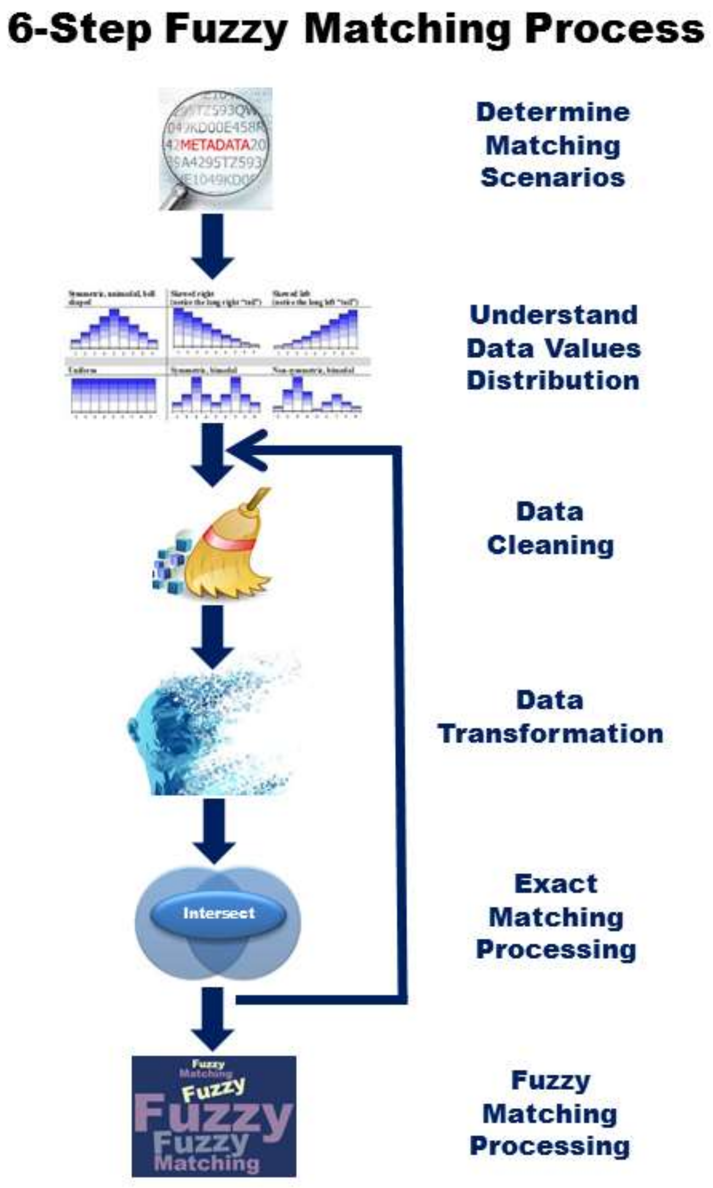
Kirk's Korner
of Quick & Simple SAS Tips
@sasNerd

Fuzzy Matching Programming Techniques Using SAS®

Data comes in all forms, sizes, layouts, and complexities. When data sources (data sets) contain consistent and valid data values, share common and unique identifier(s), and have little or no missing data, the matching process rarely presents any problems. But, when data sources do not contain unique or reliable identifiers, or “keys”, the matching process becomes more complicated and problematic, often producing unpredictable results. This tip suggests a 6-step fuzzy matching process, see figure, for users to consider, including the use of the SOUNDEX algorithm, and the SPEDIS, COMPLEV, and COMPGED functions to resolve key identifier issues and to successfully merge, join and match less than perfect or “messy” data.


6-Step Fuzzy Matching Process
Determine Matching Scenarios
Understand Data Values Distribution
Data Cleaning
Data Transformation
Exact Matching Processing
Fuzzy Matching Processing
Intersect

Step #1: Determine the Likely Matching Variables.

This first step determines whether what, if any, variables exist in the various data sources for matching purposes. Accessing the available metadata sources, including using the CONTENTS procedure, DATASETS procedure, Dictionary tables, and/or SASHELP views can be a huge asset to answer this question.

Step #2: Understand the Distribution of Data Values.

To derive a more accurate understanding of the data sources, we suggest that users conduct extensive data analysis by identifying missing values, outliers, invalid values, minimum and maximum values, averages, value ranges, duplicate observations, distribution of values, and the number of distinct values (By-groups) a categorical variable contains. One of the first things data analysts and data wranglers will want to do is explore the data using the FREQ procedure, or an equivalent approach like Excel Pivot Tables.

Step #3: Perform Data Cleaning.

Data cleaning, referred to as data scrubbing, is the process of identifying and fixing data quality issues including missing values, invalid character and numeric values, outlier values, value ranges, duplicate observations, and other anomalies found in data sets. SAS provides many powerful ways to perform data cleaning tasks.

Use SAS Functions to Modify Data

SAS functions are an essential component of the SAS Base software. Representing a variety of built-in and callable routines, functions serve as the "work horses" in the SAS software providing users with "ready-to-use" tools designed to ease the burden of writing and testing often lengthy and complex code for a variety of programming tasks. Examples of SAS functions include STRIP, CAT, CATQ, CATS, CATT, and CATX.

Explore Data Issues with the FORMAT Procedure

Problems with inaccurately entered data often necessitate time-consuming validation activities. A popular technique used by many to identify data issues is to use the FORMAT procedure.

Add Categories, if Available, to the Start of the Name

Doing this can eliminate matches that might occur if two businesses in the same general geographic area have the same name for example: Smith's could describe a hardware store, a restaurant, or another type of business.

Remove Special or Extraneous Characters

Punctuation can differ even when names or titles are the same. Therefore, we suggest removing the following characters: ' " & ? – from names, titles, and other fields.

Put All Characters in Upper-case Notation and Remove Leading Blanks

Data bases could have different standards for capitalization, and some character strings can be inserted with leading blanks. Data sources can store values as all lower-case, upper-case, or in mixed-case which can impact the success of merge and join matching techniques. Consequently, the STRIP and UPCASE functions can be used.

Remove Words that might or might not Appear in Key Fields

Commonly used words in language, referred to as stop words, are frequently ignored by many search and retrieval processes. Stop words are classified as irrelevant and, as a result, are inserted into stop lists and are ignored. Examples include The, .com, Inc, LTD, LLC, DIVISION, CORP, CORPORATION, CO., and COMPANY.

Choose a Standard for Addresses

Address fields can present a challenge when analyzing and processing data sources. To help alleviate comparison issues, decide whether to use Avenue or Ave, Road or Rd, Street or St, etc, and then convert the address fields accordingly or create a user-defined lookup process using PROC FORMAT to match the standard values.

Rationalize Zip Codes when Matching Addresses, Use Geocodes when Available

We found it useful to remove the last 4 digits of 9-digit zip codes, because some data sources might contain 5-digit zip codes. Since some data sources might contain zip codes as numeric fields, and others might contain zip codes as character fields, make sure to include leading zeroes. If working with US zip codes, make sure they are all numeric. This may not apply for other countries. One common mistake to watch for is that sometimes Canada, with abbreviation CA, is entered as the state CA (California) instead of the country CA.

Specify the DUPOUT=, NODUPRECS, or NODUPKEYS Options

A popular and frequently used procedure, PROC SORT, identifies and removes duplicate observations from a data set. By specifying one or more of the SORT procedure's three options: **DUPOUT=**, **NODUPRECS**, and **NODUPKEYS**, users are able to control how duplicate observations are identified and removed.

Step #4: Perform Data Transformations.

Data transformations may be necessary to compare data sources. Data set structures sometimes need to be converted from wide to long or long to wide and their contents may need to be reconciled by having their variables grouped in different ways. When a data set's structure and data is transformed, we typically recommend that a new data set be created from the original one. The TRANSPOSE procedure is handy for restructuring data in a data set, and is used in preparation for special types of processing like arrays. Data can be transformed with or without grouping.

Step 5: Process Exact Matches.

With our data sources cleaned and, if necessary, transformed, we then turn our focus to processing exact matches using reliable identifiers or "key" variables. Traditional DATA step match-merge or PROC SQL join techniques are typically used to satisfy this step.

Step 6: Match Key Fields using Fuzzy Matching Techniques.

For any unmatched observations from step 5, a number of fuzzy matching techniques are available for use. The Base SAS software fuzzy matching techniques include the Soundex (phonetic matching) algorithm, and the SPEDIS, COMPLEV, and COMPGED functions to help search and match problematic data sources.

The Soundex Algorithm

The Soundex (phonetic matching) algorithm involves matching data sources on words that sound alike. Soundex was invented and patented by Margaret K. Odell and Robert C. Russell in 1918 and 1922 to help match surnames that sound alike. It is limited to finding phonetic matches and adheres to the following rules when performing a search:

- Ignores case (case insensitive);
- Ignores embedded blanks and punctuations;
- Is better at finding English-sounding names.

Although the Soundex algorithm does a fairly good job with English-sounding names, it frequently falls short when dealing with the multitude of data sources found in today's world economy where English- and non-English sounding names are commonplace. It also has been known to miss similar-sounding surnames like Rogers and Rodgers while matching dissimilar surnames such as Smith, Snthe and Schmitt.

The SPEDIS Function

The SPEDIS, or Spelling Distance, function and its two arguments evaluate possible matching scenarios by translating a keyword into a query containing the smallest distance value. The SPEDIS function evaluates query and keyword arguments returning non-negative spelling distance values. A derived value of zero indicates an exact match. Generally, derived values are less than 100, but, on occasion, can exceed 200. The authors recommend using the SPEDIS function to control the matching process by specifying spelling distance values greater than zero and in increments of 10 (e.g., 10, 20, etc.).

The COMPLEV Function

The COMPLEV, or Levenshtein Edit Distance, function is another fuzzy matching SAS technique. COMPLEV counts the minimum number of single-character insert, delete, or replace operations needed to determine how close two strings are. Unlike the SPEDIS function and COMPGED function (discussed later), the COMPLEV function assigns a score for each operation and returns a value indicating the number of operations.

The COMPGED Function

The COMPGED function is another fuzzy matching technique which is facilitated by a SAS function. It works by computing and using a Generalized Edit Distance (GED) score when comparing two text strings. The GED score is a generalization of the Levenshtein edit distance, which is a measure of dissimilarity between two strings. When using the COMPGED function to match data sources with unreliable identifiers (or keys), the higher the GED score the less likely the two strings match. Conversely, for the greatest likelihood of a match with the COMPGED function users should seek the lowest derived score from evaluating all the possible ways of matching string-1 with string-2.

For the longer version of this paper, access the 2018 SAS Global Forum paper by Sloan and Lafler (SGF 2018).

About the Authors

Kirk Paul Lafler is an entrepreneur and founder at Software Intelligence Corporation, and has been using SAS since 1979. Kirk is a SAS consultant, application developer, programmer, certified professional, mentor, provider of SAS consulting and training services, advisor and adjunct professor at University of California San Diego Extension, emeritus sasCommunity.org Advisory Board member, and educator to SAS users around the world. As the author of six books including Google® Search Complete (Odyssey Press. 2014) and PROC SQL: Beyond the Basics Using SAS, Second Edition (SAS Press. 2013); Kirk has written hundreds of papers and articles; served as an Invited speaker, trainer, keynote and section leader at SAS user group conferences and meetings around the world; and is the recipient of 25 "Best" contributed paper, hands-on workshop (HOW), and poster awards.

Stephen Sloan has worked at Accenture in the Services, Consulting, and Digital groups and is currently a senior manager in the SAS Analytics area. He has worked in a variety of functional areas including Project Management, Data Management, and Statistical Analysis. Stephen has had the good fortune to have worked with many talented people at SAS Institute. Stephen has a B.A. in Mathematics from Brandeis University, M.S. degrees in Mathematics and Computer Science from Northern Illinois University, and an MBA from Stern Business School at New York University.

Comments and suggestions may be sent to:

Kirk Paul Lafler
SAS® Consultant, Application Developer, Programmer, Data Analyst, Educator and Author
Software Intelligence Corporation
E-mail: KirkLafler@cs.com
LinkedIn: http://www.linkedin.com/in/KirkPaulLafler
Twitter: @sasNerd

~ ~ ~ ~ ~ ~ ~

Stephen Sloan
Senior Manager in SAS Analytics
Accenture
E-mail: Stephen.B.Sloan@accenture.com

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration. Other brand and product names are trademarks of their respective companies.