Tony Zhang and Hannah Worrall
36-491 Data Matching Methods
Final Project Report

The Soundex Algorithm's Performance in Record Linkage and De-duplication

Abstract:

In the following report we will walk through and analyze our methods for applying the soundex algorithm for string comparison to the purpose of record linkage and record de-duplication. Our results suggest that the soundex method achieves performance on-par with the Jaro-Winkler similarity score when used in record linkage for English names.

Introduction:

The soundex algorithm generates a 4-digit code representing a 'distillation' of a given string, consisting of the first letter and three digits 0-6 representing all the remaining letters. In short, the first letter of the string is preserved and an iterative algorithm runs through the remaining characters deleting characters until three digits are left. The digits 0-6 correspond to 'buckets' of letters: groups of letters with similarities in sound.

Table 1: Soundex Character Mappings

Digit	1	2	3	4	5	6
Consonant	b,f,p,v	c,g,j,k,q,s,x,z	d,t	l	m,n	r

*0's are appended when the iteration scheme leaves fewer than 3 digits

Having been exposed to the theory and application of Jaro-Winkler string scores for the majority of class, we were curious as to why the soundex algorithm was not given much attention for homework and theory discussion. Although the method is reasonably simple we believed that the method would perform reasonably well with matching names, which was its intended purpose. Therefore, we decided to read further on how soundex works and design a project to compare its performance with the Jaro-Winkler metric.

Methods:

We used the methods presented in the homeworks to create an even testing field for the full-Jaro Winkler linkages and the soundex linkages. Since soundex does not consider number-number similarity we will use Jaro-Winkler scores for situations when we compare numbers to other numbers. Also, there is no notion of a 'score' or 'distance' for soundex; the 3 digit code merely represents the surviving sounds left after the iteration process and therefore differences in digits have little interpretability. Thus we are forced to use exact matching when comparing fields between records with soundex.

The project was split into two phases: record de-duplication and record pair linkage. For the record de-duplication phase we trained logistic models and classification trees on our data and used the predictions of each model to draw ROC curves and measure prediction accuracy. These measures are purely in-sample, since every one of our methods requires labeled data to initialize our models anyway.

For record pair linkage between two files of names, we used the Fellegi-Sunter method with and without assuming conditional independence between fields.

Data:

We selected small datasets mainly to work with soundex as a proof-of-concept. Therefore we were a little heavy-handed in our coding approach as we did not employ blocking as a size reduction technique. For record linkage we used 2 of the 100-record file pairs provided to us in the Fellegi-Sunter assignment, list1.txt and list2.txt, and the RLdata500 set from the RecordLinkage library for R. Each record in each dataset is a single row containing primary first name, secondary first name, primary last name, secondary last name, birth date, and a unique identifier for ‘true’ pairs of matches. Due to the lack of information for the secondary first and last names, they will not be used for any of our models and predictions.

We restructure the data into data frames with $\binom{n}{2}$ rows, each row representing a pair of records that were linked and their associated similarity scores/soundex agreements for each field.

Results*:*

I. De-duplication

We first ran a logistic model to predict whether or not the soundex matching for names would hold up against matching using purely Jaro-Winkler scores. The ROC curves, prediction performance table, and logistic regression output are provided below.

Jaro-Winkler Only:

```
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -76.569     26.220  -2.920 0.003498 **
fname_c1      32.375     20.087   1.612 0.107014
lname_c1      33.074     13.828   2.392 0.016766 *
by             7.616      2.501   3.046 0.002322 **
bm             4.192      1.636   2.563 0.010391 *
bd             5.728      1.651   3.469 0.000521 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 882.184  on 124749  degrees of freedom
Residual deviance:  23.216  on 124744  degrees of freedom
```

```
AIC: 35.216

Number of Fisher Scoring iterations: 19
```

Soundex + Jaro Winkler for birth dates:

```
            Estimate Std. Error z value Pr(>|z|)    
(Intercept)  -43.474      8.203  -5.300 1.16e-07 ***
fname_c1      10.941      2.583   4.235 2.29e-05 ***
lname_c1       9.515      2.357   4.037 5.42e-05 ***
by            15.068      3.681   4.093 4.25e-05 ***
bm             9.193      2.747   3.347 0.000818 ***
bd            10.316      2.732   3.776 0.000159 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 882.18  on 124749  degrees of freedom
Residual deviance:  42.94  on 124744  degrees of freedom
AIC: 54.94

Number of Fisher Scoring iterations: 17
```

We observe that the combination of soundex and Jaro-Winkler appears to make all of the variables statistically significant, while the Jaro-Winkler logistic regression output seems to indicate that first name scores are not a significant predictor for the odds of a match.

Jaro-Winkler Prediction Performance:

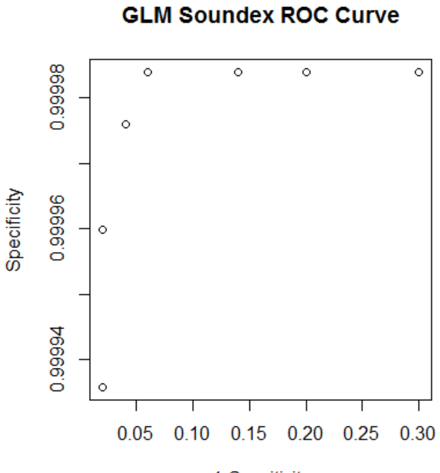
	Predicted Non Match	Predicted Match
Actual Non Match	124698	2
Actual Match	5	45

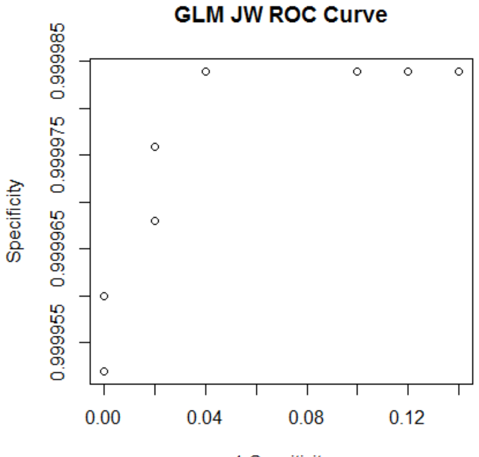
Soundex Prediction Performance:

	Predicted Non Match	Predicted Match
Actual Non Match	124698	2
Actual Match	7	43

We observe that the soundex predictions are slightly less accurate than the Jaro-Winkler predictions, but this difference is only 2 false negative predictions. Since we only change the name comparisons to soundex codes, it may also be possible that the stability in results is being driven by the birth date information staying constant.

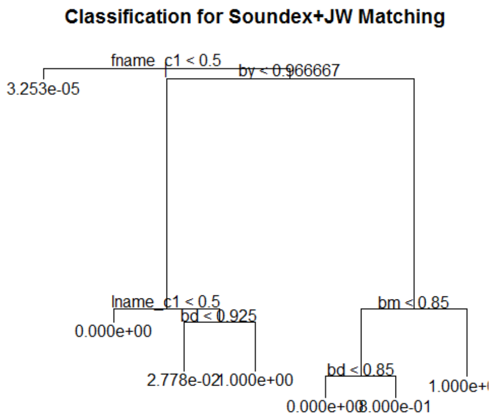
ROC Curves:

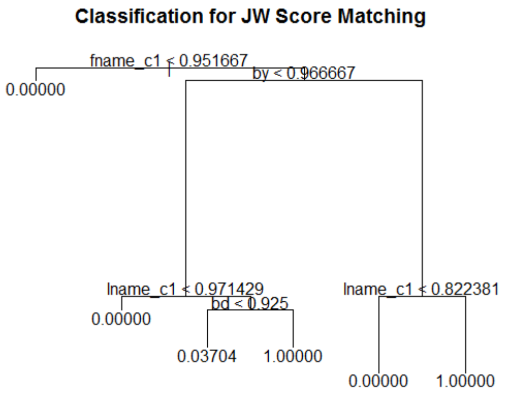




The ROC curve seems to suggest that both algorithms perform quite well for the dataset, as their specificity both dramatically increase for given levels of (inverse) sensitivity.

Classification Trees:


Classification for Soundex+JW Matching


Classification for JW Score Matching

We observe very similar shapes in the trees in terms of which first ‘cuts’ are the most important for making decisions: first name and birth year. Despite changing the way we assign first name matching, it is still identified as the most important variable to make the first branch of the tree.

Soundex Tree Performance:

	Predicted Non Match	Predicted Match
Actual Non Match	124700	0
Actual Match	9	41

Jaro-Winkler Tree Performance:

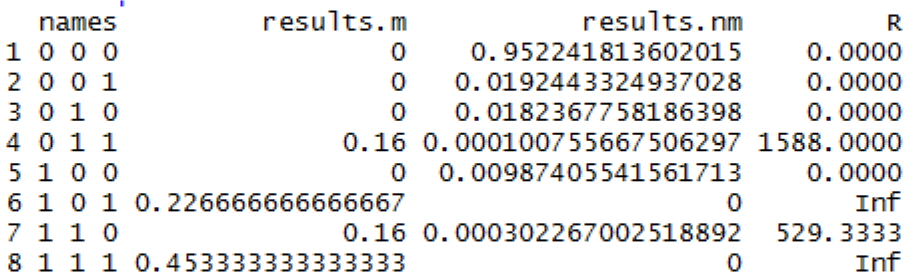
	Predicted Non Match	Predicted Match
Actual Non Match	124700	0
Actual Match	1	49

We observe that the tree for soundex performs comparably to its logistic regression counterpart, except that it makes no false-positive errors and seemingly shifts those to false-negatives. However, in this case the Jaro-Winkler tree outperformed it by a significant margin.

From these results we conclude that soundex looks promising as a tool for de-duplication, though is edged out be Jaro-Winkler in some cases.

II. Record Linkage

Soundex Fellegi-Sunter Probabilities:



```
  names          results.m          results.nm         R
1 0 0 0                  0    0.952241813602015    0.0000
2 0 0 1                  0   0.0192443324937028    0.0000
3 0 1 0                  0   0.0182367758186398    0.0000
4 0 1 1               0.16 0.000100755667506297 1588.0000
5 1 0 0                  0  0.00987405541561713    0.0000
6 1 0 1 0.226666666666667                    0       Inf
7 1 1 0               0.16 0.000302267002518892  529.3333
8 1 1 1 0.453333333333333                    0       Inf
```

*results.m = P(Observing Agreement Pattern | Match)
**results.nm = P(Observing Agreement Pattern | Non-match)

The Fellegi-Sunter method evaluates each paired row as a whole instead of a field-by-field comparison. In its simplest form, we want to see if a particular agreement pattern is more likely to be observed given that the true data is a match and vice versa. From the above table we observe the 8 possible agreement patterns for the first name, last name, and birth year of subjects in the table as well as their associated probabilities of occurrence in the match and non-match states. The high R coefficients indicate that the probability of observing a given agreement pattern when it is a match highly outweigh the probability of observing such a pattern if it wasn't a match, and therefore we would assign all such agreement patterns when evaluating the data as matches. Small R values are therefore assigned as nonmatches, and 'in betweens' are evaluated by clerks.

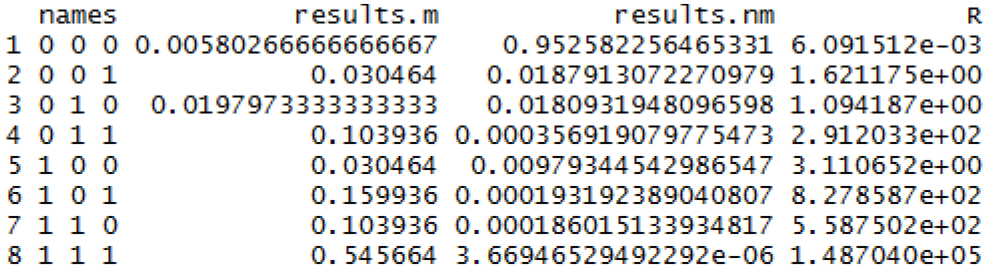
For this particular table we see that patterns 4, 6, 7, and 8 are likely to be matches if observed and the rest are likely to be nonmatches. In this case we do not send any to clerical review as we have no ambiguous R values.

Record Linkage Performance:

	Predicted Non Match	Predicted Match
Actual Non Match	9921	4
Actual Match	0	75

The soundex-derived agreement patterns identified all the actual matches, but made four false positive predictions. In contrast, the Jaro-Winkler scores applied to Fellegi-Sunter resulted in 7 false negatives and 0 false positives. Overall, this is a strong performance by the soundex matching methods.

We may also try to assume conditional independence between fields to calculate the Fellegi-Sunter probabilities for each agreement pattern. When making such an assumption, the results turn out as follows:



```
  names            results.m          results.nm            R
1 0 0 0 0.00580266666666667    0.952582256465331 6.091512e-03
2 0 0 1            0.030464   0.0187913072270979 1.621175e+00
3 0 1 0  0.0197973333333333   0.0180931948096598 1.094187e+00
4 0 1 1            0.103936 0.000356919079775473 2.912033e+02
5 1 0 0            0.030464  0.00979344542986547 3.110652e+00
6 1 0 1            0.159936 0.000193192389040807 8.278587e+02
7 1 1 0            0.103936 0.000186015133934817 5.587502e+02
8 1 1 1            0.545664 3.66946529492292e-06 1.487040e+05
```

According to the R values, we should automatically take fields 4, 6, 7, and 8 as matches, field 1 as a non match and fields 2, 3, and 5 to clerical review. The resulting predictions are:

	Predict Match	Predict Non Match	Clerical Review
Actual Match	75	0	0
Actual Non match	4	9451	470

We observe that the conditional independence assumption has not improved our prediction performance. The conditionally independent probabilities have possibly overweighted some of the less likely cases that we observed when we did not make the assumption, therefore introducing ambiguity into some of the agreement pattern cutoffs. The assumption has only created more work for the clerks, which is costly.

Overall, the soundex system has performed admirably in the two tasks we applied it to.

Conclusion*:*

Soundex does its job acceptably in both record deduplication and record linkage; we were able to coax out reasonable results from two medium sized datasets which in some cases outperformed those of the supposedly stronger Jaro-Winkler metrics. Despite being forced to exact match on some fields, we were not able to identify any systematic errors that may have caused the method to make an abnormal amount of false negatives or false positives. That said, it is difficult to recommend this algorithm given its flaws and variety of superior alternatives (Metaphone I/II/III etc.).

Unfortunately, the soundex algorithm is critically flawed in some very crucial aspects. It reduces *every* string into a 4-character code, so a lot of information from longer strings is lost through its iterative process. The iteration also creates some interesting false matches such as ‘Supercalifragilisticexpialidocious’ mapping to S162 while the much shorter ‘Superc’ maps to S162 as well. This is clearly absurd. As mentioned previously, there is no concept of distance between two string codes, which may be useful for mathematical justifications and verification of this method. If we were to make our own augmentations to the algorithm, we would look for ways to extend soundex beyond its current 4-digit limit and/or figure out a way to translate the string into a phonetic ‘score’ such that numerical differences or transformations would be interpretable. A version of soundex scoring supposedly exists in MySQL but our searches on its actual implementation came up empty.

Sources:

Sariyar, Murat. "The RecordLinkage Package." *R-project*. N.p., 1 Dec 2010. Web. 10 Oct 2013. <http://journal.r-project.org/archive/2010-2/RJournal_2010-2_Sariyar+Borg.pdf>

"Soundex." *Wikipedia*. Wikipedia, 15 Jul 2013. Web. 10 Oct 2013. <http://en.wikipedia.org/wiki/Sounde&xgt;.>