Patient Matching Deduplication

Xinning Chu, Dandan Feng, Kang Fu, Ashley Hall

PATIENT MATCHING PROVIDES THE ABILITY TO MATCH A UNIQUE INDIVIDUAL WITH A UNIQUE SET OF DATA IN A HEALTHCARE DATABASE OR DATA SET.

What is Patient Matching?

Why patient matching?

Accuracy of patient records can be significantly impacted by duplicate records. Record inaccuracy can negatively impact patient safety, speed of care delivery, and cost (spend):

1. **Safety**: An estimated 195,000 deaths occur each year due to medical errors, with 10 out of 17 being the result of identity errors.
2. **Spend**: Reported averages of $1,950 per inpatient and over $800 per ED visit for repeated medical care.
3. **Delivery**: Inaccuracy of patient history due to duplicate records result in repeated tests and delay in ER care and surgery.

Data Source

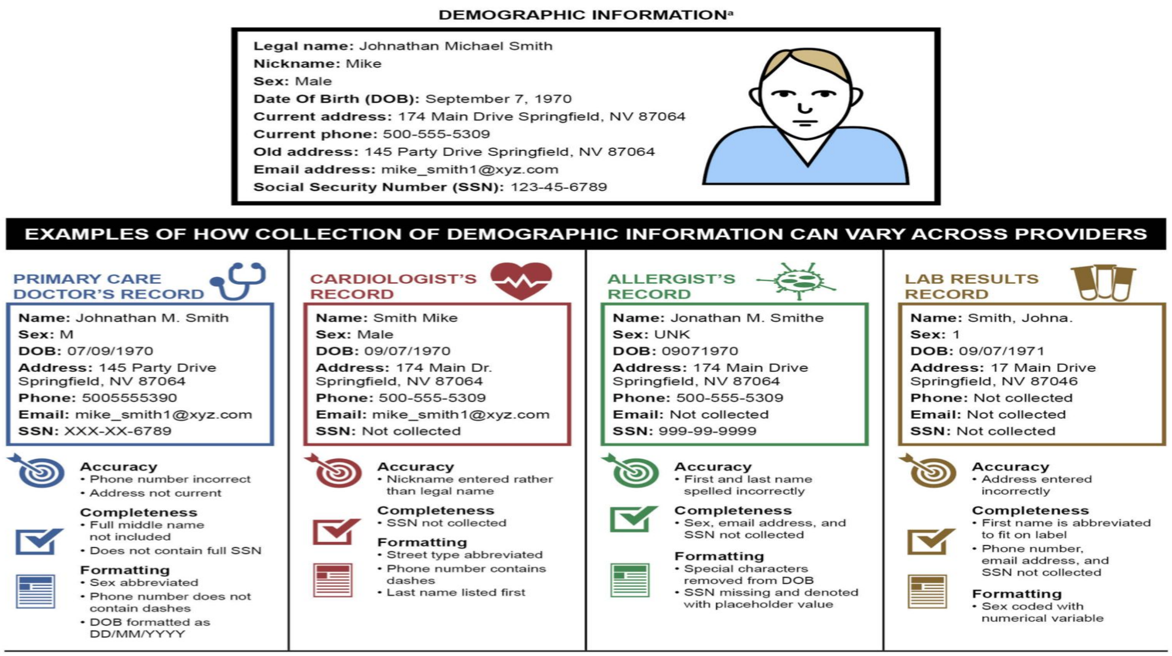
Provided by ONC (19 variables in total):

Field Name	Data Type	Description	Sample Element
Enterprise ID	Numeric	Unique patient identifier - enterprise level	12169795
LAST NAME	Text	Patient's last name	Washington
FIRST NAME	Text	Patient's first name	Jennifer
MIDDLE NAME	Text	Patient's middle name	Rennie
SUFFIX	Text	A group of letters placed after name to provide some additional information	JR, SR
DOB	Text	Patient's date of birth	7/4/1971
GENDER	Text	Patient's gender	FEMALE, M, F, Unknown
SSN	Text	Patient's social security number	999-99-9999
ADDRESS1	Text	Patient's address (typically street address or P.O. Box, etc.)	4732
ADDRESS2	Text	More specific information regarding address 1 (i.e. apartment, suite, department, room, etc.)	Unit 3
CITY	Text	Patient's city (address)	Washington
STATE	Text	Patient's state (address)	PA
ZIP	Numeric	Patient's zip code (address)	20019
PHONE	Text	Patient's phone number - primary	703-100-1234
PHONE2	Text	Patient's phone number - secondary	202-200-1234
EMAIL	Text	Patient's email address	Jen.Washington@amggt.com
ALIAS	Text	Patient's alias name - Any previous name associated with a record….can have first, last, and middle names….could be a legal name, nickname, previous married name, maiden name, IP/Alias, etc Could b another name entered in error and corrected	Jenn
MOTHERS_MAIDEN_NAME	Text	Patient's mother's maiden name	Jones
MRN	Numeric	Medical Record Number - Unique patient identifier - site level	9384895

Data Problems

The real world data is “dirty”:

Figure 1: Examples of Data Quality Issues That Can Affect Patient Record Matching



Source: GAO analysis. | GAO-19-197

Data Cleansing

Potential ideas about cleaning the dataset:

- Enterprise ID
- Last Name – Remove Special Characters (ie – ‘), all caps
- First Name – Remove Special Characters (ie – ‘), all caps
- Middle Name – Remove Special Characters (ie – ‘), all caps
- Suffix – make sure only values in the dataset currently are SR, II, JR., SR., and JR. So just remove periods
- DOB -- Could possibly be helpful to break this out into MONTH, DAY & YEAR variables. That way if there is, for example, mistake with the day, the month could maybe still be used in helping with a match.
- Gender -- FEMALE > F & MALE > M. Make U (unknown values blank?). Do we want to distinguish between genders listed as unknown explicitly and those simply left blank?
- SSN -- Remove "- “, ''Fake' to 'Null'
- Address 1 -- http://www.gis.co.clay.mn.us/usps.htm has the standard abbreviation for all types of streets according to USPS. We could use this as the reference database for converting to the standardized names, all caps, Do we maybe want to separate address elements similar to how we would separate date elements? Could help control for errors if someone for example entered that an address was a street when it really is an avenure, so there could still be a match highlighted by the number and street name but not the street type (AVE, ST, ETC) part?
- Address 2 – all caps, same rules as address 1
- City -- Use zip city, state, & zip code database from USPS to verify different spellings of cities and combinations of zip codes, states, & cities.
- State -- UN/non-state abbreviates nulled, Use the USPS database to correct any incorrection state abbreivations.
- Zip -- Add leading zeros to zip codes that are under 5 digits long, Remove any -'s and make all zip codes just 5 digits not 9,
- Phone 1 -- Make sure all numbers are in 999-999-9999 format,
- Phone 2 – same as phone 1
- Email -- Create rules for identifying emails that are clearly incorrect (i.e. they do not include @, do not end in .com .net .mail, etc), all caps
- Alias – all caps, Thinking we could split the string and make each grouping its own field (i.e. "Lisa Ferguson Potter" into "Lisa" and "Ferguson" and Potter"). From here we can compare to the data entered in First/Middle/Last/Maiden name to either check first name entered or widen the search of that person's last name to maiden name, Some alias's have "^^". We'll have to remove these
- Mothers Maiden Name -- Remove special characters (i.e. - '), all caps,
- MRN

Previously

Previously Approaches

- Deterministic matching :

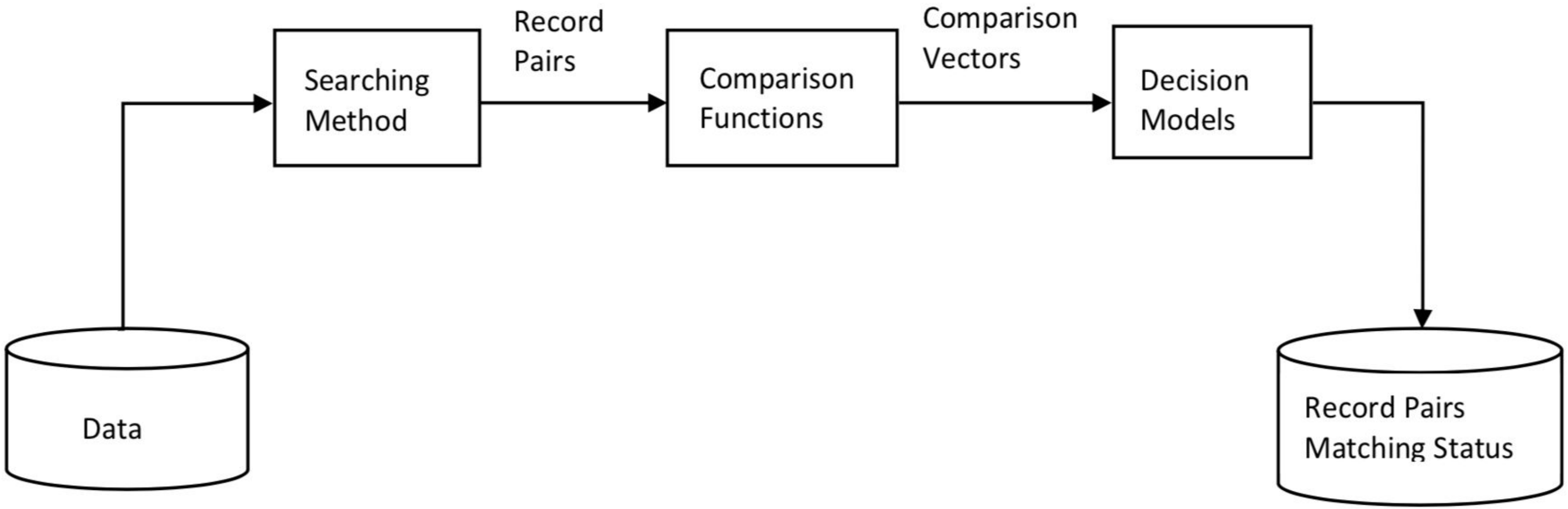
Comparing unique identifiers for each record to determine if two records are duplicates. This method tends to have high precision, low recall, which makes it a strong starting point to become familiar with a data set.

- Probabilistic matching:

In probabilistic matching, several field values are compared between two records and each field is assigned a weight that indicates how closely the two field values match. The sum of the individual fields weights indicates the likelihood of a match between two records.

Machine Learning Approach

Workflow


Searching
Method
Record
Pairs
Comparison
Functions
Comparison
Vectors
Decision
Models
Data
Record Pairs
Matching Status

Searching Method

- Blocking
- Sorting Neighborhood

Blocking

1. Reduce computation time and memory consumption.
2. Block key. Partition a file into mutually exclusive blocks.
3. Only records in the same block are considered for comparison.
4. The number of generated record pairs depends on the number of blocks.

Comparison Functions

Definition of Comparison Vectors

For each record pair $r_{i,j} = (r_i, r_j)$

1. Comparison Function
2. Comparison Vectors

$$c_k^{i,j} = C_k(r_i.f_k, r_j.f_k) \quad f_1, f_2, \ldots, f_n$$

$$C_I(value_1, value_2) = \begin{cases} 0 & \text{if } value_1 = value_2 \\ 1 & \text{otherwise} \end{cases}$$

Overview of the cleaned data set

Numeric:

SSN

DOB

Phone Number

MRN (Medical Record Number- Unique patient Identifier, site level)

String:

Name (Last, First, Middle)

Address (Street, City, State)

Comparison Functions

- Hamming Distance
- Edit Distance
- Jaro’s Algorithm
- N-grams
- Soundex Code

Jaro-Winkler Algorithm

1. Jaro-Winkler Distance is a string comparison function, primarily for short strings, e.g., First/Last Names
2. Jaro’s algorithm is used to compute the distance between two strings and is based on matching and transposition of letters. e.g.,“John & Jhon” vs “John & Jon”.
3. Winkler modification.
4. The larger the Jaro-Winkler distance is, the more similar the strings are.

Decision Models

EM-based Probabilistic Model
Hybrid Model

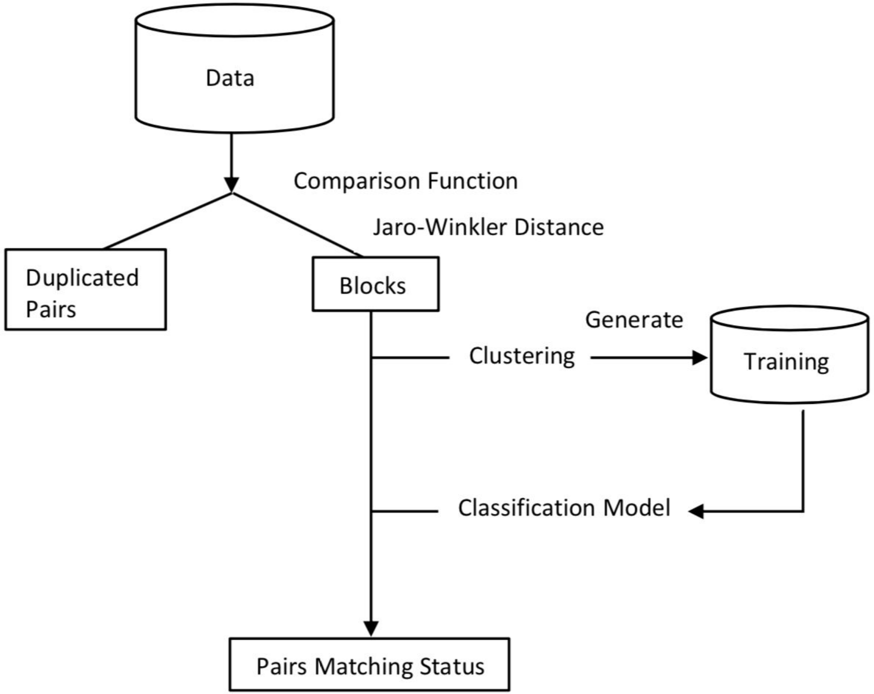
Hybrid Model

Two Steps:

1.**Clustering** is applied to predict the matching status of a small set of record pairs (blocks). **Form training and testing data set**. {c, f(c)}

2.Using training set to build a classifier, then we employ the classifier to whole set to build a **classification model**.

Hybrid Model Information Flow



R Code, Workflow, and Results - Overview

ORIGINAL DATA

GENERATING/ CLUSTERING TO A SAMPLE = TRAINING SET

GENERATE SVM MODEL, APPLY TO TRAINING SET

ORIGINAL DATA

- Original Fields = Enterprise ID, Identity, DOB, Zip Code, Last Name, First Name, Gender, SSN, State, City, Address, and Concatenated Address
- Drop Fields = Identity, State, City, Address
- Addressing Memory in R = delete original data file, set higher memory
- Create Function completeFun() = takes in data and desired column that we want to block on (in this case SSN), and deletes all entries that have N/A's in SSN's field

GENERATING/ CLUSTERING TO A SAMPLE = TRAINING SET

1. Use completeFun() to remove all N/A's from SSN
2. Create record pairs on SSN from all of the data using compare.dedup()
3. Run ClassifyUnSup() on paired data
   a. Count number of record pairs
   b. Define expected proportion of links
   c. ClassifyUnSup(paired data, Unsup_method (unsupervised method = kmeans or bclust), iter.max)
4. Take a sample: decide how many samples you want, and the defined proportion of links sends some to training set and rest to testing set

GENERATE SVM MODEL, APPLY TO TRAINING SET

1. Train model using trainSupv()
    a. Function imports training set and Sup_method (svm)

```
model <<- trainSupv(train, Sup_method, use.pred = TRUE, include.data =
FALSE)
```

Results

- Use model on testing set, and create vector called prediction = will identify the pair as either L or N, subset to prediction = L and two enterprise ID's are the matching results.

	newSSN.2	concatADDRESS.2	Weight	prediction
997370	345678901	1555ODELLSTREET	NA	N
997507	109876543	525WEST158THSTREET	NA	N
997655	777777777	446MILFORDSTREET	NA	N
997803	109876543	525WEST158THSTREET	NA	N
998236	870378003	862517AVENUE	NA	L
998269	111111111	150WESTBURNSIDEAVENUE	NA	N

	enterpriseID.1	enterpriseID.2
1	15552908	15931101
3	15803265	16008453
5	15880687	15927363
7	15869291	15975851
9	15858256	16003840
11	15565740	15752641

Unsupervised Machine Learning Algorithms

A type of ML algorithms used to draw inferences from datasets consisting of input data without labeled responses.

Most common: clustering analysis, used to find hidden patterns or grouping in the data

K-means Clustering

① stores k centroids that it uses to define clusters. A point is considered to be in a particular cluster if it is closer to that cluster' s centroid than any other centroid.

② finds the best centroids by alternating between (1) assigning data points to clusters based on the current centroids (2) chosing centroids (points which are the center of a cluster) based on the current assignment of data points to clusters.

The Algorithm

In the clustering problem, we are given a training set $x^{(1)}, \ldots, x^{(m)}$, and want to group the data into a few cohesive "clusters." Here, we are given feature vectors for each data point $x^{(i)} \in \mathbb{R}^n$ as usual; but no labels $y^{(i)}$ (making this an unsupervised learning problem). Our goal is to predict k centroids **and** a label $c^{(i)}$ for each datapoint. The k-means clustering algorithm is as follows:

1. Initialize **cluster centroids** $\mu_1, \mu_2, \ldots, \mu_k \in \mathbb{R}^n$ randomly.
2. Repeat until convergence: {

    For every i, set

    $$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2.$$

    For each j, set

    $$\mu_j := \frac{\sum_{i=1}^m 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^m 1\{c^{(i)} = j\}}.$$

    }

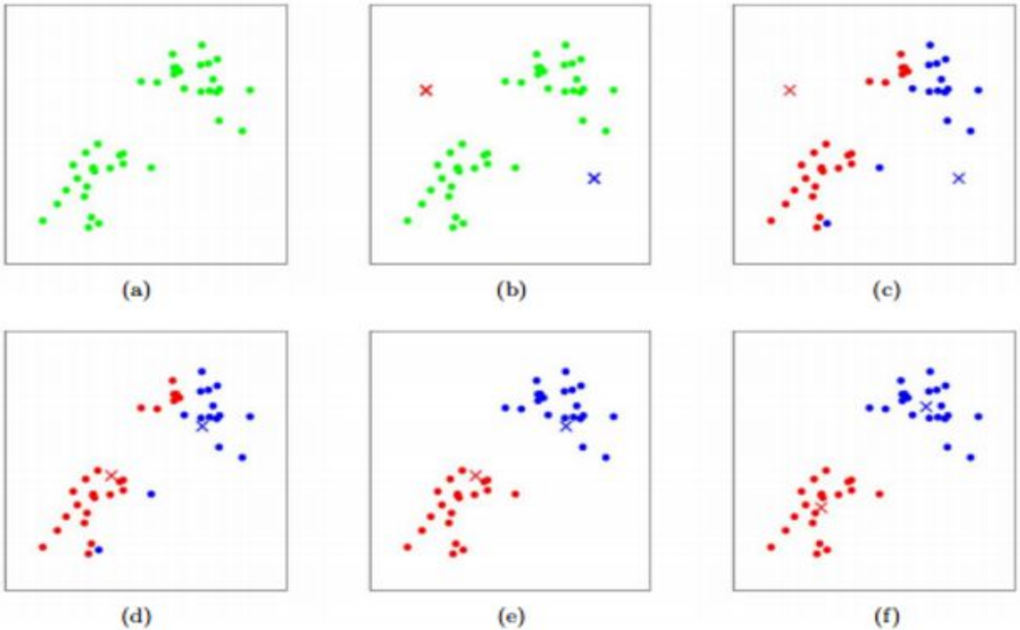


Figure 1: K-means algorithm. Training examples are shown as dots, and cluster centroids are shown as crosses. (a) Original dataset. (b) Random initial cluster centroids. (c-f) Illustration of running two iterations of k-means. In each iteration, we assign each training example to the closest cluster centroid (shown by "painting" the training examples the same color as the cluster centroid to which is assigned); then we move each cluster centroid to the mean of the points assigned to it. Images courtesy of Michael Jordan.

Reference

Elfeky, M.G., Verykios, V.S., & Elmagarmid, A.K. (2003). Record Linkage: A Machine Learning Approach, A Toolbox, and a Digital Government Web Service.