13
String Comparator Metrics for Typographical Error

Many fields such as the first name and last name occasionally contain minor typographical variations or errors. Even with high-quality lists, such as the 1990 US Census and itsPES, there were areas of the United States in which 30% of the first names and 25% of the last names of individuals who were in fact matches did not agree exactly on a character-by-character basis. If we attempt to match two such records and at least one of the first names on the two records has a typographical variation, then we may fail to match two records that may indeed be matches. Using the terminology of computer science, we can consider these records or their components to be *strings* – that is, strings of alphanumeric characters. We need a practical method for dealing with such situations.

As the name indicates, string comparator metrics are used to compare two strings. In particular, they are used to determine how much alike the two strings are to each other. Common practice is to restrict the values of the metrics to the interval from zero to one; here, one indicates perfect agreement (the two strings are identical) and zero indicates that they are highly dissimilar, the extreme case being that they have no characters in common. These values are needed to adjust the likelihood ratios of the Fellegi–Sunter scheme to account for this partial agreement. In this work we focus on the string comparator metric introduced by Jaro and enhanced by Winkler. Current research in this area is described in Cohen, Ravikumar, and Fienberg [2003a, b].

13.1. Jaro String Comparator Metric for Typographical Error

Jaro [1972, see also 1989] introduced a string comparator metric that gives values of partial disagreement between two strings. This metric accounts for the lengths of the two strings and partially accounts for the types of errors – insertions, omissions, or transpositions – that human beings typically make when constructing alphanumeric strings. By *transposition* we mean that a character from one string is in a different position on the other string. For example, in comparing "sieve" to "seive," we note that "i" and "e" are transposed from one

string to the other. The string comparator metric also accounts for the number of characters the two strings have in common. The definition of *common* requires that the agreeing characters must be within half of the length of the shorter string. For example, "spike" and "pikes" would only have four characters in common because the "s's" are too far apart.

Specifically, let s_1 denote the first string, s_2 denote the second string, and let c denote the number of characters that these two strings have in common. Then, if $c > 0$, the Jaro string comparator metric is

$$\Phi_J(s_1, s_2) = W_1 \cdot \frac{c}{L_1} + W_2 \cdot \frac{c}{L_2} + W_t \cdot \frac{(c - \tau)}{c}$$

where

W_1 is the weight assigned to the first string,
W_2 is the weight assigned to the second string,
W_t is the weight assigned to the transpositions,
c is the number of characters that the two strings have in common,
L_1 is the length of the first string,
L_2 is the length of the second string, and
τ is the number of characters that are transposed.
We require that the weights sum to 1: $W_1 + W_2 + W_t = 1$.
Finally, if $c = 0$, then $\Phi_J(s_1, s_2) = 0$.

Example 13.1: Using the Jaro string comparator metric on Higvee versus Higbee

Let the first string, s_1, be "Higbee" and the second string, s_2, be "Higvee," and let all of the weights be equal to 1/3. Find the value of the Jaro string comparator metric for these two strings.

Solution

Because the two have five of six letters each in common, we have

$$L_1 = L_2 = 6, \; c = 5, \text{ and } \tau = 0.$$

Hence,

$$\Phi_J(s_1, s_2) = W_1 \cdot \frac{c}{L_1} + W_2 \cdot \frac{c}{L_2} + W_t \cdot \frac{(c - \tau)}{c} = \left(\frac{1}{3}\right) \cdot \left(\frac{5}{6}\right) + \left(\frac{1}{3}\right) \cdot \left(\frac{5}{6}\right) + \left(\frac{1}{3}\right) \cdot \left(\frac{5 - 0}{5}\right) = \frac{8}{9}.$$

Example 13.2: Using the Jaro string comparator metric on Shackleford versus Shackelford

Let the first string, s_1, be "Shackleford" and the second string, s_2, be "Shackelford" and let all of the weights be equal to 1/3. Find the value of the Jaro string comparator metric for these two strings.

Solution

Because two letters "l" and "e" are transposed, we have

$$L_1 = L_2 = 11, \; c = 11, \text{ and } \tau = 2.$$

Hence,

$$\Phi_J(s_1, s_2) = W_1 \cdot \frac{c}{L_1} + W_2 \cdot \frac{c}{L_2} + W_t \cdot \frac{(c-\tau)}{c} = \left(\frac{1}{3}\right) \cdot \left(\frac{11}{11}\right) + \left(\frac{1}{3}\right) \cdot \left(\frac{11}{11}\right) + \left(\frac{1}{3}\right) \cdot \left(\frac{11-2}{11}\right) = \frac{31}{33}.$$

13.2. Adjusting the Matching Weight for the Jaro String Comparator

To take into account the partial weight from the Jaro string comparator metric, Winkler [1990] suggested redefining the weights when there is agreement, as the adjusted weight

$$w_{am} = \begin{cases} w_a & \text{if } \Phi_J = 1 \\ \max\left[w_a - \{(w_a - w_d) \cdot (1 - \Phi_J) \cdot (4.5)\}, w_d\right] & \text{if } 0 \le \Phi_J < 1 \end{cases}$$

where *the full agreement weight* is $w_a = \log_2\left(\frac{m}{u}\right)$ and the *full disagreement weight* is $w_d = \log_2\left(\frac{1-m}{1-u}\right)$. The constant 4.5 controls how quickly decreases in partial agreement factors force the adjusted weight to the full agreement weight. We can write the full agreement weight, w_a, for example, as

$$w_a = \log_2\left(\frac{m}{u}\right) = \log_2\left(\frac{P[\Phi_J = 1 \mid r \in M]}{P[\Phi_J = 1 \mid r \in U]}\right).$$

13.3. Winkler String Comparator Metric for Typographical Error

Winkler [1990] introduced the following enhanced version of the Jaro string comparator metric:

$$\Phi_W(s_1, s_2) = \Phi_J(s_1, s_2) + i \cdot 0.1 \cdot (1 - \Phi_J(s_1, s_2))$$

where $i = \min(j, 4)$ and j, in turn, is the number of initial characters the two strings have in common on a character-by-character basis. This metric gives more importance to agreement on the initial characters of the strings than to agreement on the later characters of the string. This idea was inspired by the results of a large empirical study that Pollock and Zamora [1984] conducted to

develop a spell checker. (This was part of a Chemical Abstracts Service project funded by the US National Science Foundation.) Their study concluded that the most reliable characters of a string are those that occur first and that the data quality deteriorates monotonically as one moves from the beginning of the string to the end of the string. Winkler's enhancement increases the metric by a constant amount, $1 - \Phi_J(s_1, s_2)$, for each of the consecutive initial characters that match exactly between the two strings, up to a maximum of four characters.

Example 13.3: Using the Winkler string comparator metric on Shackleford versus Shackelford

Compute the Winkler metric under the conditions of Example 13.2.

Solution

Here, we have the first five letters of the two strings in character-by-character agreement. Hence, $i = 5$ and $j = \min(i, 4) = \min(5, 4) = 4$. So,

$$\Phi_W(s_1, s_2) = \Phi_J(s_1, s_2) + i \cdot 0.1 \cdot (1 - \Phi_J(s_1, s_2)) = \frac{31}{33}$$
$$+ 4 \cdot 0.1 \cdot \left(1 - \left(\frac{31}{33}\right)\right) = 0.9636.$$

13.4. Adjusting the Weights for the Winkler Comparator Metric

Instead of applying identical adjustment schemes to each of the diverse fields (e.g., last name, first name, and street number), Winkler and Thibaudeau [1991] proposed a scheme under which each of the fields would be assigned a distinct, but more appropriate, weight. This scheme involved the use of a piecewise linear function and required representative sets of pairs for which the truth of matches is known. The newly adjusted weights are specified as:

$$w_{na} = \begin{cases} w_a & \text{if } \Phi_W \geq b_1 \\ \max[\{w_a - (w_a - w_d) \cdot (1 - \Phi_W) \cdot a_1\}, w_d] & \text{if } b_2 \leq \Phi_W < b_1 \\ \max[\{w_a - (w_a - w_d) \cdot (1 - \Phi_W) \cdot a_2\}, w_d] & \text{if } \Phi_W < b_2 \end{cases}$$

The constants a_1, a_2, b_1, and b_2 depend on the specific type of field (e.g., last name) to which the weight adjustment is being applied. In most practical applications, we have $a_1 < a_2$. Some of the specific constants used are given in Table 13.1.

While this individual-adjustment scheme led to a slightly more accurate matching scheme with the high-quality files of the 1990 Census, it usually hurts matching with other files that are of lesser quality. For general matching applications, the Census Bureau applies the Jaro–Winkler scheme of Section 13.3 to the surname when it is used as a matching variable.

ypographical Error

of a Chemical Abstracts Service project
undation.) Their study concluded that
are those that occur first and that the
is one moves from the beginning of the
s enhancement increases the metric by
ch of the consecutive initial characters
gs, up to a maximum of four characters.

omparator metric on Shackleford versus

onditions of Example 13.2.

le two strings in character-by-character
$4) = \min(5, 4) = 4$. So,

$$0.1 \cdot (1 - \Phi_J(s_1, s_2)) = \frac{31}{33}$$

$$\left(\frac{31}{33}\right)\Big) = 0.9636.$$

for the Winkler

schemes to each of the diverse fields
mber), Winkler and Thibaudeau [1991]
the fields would be assigned a distinct,
le involved the use of a piecewise linear
of pairs for which the truth of matches
re specified as:

$$\begin{cases} \text{if } \Phi_W \geq b_1 \\ \cdot \Phi_W) \cdot a_1\}, w_d] & \text{if } b_2 \leq \Phi_W < b_1 \\ \cdot \Phi_W) \cdot a_2\}, w_d] & \text{if } \Phi_W < b_2 \end{cases}$$

on the specific type of field (e.g., last
being applied. In most practical appli-
specific constants used are given in

eme led to a slightly more accurate
files of the 1990 Census, it usually
of lesser quality. For general matching
le Jaro–Winkler scheme of Section 13.3
hing variable.

TABLE 13.1. Constants used in piecewise linear weight adjustments

Field type	Constant			
	a_1	a_2	b_1	b_2
Given name	1.5	3.0	.92	.75
Surname	3.0	4.5	.96	.75
House number	4.5	7.5	.98	.83

13.5. Where are We Now?

We have now completed Part II, and thereby completed all of our methodology chapters on editing, imputation, and record linkage. All of this is summarized in Chapter 20, our Summary chapter. In Chapters 14–17 we present case studies illustrating the application of these methodologies. In particular, Chapter 14 describes an application of both basic editing techniques and record linkage procedures to detect and repair errors in a large database.