} b_1 & a_{n_A} b_2 & a_{n_A} b_3 & \cdots & a_{n_A} b_{n_B}
\end{bmatrix}
\]

- But life is rarely perfect...
Solution Challenges

Record Set-A
Birth Certificates

Record Set-B
Enrollments

\[
\begin{bmatrix}
  a_1 \\
  a_2 \\
  a_3 \\
  \vdots \\
  a_{n_A}
\end{bmatrix}
\times
\begin{bmatrix}
  b_1 & b_2 & b_3 & \cdots & b_{n_B}
\end{bmatrix}
\Rightarrow
\begin{bmatrix}
  a_1 b_1 & a_1 b_2 & a_1 b_3 & \cdots & a_1 b_{n_B} \\
  a_2 b_1 & a_2 b_2 & a_2 b_3 & \cdots & a_2 b_{n_B} \\
  a_3 b_1 & a_3 b_2 & a_3 b_3 & \cdots & a_3 b_{n_B} \\
  \vdots & \vdots & \vdots & \ddots & \vdots \\
  a_{n_A} b_1 & a_{n_A} b_2 & a_{n_A} b_3 & \cdots & a_{n_A} b_{n_B}
\end{bmatrix}
\]

• One mistake and the dominoes begin to fall…
Solution Challenges

Record Set-A  Record Set-B
Birth Certificates  Enrollments

$$\begin{bmatrix}
  a_1 \\
  a_2 \\
  a_3 \\
  \vdots \\
  a_{n_A}
\end{bmatrix} \times \begin{bmatrix}
  b_1 & b_2 & b_3 & \cdots & b_{n_B}
\end{bmatrix} \Rightarrow$$

Possible Matches

$$\begin{bmatrix}
  a_1 b_1 & a_1 b_2 & a_1 b_3 & \cdots & a_1 b_{n_B} \\
  a_2 b_1 & a_2 b_2 & a_2 b_3 & \cdots & a_2 b_{n_B} \\
  a_3 b_1 & a_3 b_2 & a_3 b_3 & \cdots & a_3 b_{n_B} \\
  \vdots & \vdots & \vdots & \ddots & \vdots \\
  a_{n_A} b_1 & a_{n_A} b_2 & a_{n_A} b_3 & \cdots & a_{n_A} b_{n_B}
\end{bmatrix}$$
Solution Challenges

Record Set-A  Record Set-B
Birth Certificates  Enrollments

\[
\begin{bmatrix}
  a_1 \\
  a_2 \\
  a_3 \\
  \vdots \\
  a_{n_A}
\end{bmatrix} \times \begin{bmatrix}
  b_1 & b_2 & b_3 & \cdots & b_{n_B}
\end{bmatrix} \implies
\]

Possible Matches

\[
\begin{bmatrix}
  a_1 b_1 & a_1 b_2 & a_1 b_3 & \cdots & a_1 b_{n_B} \\
  a_2 b_1 & a_2 b_2 & a_2 b_3 & \cdots & a_2 b_{n_B} \\
  a_3 b_1 & a_3 b_2 & a_3 b_3 & \cdots & a_3 b_{n_B} \\
  \vdots & \vdots & \vdots & \vdots & \vdots \\
  a_{n_A} b_1 & a_{n_A} b_2 & a_{n_A} b_3 & \cdots & a_{n_A} b_{n_B}
\end{bmatrix}
\]
## Estimating M and U

<table>
<thead>
<tr>
<th>Predicted Matches</th>
<th>→</th>
<th>Set M: “True” Correct Matches</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a_1 b_3, a_2 b_\text{nB}, a_3 b_3, \ldots)</td>
<td></td>
<td>(a_1 b_3, a_2 b_\text{nB}, a_3 b_1, \ldots, a_n A b_2)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Predicted Non-Matches</th>
<th>→</th>
<th>Set U: “True” Incorrect Matches</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a_1 b_1, a_1 b_2, a_2 b_1, a_3 b_1, a_2 b_3, a_3 b_2, \ldots)</td>
<td></td>
<td>(a_1 b_1, a_1 b_2, \ldots, a_1 b_1, a_2 b_2, a_2 b_3, a_3 b_2, a_3 b_3, \ldots, a_n A b_2, a_n A b_2, a_n A b_{\text{nB}}, \ldots)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Uncertain Matches</th>
<th>→</th>
<th>?</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a_2 b_2, a_n A b_2, \ldots)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Start by throwing the majority of possible matches into predicted non-matches through blocking
Blocking Techniques

• Typically begin by essentially eliminating the vast majority of possible matches
  – Automatically code as non-matches

• All-to-All Comparison
  – Every record in A compared to every record in B
  – Generally only practical in smaller databases
Blocking

- Possible matches must agree on a subset of specified fields
- Possible matches not agreeing on those fields automatically classified as non-matches
- Can dramatically reduce the number of possible comparisons to make
- Generally some form of blocking is necessary for computational efficiency
Blocking

- Selection of fields used for blocking is key
- The poorer the quality of the fields used in blocking, the more erratic the results
  - If data in blocking fields are random, results are meaningless
  - If data in blocking fields are “perfect”, results will still contain all correct matches
  - Quantify the upfront minimum/base error rate as the product of the corresponding $m$ probabilities (more later..)
Moving Window Blocking

• Records are sorted based on some combination of one or more fields

• Records in A are compared to similar records in B that fall within a specified window size
  – Sort by date of birth
  – Select window size of 30
  – If a baby in A was born on August 18, 2007 only consider the 30 records in B around that value
Moving Window Blocking

• Selection of fields for sorting is again key
  – Sorting on different fields may increase the separation of “correct” matches
  – Typos may dramatically impact windows

• Multiple iterations with different sort orders may be valuable

• Powerful tool when one wants to block on quantitative field with expected small errors (dates, birth weight…)
Comparison of Blocking Approaches

• If great confidence in a given field, traditional blocking may make the most sense
• If one wishes to use a quantitative variable (birthweight, date of birth) for blocking, moving window comparison probably makes the most sense
• Note that this is different than a distance measure for an individual field (e.g., a birth within 100g is considered a match)
Non-Deterministic Methods

Weighted matches
Probabilistic methods
Machine learning
Hold on Tight for a Minute…

• Winnie the Pooh Video
Non-Deterministic Methods

• Two records do not have to agree across all fields in order to be matched
  – Weights are used to quantify the likelihood that a pair of records are a true match

• A record in one file is compared to multiple records in another file
  – Using the weights, scores are calculated for possible matches that suggest whether it is correct (i.e., are the same person)
Cutoff Scores

• Cutoff scores
  – Above some value (e.g. “14”) conclude it is a true match
  – Below some value (“11”), conclude it is not a true match
  – Any possible match with a score between these two values is a manually reviewed

• Individual choice in cutting off low weights
Weighted Fields

• Match two data files based on SSN, First Name, Last Name, Date of Birth, …

• Agreement in some fields are given more “weight” than agreement in other fields
  – A match on SSN has a weight of 3
  – A match on First Name has a weight of 1
  – A match on Last Name has a weight of 2

• The weight is non-specific (i.e., does not change based on the values in the field)
Weighted Fields

• Sum the weights for each possible match
  – Agreement in different fields results in different sums
  – Larger sums reflect greater confidence of a match
  – Distribution evaluated and cutoff scores determined
  – Each possible match is compared to these cutoff scores to conclude whether a match, non-match, or review
Weighted Fields

• Determining values for the weights
  – Different fields may be a stronger indicator of a true match (ID number versus name)
  – Can assign “penalty” weights for non-matches
    • High quality field that *should* almost always agree
      – Subjective, EM algorithm, machine learning
• With *non-specific weights*, the same weight applies for regardless of the actual data values being matched
  • “Jones-Jones” gets the same weight as “Szapocznik-Szapocznik”
Probabilistic Matching

- In contrast, probabilistic matching takes into consideration the specific values in the fields being matched
  - Considers the quality of the data in the field
  - Open to further analysis of matching strength

<table>
<thead>
<tr>
<th>Birth Certificate</th>
<th>Enrollment Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>First</td>
</tr>
<tr>
<td>9</td>
<td>Zbignew</td>
</tr>
</tbody>
</table>
Probabilistic Matching

- More complicated (and expensive) strategy
- Still need to estimate some weights (m and u probs)

<table>
<thead>
<tr>
<th>Birth Certificates</th>
<th>Birth Defects Registry</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>First</td>
</tr>
<tr>
<td>9</td>
<td>Zbignew</td>
</tr>
</tbody>
</table>

50
Probabilistic Matching

<table>
<thead>
<tr>
<th>Birth Certificate</th>
<th>Enrollment Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>First</td>
</tr>
<tr>
<td>9</td>
<td>Zbignew</td>
</tr>
</tbody>
</table>

- Two records are compared on each of the specified fields.
- A weight—$w_i$—is calculated for each field in a potential match reflecting the strength of the agreement or disagreement.
Factors Influencing Likelihood of Match

• Reliability of data fields
  – Good quality data counts more than poor quality
    • High quality data suggests that fields should agree if a correct match
    • Low quality data suggests that even if fields don’t agree, it may still be a correct match
  – If a field is pure noise, correct matches will be random across the databases
  – Reflects the likelihood that fields would agree if it is a correct link
Factors Influencing Likelihood of Match

• Frequency of field values
  – The more common the value in a field, the greater the odds that records will be erroneously matched
  • A match based on the Zbignew is a good indicator of a match, even if there may be disagreement in other fields
  • A match based on the John is of less value, requiring matches on more fields to conclude its the same person
  • Rare values count more than common values
  – Reflects the likelihood that fields would agree if not a correct link
Factors Influencing Likelihood of Match

- Number of Matches
  - The greater the number of individuals in one database that also appear in the other database, the greater probability of linkage across databases.
  - If two databases are the same size, and every record in one has a match in the other, it is easier to infer that two records are the same individual.
  - If two databases have **no** individuals in common, the probability of a linkage across the databases is nil, regardless of how well two records agree.
Calculating Match Weights

<table>
<thead>
<tr>
<th>Birth Certificate</th>
<th>Enrollment Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>First</td>
</tr>
<tr>
<td>9</td>
<td>Zbignew</td>
</tr>
</tbody>
</table>

- **Weight Calculation**
  - **M-probability**
    - Probability that a field agrees if the pair is a correct match
  - **U-probability**
    - Prob that a field agrees if the pair is an incorrect match
    - Chance that a given field will agree randomly
    - $\approx$ the proportion of records with a specific value
### Probabilistic Matching

- If the field agrees, \( w_i \) is equal to ....

\[
w_i = \log_2 \left( \frac{m_i}{u_i} \right)
\]
## Probabilistic Matching

<table>
<thead>
<tr>
<th>Birth Certificate</th>
<th>Enrollment Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>First</td>
</tr>
<tr>
<td>9</td>
<td>Zbignew</td>
</tr>
</tbody>
</table>

- \( m_i \) for first name = .98, or 98% of the time, if it’s a correct match, the first names will agree
- \( u_i \) for Zbignew is .00001 is the probability of randomly getting two first names that are Zbignew

\[
w_{il} = \log_2 \left( \frac{m_i}{u_i} \right) = \log_2 \left( \frac{.98}{.00001} \right) = 16.58049
\]
Probabilistic Matching

- In cases where two records disagree on a specified field, \( w_i \) is equal to ….

\[
    w_i = \log_2 \left( 1 - \frac{m_i}{1 - u_i} \right)
\]
### Probabilistic Matching

- $m_i$ for last name = .96, or 96% of the time, if it’s a correct match, the last names will agree.

- $u_i$ for Brezinsky is .00003 is the probability of randomly getting two last names that are Brezinsky.

$$w_{i2} = \log_2 (\frac{(1 - m_i)}{(1 - u_i)}) = \log_2 (\frac{(1 - .96)}{(1 - .00003)}) = -4.64381$$
Calculating Match Weights

• A composite weight, $w_t$ calculated for each pair of records
  - The sum of weights across all fields used in linkage

$$w_t = \sum_{i=1}^{k} w_i$$

$$w_{it} = 16.58049 - 4.64381 = 11.93668$$

• Larger $w_t$ suggest a correct match,
• Smaller or negative $w_t$ suggest an incorrect match.
Probabilistic Matching

• Two fields disagree
  – m-probabilities generally come into play
  – How big of a hit do you take when last name doesn't agree across a possible match?

• Two fields agree
  – Differences in the u-probabilities that typically matter most
  – (e.g., last name of Smith versus Brezinski).
Probabilistic Match Weights

• Cutoff values for $w_t$ are determined and used to classify possible matches
  – Automatic matches
  – Manual review
  – Automatic rejection

• Traditionally, techniques end at this point
Probabilistic Match Weights

• Issues
  – $w_t$ values have no inherent meaning
    • No set range as to large or small $w_t$s
  – If multiple iterations are performed, cutoffs must be analyzed and determined for every iteration
Estimating Probabilities

- The total-weight required for two records to have a probability, $p$, of being a match is equal to...

$$ w_t = \log_2(p/(1 - p)) - \log_2 (E/(N_1N_2 - E)) $$

...and...

$$ \log_2 (E/(N_1N_2 - E)) $$ is the base 2 log of the odds of a random match
Estimating Probabilities

From this formula, it is possible to derive an equation for estimating $p$, the probability that any two records are a match, where...

$$p = \frac{\prod_{i=0}^{K} x_i}{\prod_{i=0}^{K} x_i + 1}$$

$x_0 = E/(N_1N_2 - E)$ odds of a random match,

$x_{i,i>0} = m_i/u_i$ if two fields agree, and...

$x_{i,i>0} = (1 - m_i)/(1 - u_i)$ if two fields do not agree
Sum of Weights ($w_t$) vs Probabilities

- **Sum of Weights ($w_t$)**
  - Requires repeatedly calculating $\log_2$’s
  - No inherent interpretability, must subjectively determine a “large” $w_t$ with each linkage

- **Probabilities**
  - Does not requires $\log_2$’s, and so improved speed with large linkage projects
  - More readily understood and interpreted criteria for determining whether to classify two records as being a match
Machine Learning

• Through a series of software-driven iterations, software “learns” which weights to use or which combinations of linkage fields are best

• May use any of these approaches
  – Most likely use weighted fields or probabilistic
Machine Learning

• Typically, use training datasets
  – Software fed training data with known solution regarding “true” matches
  – Algorithms match records based on initial sets of weights
  – Results compared to information regarding “true” matches to see if replicated
  – Modifies weights and re-runs with same or different training data
Machine Learning

• After various trials with training data, final set of weights determined
  – May see minimal modifications made to new trials
• Algorithm should now work with real data
• Issues
  – Identifying a training data set that reflects the nature, qualities, and issues in the data sets you wish to link
  – Complexity
  – Some approaches seek to skip need for training sets
Probabilistic Linkage Methods

• Some SAS programmers write their own code for probabilistic matching

• Software packages
  – Can be very expensive
  – Difficult to use
  – Some applications are available as freeware or shareware
## Choosing Probabilistic Software

<table>
<thead>
<tr>
<th>Program</th>
<th>OS</th>
<th>Initial $</th>
<th>Yearly $</th>
<th>Link Type</th>
<th>Desc</th>
<th>Audience</th>
<th>Organization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Automatch</td>
<td>Windows</td>
<td>$100,000</td>
<td>???</td>
<td>Probabilistic</td>
<td>GUI</td>
<td>Marketing</td>
<td></td>
</tr>
<tr>
<td>Generalized Record Linkage System (GRLS)</td>
<td>UNIX</td>
<td>$18,800</td>
<td>10%</td>
<td>Probabilistic</td>
<td>ORACLE</td>
<td>Health care</td>
<td>Stats Canada</td>
</tr>
<tr>
<td>LinkPro</td>
<td>Windows/Server</td>
<td>$1,455 / $1,190</td>
<td>None</td>
<td>Determ &amp; Prob</td>
<td>SAS</td>
<td>Health care</td>
<td>U. of Manitoba</td>
</tr>
</tbody>
</table>

- **Links**: same as LinkPro but freeware
- **FRIL**: also freeware, open source
Overall Summary of Linkage Methods

• Many tools and strategies available
• No single approach is perfect for every situation
• Factors to consider
  – Purpose of linkage
  – Nature of the data
    • Quality of fields used for linkage
    • Type of data (string, date, numeric) used for linkage
    • Resources available