2017-12-01

Recommender Systems for Family History Source Discovery

Derrick James Brinton
Brigham Young University

Recommender Systems for Family History Source Discovery

Derrick James Brinton

A thesis submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Master of Science

Christophe Giraud-Carrier, Chair
Daniel Zappala
Eric G. Mercer

Department of Computer Science

Brigham Young University

ABSTRACT

Recommender Systems for Family History Source Discovery

Derrick James Brinton
Department of Computer Science, BYU
Master of Science

As interest in family history research increases, greater numbers of amateurs are participating in genealogy. However, finding sources that provide useful information on individuals in genealogical research is often an overwhelming task, even for experts. Many tools assist genealogists in their work, including many computer-based systems. Prior to this work, recommender systems had not yet been applied to genealogy, though their ability to navigate patterns in large amounts of data holds great promise for the genealogical domain.

We create the Family History Source Recommender System to mimic human behavior in locating sources of genealogical information. The recommender system is seeded with existing source data from the FamilySearch database. The typical recommender systems algorithms are not designed for family history work, so we adjust them to fit the problem. In particular, recommendations are created for deceased individuals, with multiple users being able to consume the same recommendations. Additionally, our similarity computation takes into account as much information about individuals as possible in order to create connections that would otherwise not exist. We use offline n-fold cross-validation to validate the results. The system provides results with high accuracy.

Keywords: Recommender Systems, Genealogy

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 The Domain

Family history is increasing in popularity, but the number of people who actively participate in family history research is orders of magnitude lower than the number of people who claim interest. Market research shows that interest in genealogy in the United States is high and has been following an upward trend.

> Maritz reports a rise in genealogical interest from 45% in 1996 to 60% in 2000 (genealogy.com, 2000), and since that date, the figure has been steadily rising to 75% in 2003 (Weiss, Nolan, Hunsinger, & Trifonas, 2006, p. 939).

People tend to be interested in their origins, and genealogy is one way to answer that question. Additionally, genealogy is an important aspect of the medical field, as many diseases and conditions arise from genetics. With the vast number of digitized records available on the Internet, along with other technological advances such as DNA analysis, it is more possible than ever before to trace heritage. However, despite the increasing availability and interest in genealogy, actual participation in family history research remains relatively low.

As an example of this, Ancestry.com, one of the largest family history technology companies in the industry, reports having 2.2 million users worldwide (Ancestry.com, 2015). Even if all of those 2.2 million users were just in the United States, that would mean Ancestry.com only serves 0.9% of the 75% who claim interest in family history (based on a US population of 318.9 million people (U.S. Census Bureau, 2014)). If we can extrapolate the 75% interest to the world population, then 2.2 million users is 0.04% of the potential 5.25 billion people who have interest. This discrepancy between the number of people who have interest in researching their family history, and those who actually do it, suggests significant room for improvement.

1.2 Family History Work is Hard[1]

One explanation for this disparity is that family history research is often a daunting, if not insurmountable venture. Finding credible sources that give information on ancestors is difficult and sometimes impossible.

> I know everyone has to start somewhere, but trying to participate in genealogy on the Internet before you have familiarised yourself with the basic methodology of...research and sources is like trying to drive a Formula 1 racing car before you've learned to ride a bike (Weiss et al., 2006, p. 940).

With such a high barrier to entry, coupled with an increasing interest in family history work, it should not be surprising that many try to cut corners.

> The many genealogies published on the Internet have given rise to the "quickie genealogist"—those who go online to pursue their ancestry, and by using the work of others, copy the information verbatim, disregarding basic genealogical methodology, to regurgitate the material, mistakes and all, as their own. This quick entry into genealogy results in new hobbyists not being socialised into the basic and specific values, skills, and methods of genealogy, such as citing references, and confirming sources to primary records (Veale, 2005, p. 10).

With so much inconsistency, requiring such specialized knowledge to sort out, how can family history work be made more accessible to the inexperienced amateurs as well as the much

[1] See Appendix section "Lies in Genealogy" for more information.

larger group of people who wish to trace their families' roots, but do not do so?

1.3 The Role Computing Plays

One potential answer comes with technological advances. Family history and computing found their intersection in 1979, when Personal Computing Magazine featured family history on the cover of their September issue. The issue included BASIC code for the first published computer program for genealogy. Since then, computers have increased their presence dramatically in the family history world. Most notably, the advent and spread of the Internet accelerated the adoption of computing as an essential aspect of family history work.

> The Internet explosion of the late 1990s has had a major effect on genealogy. Transcribed documents of genealogical interest have popped up everywhere on the Internet. More recently, scanned images of original documents also began to appear in significant numbers. Databases of birth, marriage, and death information proliferated. Mailing lists and personal e-mail messages soon supplemented and even replaced the written correspondence so common among earlier genealogists. The Internet has popularized family tree searches (Eastman, 2002).

This proliferation of family history work on the Internet has grown to the point where it is nearly ubiquitous in the genealogical realm. Perhaps even more impressive, 29% of all Internet users report that either they or a family member have used the Internet for family history research (The Pew Internet & American Life Project, 2000).

1.4 Recommender Systems

Recommender systems is a field of computing that has seen tremendous growth in the past two decades. Recommender systems are a class of algorithm that attempts to automatically determine user preference. They have typically been employed to assist consumers in finding new products. One well-known example of this is Amazon.com's recommender system, which gives recommendations for related products. Ricci, Rokach, Shapira, and Kantor give us a succinct description of recommender systems in their *Recommender Systems Handbook*.

> Upon a user's request, which can be articulated, depending on the recommendation approach, by the user's context and need, [recommender systems] generate recommendations using various types of knowledge and data about users, the available items, and previous transactions stored in customized databases (Ricci, Rokach, Shapira, & Kantor, 2015, p. 3).

Recommender systems are a subfield of both the machine learning and data mining fields. The more data they have available to them, the better able the algorithms are to give accurate recommendations.

1.5 Applying Recommender Systems to Family History

As far as the literature is concerned, recommender systems do not seem to have ever been applied to family history[2]. However, it is a natural step to apply them in the family history research process. While there are many potential applications for recommender systems in the family history realm, we will focus on applying recommender systems to helping users find sources for their ancestors. We choose source finding because it is an especially difficult task in genealogy work, and one for which recommender systems are especially suited.

Since knowing the accuracy of a piece of information in family history research is already extremely difficult, sources become essential for validation. Without a source, a piece of information carries almost no credibility. Furthermore, not all sources are created equal; some are more credible than others. Thus, finding the most credible sources of information on individuals is an essential task in family history research.

[2] It is possible that some proprietary software on the market today may use recommender systems. For example, Ancestry, FamilySearch, and FindMyPast all have "hints" which could be seen as recommendations. But their algorithms are proprietary and their approaches are unknown. See appendix section "Potential Benefits to Family History" for more information.

Recommender systems, like most data mining algorithms, require training on large amounts of data in order to be useful. The challenge of obtaining enough data for the recommender system to generate recommendations is often referred to as the “cold start” problem. Many efforts have been made over the years to digitize potential sources of genealogical information. Today, billions of indexed and searchable records exist in many different databases. These digitized records are important to utilize in the creation of a recommender system for family history sources.

Building a recommender system for genealogical sources presents some unique challenges whose solutions provide new contributions to the worlds of recommender systems and family history.

Chapter 2

Related Work

As previously mentioned, there is no readily discoverable evidence in the literature of work done on recommender systems in the family history research realm. As such, we focus instead on the literature that exists in each field separately.

While there has been work done in the intersection of family history and computing, that work tends to focus on other aspects of family history, such as handwriting recognition, deduplication, GIS, or DNA.[1]

2.1 Family History Research

The academic world of family history research itself (ignoring the technological aspect) is lacking, to say the least. As evidence of this, there have been several recently-published articles lamenting the lack of an academic discipline surrounding family history research. These articles propose what would need to change in order to establish family history as a credible research field (Hershkovitz, 2011; Jones, 2007; Mills, 2003a). Furthermore, the peer-reviewed publications that do exist are mostly example research stories. Such publications typically suggest that the patterns they follow might apply to other research as well, but they do not validate that assertion (e.g., (Sheppard, 1977)).

There is virtually no empirical data on best practice in family history research. Furthermore, when it is defined, best practice in family history research is mostly specific to

[1] This intersection is represented academically by The Family History Technology Workshop, hosted annually at Brigham Young University. Aside from this workshop, there are no other venues dedicated to publication in the realm of technology applied to family history.

region. There are, however, a small number of documents that attempt to outline the general genealogical research process. These are accepted as the standards for how to perform family history research. The Board for Certification of Genealogists (BCG) provides one example of such an accepted standard. It spells out rules to govern the research practices of professional genealogists (BCG, 2000). Perhaps the most well-known and accepted of these standards is referred to as the Genealogical Proof Standard. This states that in order for genealogical research to merit confidence, it must exhibit reasonably exhaustive research, sources with accurate citations, analytical tests of those sources, resolution of conflicts, and coherent conclusions. Aside from these standards, there are some select, applicable works on genealogy.

Duff and Johnson examine the ways in which genealogists use archives (Duff & Johnson, 2003). They discuss methods used by professional genealogists to find reliable sources of information about individuals. The results confirm that finding such sources today requires a certain amount of expertise that novices typically lack.

Grabowski finds that genealogists are the fastest growing group of researchers that use archives (Grabowski, 1992). Both Grabowski and Duff and Johnson give the impression that archivists view genealogists as lesser researchers, though they both make arguments to counter that sentiment. But the mere existence of such a prejudice confirms that genealogists as a whole could use some assistance in their research process.

Fulton finds that amateur genealogists can gain benefit in their information seeking by participating in sharing activities with other genealogists (Fulton, 2009). Such networks are typically ad hoc and unstructured in their organization. Collaborative recommender systems have the potential to encapsulate the benefit of such collaboration with others in a single, structured system.

Mills discusses the implications that the “Information Age” has on genealogy, though she focuses more on the history of genealogy than she does on the impact of computers on family history work. Nevertheless, she also mentions the importance of good research for genealogists in the age of the Internet.

> Digitalization and the Internet offer truly infinite opportunities for the dissemination of information. However, *information* is not synonymous with *knowledge*. Our challenge is to ensure that those who harvest that information (whether in the name of genealogy or history) process it in a way that preserves its integrity, that they interpret it knowledgeably, and then reassemble the evidence analytically and innovatively (Mills, 2003a, p. 277).

The challenge to process the massive amounts of genealogical data in an innovative way that furthers knowledge resonates with the goals of this thesis.

2.2 Recommender Systems

In contrast to the field of family history, there is an overabundance of published research in the field of recommender systems. Recommender systems research tends to focus on areas like security, serendipity, applications to sales and marketing, accuracy, optimization, etc.

As explained by Adomavicius and Tuzhilin, the literature breaks recommender systems down into three primary categories: content-based, collaborative, and hybrid systems (Adomavicius & Tuzhilin, 2005). Content-based recommender systems attempt to generate recommendations based on algorithmic parsing of the content of whatever is being recommended. Collaborative recommender systems, on the other hand, do not need to know content. Instead, they evaluate usage patterns and provide recommendations based on usage similarities between users. Hybrid approaches use some combination of the different kinds of recommendation system.

Knowledge-based recommender systems are a smaller class of algorithms that neither fall under content-based nor collaborative systems. They attempt to provide recommendations based on the user's search context. They require the user to explicitly communicate that context to the system.

Almazro, Shahatah, Albdulkarim, Kherees, Martinez, and Nzoukou give an excellent overview of recommender systems, specifically focusing on collaborative filtering and its associated algorithms (Almazro et al., 2010).

Perhaps the most widespread textbook on recommender systems is Ricci, Rokach, Sapira, and Kantor's *Recommender Systems Handbook* (Ricci et al., 2015). Of particular interest are the discussions on data preprocessing, collaborative filtering methods, and evaluation of recommender systems. They discuss numerous algorithms used in data preprocessing. Specifically, the algorithms on similarity metrics are useful to our work. They also examine various algorithms used in collaborative filtering. We build upon these algorithms in our recommender system's implementation. Their enumeration of considerations when evaluating a recommender system also informed our own evaluation.

Kembellec, Chartron, and Saleh also contribute a wonderful, slightly newer textbook on recommender systems, aptly named *Recommender Systems* (Kembellec, Chartron, & Saleh, 2014). They take a more business-application-oriented approach than Ricci et al., though they also provide an excellent overview of the recommender systems space along with some specific discussions of technologies. They also include some more recent developments in recommender systems that didn't exist at the time of Ricci et al.'s contribution.

Herlocker, Konstan, Terveen, and Riedl examine the key considerations when evaluating recommender systems (Herlocker, Konstan, Terveen, & Riedl, 2004). Their work was among the first to establish a deliberate methodology in recommender systems evaluation. They have generalized the evaluation process enough that their guidelines instruct our own evaluation.

Chapter 3

Thesis Statement

The Family History Source Recommender System can be used effectively in recommending sources that provide valuable information about individuals in a pedigree.

Chapter 4

Project Description

We create the Family History Source Recommender System in order to assist people who struggle to find sources in their family history work. We implement the Family History Source Recommender System such that when provided with a deceased individual from the FamilySearch tree, it generates a list of top recommended sources for that individual.

One significant challenge to successfully implementing a collaborative recommender system in any context is the cold start problem. Without a sufficient accumulation of data, collaborative recommender systems provide low-quality recommendations.

Luckily, FamilySearch provides an application programmer interface that grants access to their data. The available data includes documentation of sources attached to information on individuals in the FamilySearch tree. Consequently, our first step in implementing the Family History Source Recommender System is to crawl the existing source data in FamilySearch's database in order to seed the recommender engine and overcome the cold start problem.[1]

The source documentation in FamilySearch varies widely in quality. The source information on an individual is returned as simple text fields. Unfortunately, these fields are populated in a number of different ways, which complicates the recommendation of these sources. The field may be a user-entered source, which can range anywhere from a well-formed, specific reference, to a vague attempt at a source. FamilySearch encourages users to enter a source whenever they enter new information about an individual. Because of this, some of the data is useless (e.g., "I think my grandmother told me this once"). Even when

[1] See "Data Acquisition" for a full description of this process

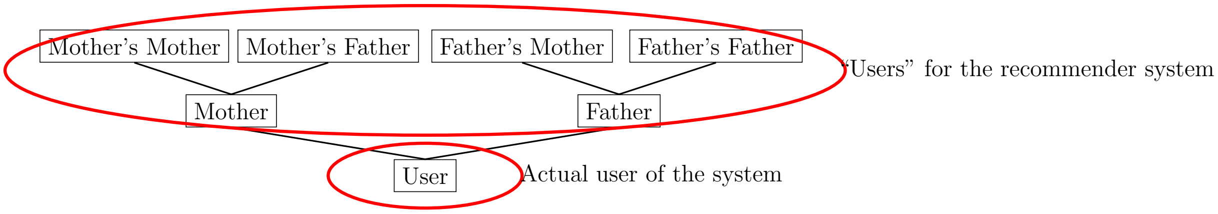

Figure 4.1: Users of the system vs. "users" from the recommender system's standpoint.

the user-entered data may actually be for a useful source, because it is free-entry text, it is difficult to parse. However, the source field may also be populated automatically by the FamilySearch source linker engine. When this is the case, FamilySearch holds a specific link to the source of the information, and the source field is specifically formatted to reflect that. These sources are not only more likely to be quality sources, but they are also much more automatically parsable.[2]

We extract these automatically-populated sources from the source fields. Parsing the user-entered data into a usable format could provide future benefit (see "Future Work").

One way in which our genealogical recommender system deviates from the recommender system norm is that users of the program and "users" in the recommender system sense are different. Our recommender system generates recommendations for sources for deceased individuals. The people who consume the recommendations are not directly considered by the recommender system. In contrast, in the typical recommender system setup, the user of the program and the user of the recommender system are one and the same (Figure 4.1 shows how users of the system differ from the deceased individuals for whom the recommender system generates recommendations).

Recommender systems rely on similarity metrics to determine how closely related two users are. This comparison is very important to the recommendation process. In a purely collaborative setting, the recommender system uses known usage patterns of users to compare them to each other.

In our genealogical context, the most valuable recommendations are the ones that

[2] See Appendix section "Source Quality" for more information

lead us to sources for individuals who do not yet have any sources attached at all. In a purely collaborative setting, these individuals would have too little information to compare them to others. As such, it is necessary to augment our similarity metric with as much information about the individuals as we have access to. Information like dates and locations of events (e.g., birth and death) for the individual are included in the comparison. It is also useful to include such information about the "one-hops" from the person. We define one-hops as people who are directly related to the person. This includes parents, children, spouses, and siblings.

With these goals in mind, the crawling and extraction of source data from FamilySearch results in a database, filled with people and sources. For each entry, the source has been parsed into a string representation of the archive in which it resides. The person object contains as much information as can be easily extracted from FamilySearch about the individual, as well as information about directly-related individuals. This facilitates similarity comparisons for nearly every individual in a tree, including the potential to automatically locate sources for people who are not yet known.

Generalizing the source information is another deviation from the standard recommender system. Typically, recommender systems operate in a sales environment where an individual might be given a recommendation for an item because there are many similar users who like that same item. The item is available and potentially applicable to all users. In the case of sources, however, the source information may be pointing to a specific record, which only holds information on a single person.

Just because one individual is found in one record does not mean another individual could be found in that same record. In fact, it might mean that they could not. However, records are usually found in record collections, which often contain similar records. For this reason, we generalize the source information so that it no longer refers to a specific record, but instead refers to a collection or archive of records. In a sales environment, this might be equated to a recommender system suggesting that the user buy a television in general,

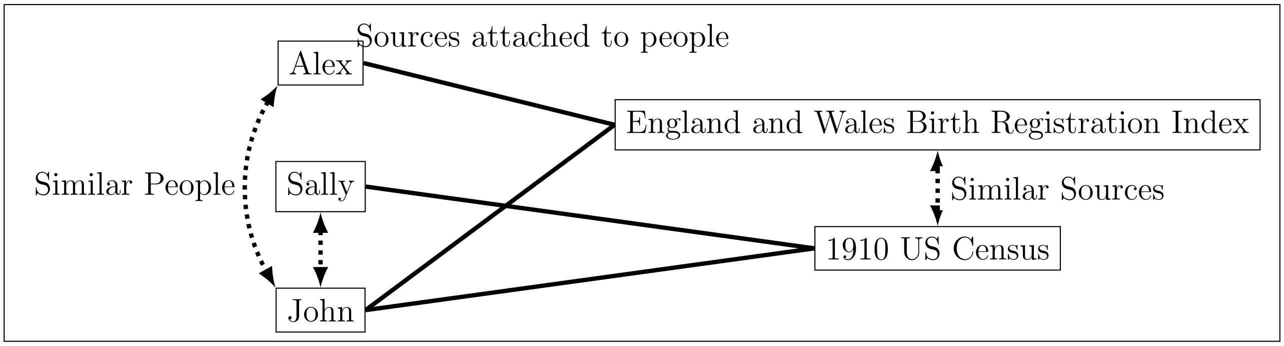
Sources attached to people
Alex
Sally
John
Similar People
England and Wales Birth Registration Index
Similar Sources
1910 US Census

Figure 4.2: Examination of micro-level collaborative relationships that create similarities between individuals and sources.

rather than a specific brand and model of television.

Using the crawled and extracted data, recommendations are generated by looking at the relationships between individuals and sources. The graph of the individuals and sources (with attached sources for an individual resulting in an edge between the source and the individual) is a bipartite graph, as shown in Figure 4.2. One way of looking at collaborative recommendation algorithms is by considering that they perform a transitive closure on the graph, generating ties between individuals and ties between sources. The ties between individuals represent similarity of individuals; the ties between sources represent similarity of sources.

However, this method would create a fairly minimal recommender system, since it has no way to give relative strength to the connections. Without that, ranking recommendations is impossible. This situation is rectified by adding edge weights that take into account the number of shared connections, or by adding another similar metric.

Finding the right variation on the method of ranking recommendations is essential to an optimal algorithm. In a bachelor's program honors thesis, the author discusses some approaches to incorporating time proximity into recommendation rankings (Brinton, 2012). We borrow this approach and take a similar approach to incorporate geophysical proximity.

However, as the amount of data increases, the difficulty of performing the transitive closure also increases. For this reason, most recommendation algorithms take a different approach, typically only computing the top recommendations. One example of this is the nearest neighbor algorithm. For our purposes, we use a simple metric as a filter to find nearest neighbors. We then use a more involved similarity computation to compute recommendations on those nearest neighbors.

As previously mentioned, similarity between individuals includes comparisons of known information about the individuals, rather than just being based on patterns in their attached sources. This extra information allows important similarities to exist where the graph would otherwise not show similarity. 32% of the individuals in our dataset have no source information attached to them, making it impossible to provide recommendations for them in a traditional collaborative recommender system. However, adding in this extra element of similarity for individuals allows us to give recommendations where they wouldn't have been possible otherwise.

Chapter 5

Survey

5.1 Data Acquisition

As previously mentioned, one potential drawback to collaborative recommender systems is the cold start problem. A successful recommender system not only thrives on large amounts of data, but requires extensive data for the system to return meaningful results. As such, we require access to ample data for our system.

FamilySearch provides an application programmer interface (API) to allow external entities to access their vast genealogical data. Few other genealogy companies have such open access to their data, which makes FamilySearch a logical choice. That being said, using FamilySearch for the data source has some drawbacks.

First, FamilySearch's API was not designed to allow data to be accessed at random. Instead, data is retrieved in relation to a known starting point. For our part, we retrieve the data by making jumps to parents, children, and spouses. We use the author as the starting point. At each iteration of the data retrieval process, the system alternates between retrieving parents of all known people and retrieving children of all known people. Additionally, it retrieves the spouses of all known people at each iteration. Naturally, each retrieval increases the total number of known people.[1] Given the limitations, there is little reason to think that another approach would be better, but it is important to recognize the potential bias that this approach introduces to the system.

Second, FamilySearch data is highly error-prone. Most of the fields on a given individ-

[1] See "Data Acquisition" for further explanation of this process.

Table 5.1: Data Composition

Statistic	Cardinality	%	Range	Std. Dev.	Mean	Median	Mode
Have Events	151588	99.96%	1 - 41412	137.988	61.746	30	6
Have Sources	103565	68.29%	1 - 187	7.934	8.929	7	1
Have A Parent	83950	55.36%	1 - 6	0.107	1.010	1	1
Have A Child	43966	28.99%	1 - 67	4.001	4.384	3	1
Have A Spouse	92328	60.88%	1 - 198	0.910	1.162	1	1
Have A Sibling	85528	56.40%	1 - 75	7.101	9.495	9	9

ual are user-entered. This means that automatically parsing simple data points is difficult. For example, there are numerous valid ways to represent the same date. On top of that, user entry introduces typos, formatting errors, and bad data. As an example of bad data, we have seen some date fields that contain a written location or name (users sometimes confuse which field is which in data entry). This introduces complexities in cleaning and preparing the data. We will discuss our solutions later.

5.2 Data Composition

We run our crawling algorithm, saving the aforementioned data into a database. After cleaning, we are left with 151,649 individuals in the dataset. Table 5.1 provides some pertinent statistics about the composition of data for these 151,649 individuals. It explores those individuals who have at least one event attached, one source attached, one parent, one child, one spouse, and one sibling, along with the statistics for how many are actually attached. The statistics shown are only for those individuals with at least one (event, source, parent, etc.).

To help understand the above table, the first entry can be read as follows: 151,588 (99.96%) of the 151,649 people have at least one event attached to them. Of those with events, the number of events ranges from 1 to 41,412 with a standard deviation of 137.988, mean of 61.746, median of 30, and mode of 6.

The unusually high values on the upper end of the ranges for these statistics illustrates the error-prone nature of FamilySearch data which we discussed earlier. Data is often duplicated in FamilySearch to represent varying possible answers. The individual with 41,412 events attached has many events duplicated many times in FamilySearch. Our system is engineered to handle these duplications. We explain the solutions later.

Chapter 6

Implementation

6.1 Data Acquisition

Kinpoint, Inc., a genealogy company that the author co-founded, has developed an engine for interfacing with FamilySearch. For convenience, we use this engine to gather our data. It parses the data from FamilySearch, saving it into a more easily-accessed format inside a Redis datastore. As far as the data itself is concerned, it is no different than what could have been accessed directly from FamilySearch. Furthermore, the author created a traversal engine for Kinpoint, which we utilize in retrieving the data from FamilySearch. Given a string description of a traversal, this engine follows the traversal to its end, bringing back all person information along the way.

Since data cannot be retrieved from FamilySearch at random, we instruct the Kinpoint engine to start at the author and alternately retrieve parents and children of everyone in the set of known people. It retrieves spouses at each step. We ran this process for several days, pulling down 151,649 individuals from FamilySearch. Once the traversal is finished, a simple python script dumps all the data from the Redis datastore into a PostgreSQL database for our usage.

6.2 Data Parsing

As described previously, most of the data fields from FamilySearch are user-entered. This makes parsing the data complex and cumbersome. It also means that data is often missing or unparsable. For our purposes, we parse only date and location fields.

6.2.1 Dates

Our similarity computation relies partially on date fields being parsed in order to perform date proximity comparisons. Of the fields we use, date fields are the most challenging to parse correctly.

As an example of the complexities that can arise in user-entered date formats, a very common date representation follows a format similar to `##/##/##`. In the United States, this is often given as `MM/DD/YY`. However, in other situations, it is often given as `DD/MM/YY`. Now, imagine that a user enters a date as `5/8/29`. Does this represent the 8th day of May, or the 5th day of August? Such ambiguous date formats make it difficult to know if the date is being parsed correctly. Even more damaging is the potential that the century might also be misunderstood.

Another major challenge in date parsing comes with incomplete dates. Sometimes, a date is entered as nothing more than "`January`" or "`1857`." We create a special date container that allows portions of a date to be missing, while still providing similarity computations. Allowing a portion of a date to exist in our system is useless, unless we allow partial dates to be parsed from the user-entered date strings.

Ultimately, a complex regular expression catches the majority of the myriad date formats that may have been entered. When matched using case insensitivity, the regular expression matches 99.98% of the dates in our system. We create this regular expression to handle the many unusual ways users enter dates in FamilySearch. For example, "`About June 20th, 1907`" or "`Around 5/13/43`" are common, as are date ranges, which can be entered in many ways as well (e.g., "`From 1895 through 1904`"). In the case of the date range, for simplicity's sake, we parse out the first date, discarding the second. Our regex handles these unusual challenges better than other date parsing libraries we encountered.

The appendix contains the fully-expanded regex pattern we use for date parsing. However, there are many portions of the pattern that repeat. In code, we handle these various components individually, so as to more easily retrieve the day, month, and year

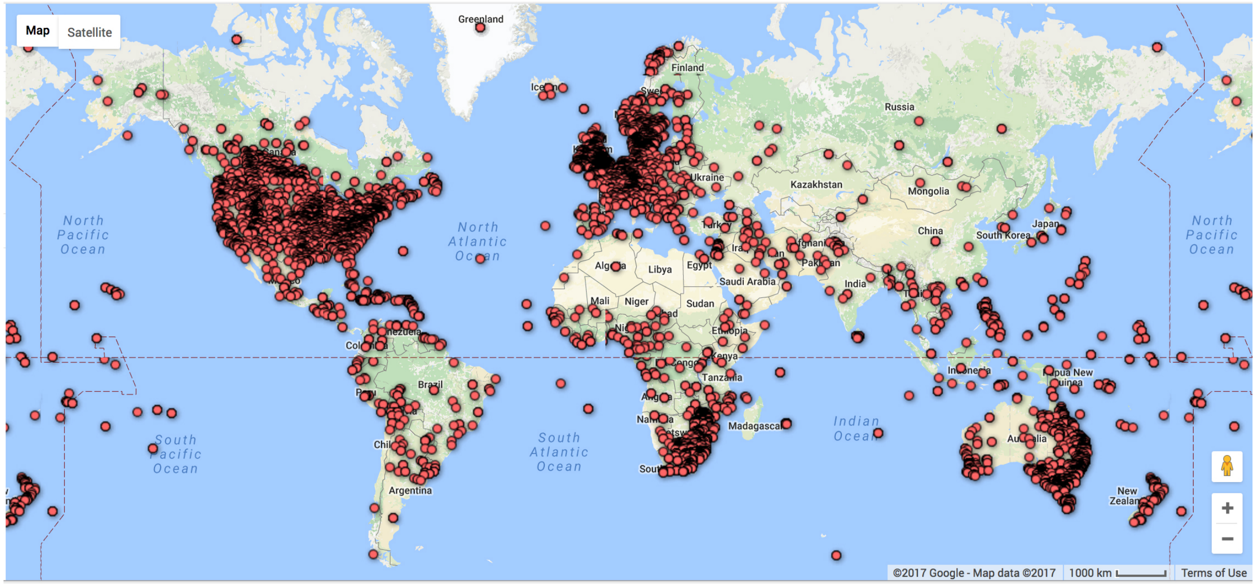

Figure 6.1: Geocoded Locations, Mapped to the World

represented in the date string.

6.2.2 Locations

Location parsing poses less of a challenge than date parsing because we rely on FamilySearch's API to perform the parsing for us. FamilySearch provides a geolocation endpoint which accepts a location as a string and returns an ordered list of guesses at locations (with latitude and longitude) that match the query. The first item in the list is FamilySearch's guess at the most likely candidate, and the list is empty when the query is bogus.

Initially, we surveyed the various geolocation APIs that exist, before ultimately choosing the FamilySearch API for geocoding. Google, Yahoo, Bing, and MapQuest, to name a few of the more well-known geolocation APIs, all provide free geocoding. However, all of them suffer from a condition that we will call currentism. Their focus is on the current state of the world, not the world as it may have existed in the past. As an example, Google's geocoding API places "Prussia" somewhere in Iowa, USA. FamilySearch is the only API we are aware of that provides historical as well as current geocoding of locations. It is also capable of providing boundary data, which could prove useful in future work.

We rely on FamilySearch's API to make intelligent choices in the geocoding process. That is to say, we accept the first result that FamilySearch returns as the answer, since we have no easily-accessible data that might lead us to question the API's choice. Figure 6.1 shows the results of geocoding our event data, mapped using Google Fusion Tables, a feature of Google Documents. The majority of locations represented in the dataset come from either the United States or Europe. Some of the points outside of these areas appear to be false matches. Two interesting sample points come from original user-entered location strings: "ABOARD SHIP WM. Tapscott,,, At Sea" and "Drowned at sea." The geocoder correctly (at least as well as could be expected) places both of these points in the middle of the Atlantic Ocean.

6.2.3 Sources

Sources, as with other fields in our dataset, are user-entered. For our purposes, we need to be able to recognize sources wherever they are repeated in the dataset. There is, however, one kind of source which FamilySearch enters into the source field automatically, in an easily-parsable format. Whenever a source which is known to FamilySearch is attached to an individual, it will be entered automatically in this way. Known sources to FamilySearch may be either indexed or non-indexed. With no readily available methods for parsing the rest of the source information, we focus our attention on these known, parsable sources. FamilySearch provides collection IDs to go along with known sources. These collection IDs are useful in grouping sources.

6.3 Similarity Computation

In a purely collaborative recommender system, similarity is computed by comparing known profiles of individuals. These known profiles allow a similarity computation between individuals. The way this typically works is for the system to judge two users to be highly similar when they have similar purchase histories. When brand new users begin using the system, there is no known information about them that can be harnessed in computing their similarity to other users. Collaborative recommender systems will usually create a rapid onboarding process to overcome this hurdle. This can be done by asking the user to provide some personal information, using the user's site navigation behavior (such as the searches they perform or the products they view), gathering browser and location data, and/or gleaning user information from the user's online presence on other sites.

For our purposes, the "users" in play in our recommender system represent deceased individuals. This makes the aforementioned methods impossible. Instead, we leverage the information we do know about an individual in order to generate a profile. Without this step, we would only be capable of providing recommendations of new sources for those individuals who already have at least one source attached to them. Those individuals with at least one source already attached are typically the ones for whom a new source is the easiest to find. They are also the ones for whom another source is usually the least beneficial. Furthermore, in juxtaposition with the typical recommender system, the nature of our dataset and genealogy work in general is such that those with no sources attached make up a significant chunk of the "user" base of the system. Our supposition is that if our system can generate quality recommendations for these individuals, it will be indicative of its utility and power for the average genealogist.

Every individual in our dataset was retrieved by his or her relation to another individual in our dataset. Furthermore, individuals rarely exist in isolation in the larger FamilySearch database. We take advantage of these connections to establish a profile for individuals for whom we would otherwise have no information. As such, our similarity comparison algorithm is fairly complex. Comparing two individuals involves comparing everything that we know about the individuals themselves, plus everything we know about all of the people related to them (parents, children, siblings, and spouses). Comparing only what we know about two individuals (not looking at their relationship) is called a base person comparison. A base person comparison involves comparing names, genders, and events (which are composed of dates and places that must each be compared). Comparing two individuals, then, is achieved by performing their own base comparison and aggregating their base comparison with the base comparisons of all parents, children, siblings, and spouses. At any step along the way, data may be missing, so our comparison algorithm is robust to missing data.

6.3.1 Inverted Sigmoid Falloff

A standard sigmoid of the form $f(x) = \frac{1}{1+e^{-x}}$ has a range from 0 to 1, where $f(x) \to 1$ as $x \to \infty$ and $f(x) \to 0$ as $x \to -\infty$. However, $f(-5) \approx 0.0067$ and $f(5) \approx 0.9933$. This means that almost the entirety of the range of a sigmoid function is contained from $x = -5 \to 5$. We use this to our advantage in generating a continuous falloff for several of our comparison functions. For our purposes, we desire a sigmoid falloff that has its maximum (close to 1) at $x = 0$, then falls off to nearly 0 over a specified domain. We want a maximum at $x = 0$ for most of our similarity computations, because the x-axis represents distance of some form. Closer items should be more highly related. A standard sigmoid curve increases as x increases, hitting $y = 0.5$ at $x = 0$. Therefore, in order to accomplish our requirements, we invert the sigmoid and shift it to the right by half of our desired domain, achieved by subtracting half of our desired domain from the x in the exponent. We then scale the inverted sigmoid horizontally by a factor of $\frac{domainSize}{10}$, achieved by multiplying the x component of the sigmoid (including the previous subtraction) by its inverse ($\frac{10}{domainSize}$). If desired, this scalar can be further adjusted to generate either a sharper or more gradual falloff by multiplying by an additional scalar. The resulting inverted sigmoid function follows.

$$\varsigma(x) = \frac{1}{1 + e^{(sharpnessScalar\frac{10}{domainSize}(x-\frac{domainSize}{2}))}}$$

6.3.2 Name Comparison

Names are compared by using a common form of edit distance, the Levenshtein Distance algorithm. To compute the name similarity, an inverted sigmoid falloff is applied to the result

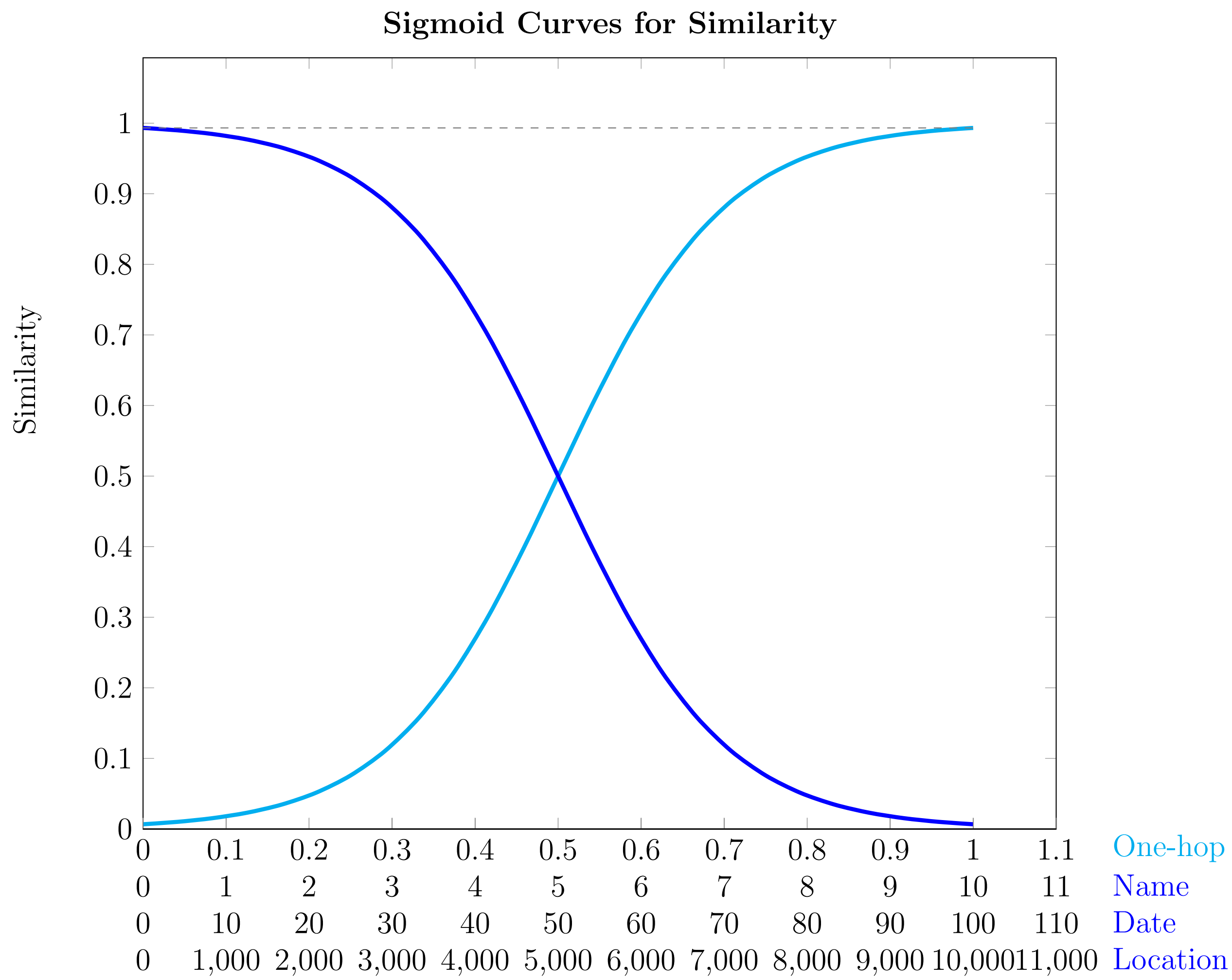

Figure 6.2: The inverted sigmoid falloff curve applies for name, date, and location comparisons, where the x-axis represents edit distance, years (very roughly), and meters, respectively. The normal sigmoid curve applies for one-hop similarity comparisons, where the x-axis represents our computed similarity. The y-axis for all four represents our output similarity.

of computing edit distance on the two names. The sigmoid attempts a falloff that ranges from maximum to minimum as edit distance ranges from 0 to 10. Beyond 10, the output will only grow closer to 0. To achieve this, we don't need to scale the sigmoid horizontally, but we do need to shift it to the right by 5 (half the domain). This yields the following overall function (Figure 6.2 shows it graphically on the "Name" x-axis).

$$\varsigma(x) = \frac{1}{1 + e^{(x-5)}}$$

Finally, the Base Person Comparison downscales the name computation fairly heavily (achieved via a scalar that we apply to the comparison). This is due to the assumption that people's names have little bearing on the sources which contain information about them (see Future Work for more information).

6.3.3 Gender Comparison

Gender comparison is simple. If the two genders are the same, then the resulting similarity value is 1. Otherwise, it is 0. However, the Base Person Comparison downscales the gender computation fairly heavily. This is based on the assumption that people's genders are likely not a significant determining factor for the sources which contain information about them.

6.3.4 Date Comparison

After successfully parsing a date, we are left with a date container, which is capable of holding the day, month, and year for a given date. The comparison of two dates is performed by separately comparing the day, the month, and the year. Each component has a multiplicative scalar, and the resulting comparisons are added together to generate a value that will be used to create the final comparison value. While configurable, our current implementation uses a day scalar of 0.01, a month scalar of 0.1, and a year scalar of 1.0. This resulting sum is then used as the x-value in an inverted sigmoid function, designed to give a maximum value when the difference between the dates is minimized. This yields the final value for the comparison.

The reason for comparing each component of the date individually is that some dates exist as nothing more than a year, a month, or a day. We want two dates to be comparable even when components are missing. Furthermore, when no known components line up to be compared, the output is 0. When a single component cannot be compared, because one date or the other is missing the component, a fixed amount is added to the total computation. That amount varies depending on the component. For example, year has the highest effect on the outcome and is fixed at 50 (halfway through our domain) when unknown.

We desire a sigmoid falloff that ranges from maximum to minimum over the course of roughly 100 years (a gross approximation of the length of a lifetime). To achieve this, we shift the sigmoid to the right by half of our desired domain (50), achieved by subtracting 50 from the x in the exponent. We then scale the inverted sigmoid horizontally by a factor of 10, achieved by multiplying the x component of the sigmoid (including the previous subtraction) by 0.1. The resulting inverted sigmoid function follows, and is shown graphically in figure 6.2 (see “Date” on the x-axis).

$$\varsigma(x) = \frac{1}{1 + e^{(0.1*(x-50))}}$$

The final equation for date similarity after substituting our date comparison sum for x follows.

$$dateSimilarity = \frac{1}{1 + e^{(0.1*(0.01*|\Delta Day|+0.1*|\Delta Month|+1.0*|\Delta Year|-50))}}$$

This equation yields a value ≈ 1 when all known components of the dates to compare are identical.

6.3.5 Location Comparison

Location (place) comparison is fairly straightforward. Our parsing step provides us with latitude and longitude values for each place in our dataset. We use a modification of the

Haversine formula to compute distance from latitude and longitude. We convert the distance to meters, so as to make sense of the results when we perform our falloff.

Our initial attempt at distance comparison used PostgreSQL's built-in geometry functions to perform distance computations. However, these functions, after some optimization, still did not run fast enough for our purposes. We found that using the latitude and longitude values to compute the distance manually was much faster.

After calculating the distance between the two locations, based on the latitude and longitude of each, we put the distance through an inverted sigmoid falloff to determine the similarity. We choose a sigmoid falloff that ranges from maximum to minimum over the course of roughly 100 kilometers. To achieve this, we shift the sigmoid to the right by 50,000 (in meters) and scale the function horizontally by a factor of 10,000 by multiplying the x component of the sigmoid by 0.0001. The resulting inverted sigmoid function follows, and is shown graphically in figure 6.2 (see "Location" on the x-axis).

$$\varsigma(x) = \frac{1}{1 + e^{(0.0001*(x-50000))}}$$

This approach is convenient, but potentially flawed. To illustrate why, consider the example of three fictitious individuals: John, living in San Diego, California, United States; Pablo, living in Tijuana, Mexico; and James, living in Portland, Maine, United States. Suppose they were all born on the same day. Now consider that John and Pablo live a mere 15 kilometers from each other, though in different countries. However, John and James live more than 4,000 kilometers away from each other, though still in the same country. John and James may appear in the same census records, though John and Pablo are unlikely to share any such source records. With countries like Russia, these differences may be even more pronounced.

Ideally, a location similarity comparison would be capable of comparing not only distance, but regional boundaries as well. However, regional boundaries are subject to change. As such, as time passes, it becomes increasingly complex to know how these regional boundaries might affect the likelihood of two individuals sharing a source. Furthermore, when date information is missing, those boundaries may be impossible to know. Thus, we implement location similarity as nothing more than a Euclidean distance comparison and rely on a multitude of data to smooth out the imperfections.

6.3.6 Base Person Comparison

The Base Person Comparison compares all known data about two individuals without considering the people who are related to them. It is composed of the name comparison, the gender comparison, and date and place comparisons for each event attached to the individuals. The formula follows.

$$\begin{aligned} compare_{base}(p_1, p_2) = {} & c_{name} * compare_{name}(p_1, p_2) \\ & + c_{gender} * compare_{gender}(p_1, p_2) \\ & + c_{event} * \underset{val}{\operatorname{argmax}} \left(\forall e_1 \in p_1 \left(\forall e_2 \in p_2 \left\{ \begin{array}{l} val = c_{date} * compare_{date}(e_{1_{date}}, e_{2_{date}}) \\ \quad + c_{place} * compare_{place}(e_{1_{place}}, e_{2_{place}}) \\ \quad + c_{eventType} * compare_{eventType}(e_1, e_2) \end{array} \right\} \right) \right) \end{aligned} \tag{6.1}$$

p_1 and p_2 in the previous equation represent the two people being compared. e_1 and e_2 represent events contained in the person. c_{name}, c_{gender}, c_{event}, c_{date}, and c_{place} are all scalars. $compare_{eventType}$ is a function which determines whether the two events are of the same type. If they are, it returns 1; otherwise it returns 0. For example, suppose a birth event and death event are compared and computed to be of a certain similarity val. Now suppose that the same comparison is computed, but this time they are both birth events, with all else equal. The $compare_{eventType}$ function will ensure that the latter receives a higher overall similarity.

As may be obvious, the event comparison returns the similarity of only the most similar events between the two people, due to the argmax. Some individuals have more events attached than others, and were we to allow all events to contribute to overall similarity additively, those individuals with many events would have an immediate advantage. To remedy this, we choose to allow each individual to contribute only one highest-value event to the similarity. Other possible alternative solutions could include averaging all events, normalizing to the number of events, or scaling down similarities as a function of the number of events compared.

6.3.7 Parent, Child, Sibling, and Spouse Comparisons

Comparing similarities for the one-hop individuals involves computing the base person comparisons between each of the corresponding people. For example, the parent computation compares all of p_1's parents to all of p_2's parents, via the base person comparison. The parent comparison follows.

$$\begin{gathered} compare_{parent}(p_1, p_2) = \\ \varsigma\left(\underset{val}{\operatorname{argmax}}\left(\forall parent_1 \in p_{1_{parents}}\left(\forall parent_2 \in p_{2_{parents}}\{val = compare_{base}(parent_1, parent_2)\}\right)\right)\right) \end{gathered} \tag{6.2}$$

Comparison computations for child, sibling, and spouse all look similar. As with the event comparison, and for a similar reason, we accept only the highest similarity as the value for the whole comparison. The sigmoid curve for these comparisons is different from the others to prevent it from being inverted, as similarity should increase as the one-hop individual comparison computations increase. With our other comparisons, we are comparing proximity, so larger x-values should result in lower similarities. But with one-hop comparisons, a larger x-value relates to a higher similarity. A range from minimum to maximum over a domain of 0 to 1 maximizes comparability. To achieve this, we shift the sigmoid to the right by 0.5, achieved by subtracting 0.5 from the x in the exponent. We then scale the sigmoid horizontally by a factor of -0.1, achieved by multiplying the x component of the sigmoid (including the previous subtraction) by -10. The resulting sigmoid function follows, and is shown graphically in figure 6.2 (see “One-hop” on the x-axis).

$$\varsigma(x) = \frac{1}{1 + e^{(-10*(x-0.5))}}$$

6.3.8 Person Comparison

The final piece to our similarity metric is the full person comparison. This comparison combines the base person compare function for the people in question with all of the one-hop relationships connected to them. The function follows.

$$\begin{aligned} compare_{person}(p_1, p_2) = &c_{base} * compare_{base}(p_1, p_2) \\ &+ c_{parent} * compare_{parent}(p_1, p_2) \\ &+ c_{child} * compare_{child}(p_1, p_2) \\ &+ c_{sibling} * compare_{sibling}(p_1, p_2) \\ &+ c_{spouse} * compare_{spouse}(p_1, p_2) \end{aligned} \tag{6.3}$$

What is the range of this function? Each component is clamped to the range of a sigmoid. With five individual components, we have a maximal output of $\approx c_{base} * (c_{name} + c_{gender} + c_{date} + c_{place} + c_{eventType}) + c_{parent} + c_{child} + c_{sibling} + c_{spouse}$. The actual maximum of the range will be slightly less than this computation, since several of the components' sigmoid functions achieve a maximal similarity output ≈ 0.9933.

Similarity metrics often attempt to force similarity into a range between 0 and 1, where 0 represents no similarity, and 1 represents complete similarity. This is useful when using similarity as a scalar for recommendations or other similar computations. We normalize the similarities before proceeding with the recommendation computation. To perform this normalization, we compute the similarity of the subject with him/herself. All similarities are divided by this value to give a normalized similarity. This may result in the most similar person's computed similarity being a fairly low number, which is desirable for giving an idea of how likely it is that the recommendations are correct.

From a runtime standpoint, this computation is somewhat expensive. While a single similarity comparison between individuals is measured in milliseconds, with 151,649 potential individuals to compare, we cannot possibly compute the similarity between all individuals in the database. Even if a single similarity takes merely 10 milliseconds (a conservative estimate), the following computation shows that this would take several years to compute for all individuals.

$$\begin{aligned} C_2^{151649} = \frac{151649!}{2!(151649-2)!} &= 11498633776 \text{ comparisons @10ms each} \\ &= \frac{11498633776}{100ms * 60s * 60m * 24h * 365.2425d} \text{ years} \\ &= 3.64 \text{ years} \end{aligned} \tag{6.4}$$

Furthermore, the results would need to be stored, requiring 8 bytes for the similarity result (double precision), and 8 more bytes to store the references to the people. Using the results from our data, this would be around 184 GB of data. While that is not unthinkably large, an ideal implementation of this system would utilize a much larger dataset and would quickly accumulate excessive storage requirements. The amount of required storage grows factorially with the number of people in the dataset.

For these reasons, computing recommendations for an individual must be performed on a subset of the total dataset.

6.4 Working Set Selection

In order to maintain a reasonable runtime, we limit our working set to a small subset of the total database. Intelligent selection of this subset is crucial. If the subset is nothing more than a random selection, then we gain no benefit from the larger dataset we have at our disposal. However, the selection must run relatively quickly. Running our complete similarity computation to determine the k nearest neighbors, while optimal in terms of output, is not a feasible task because of runtime. Instead, we use a combination of date and location information as a blocking factor to gather those people who are most likely to be similar. These are the candidates for high similarity which we will run through the full similarity computation.

One other (adjustable) criterion for these candidates is that they must have at least one source attached to them to be considered a candidate. As far as recommendation computation is concerned, if we are looking for sources for the subject, there is no point in considering individuals with no sources attached. However, the system could also be used to return recommendations in the form of similar individuals instead of sources. In this case, requiring that the candidates have sources attached may be undesirable. This might be a useful addition to giving sources as recommendations. We discuss this concept further in *Future Work*.

6.4.1 The Simple Case

In the simplest case, the user petitions the system for recommendations for a specific individual, called the subject. The system queries the database for known information about the subject. If none of the events attached to the subject contain date or location information, then we move on to the more complex cases. Otherwise, we select from the database those individuals who are closest to the known date and location information attached to the subject.

For each date/location pair attached to the subject, this closeness selection finds a predefined number of people who are geographically closest to the location. It orders those results by date proximity to the corresponding date from the subject. It limits the number of people returned at this intermediate step with a predefined limit. Then it groups together all the remaining people and condenses them down to their ID and a number (being the number of results that made it into the limited group for that individual). Once this has been done on all date/location pairs, we are left with a mapping of people to their number of occurrences (at least one) for each date/location pair. We join these into one single mapping, adding occurrences when a person appears multiple times. We order the final mapping by number of occurrences and take the top n people from this ordering. These become the candidates for high similarity.

Once the candidates have been selected, the full person similarity comparison is computed between each of the candidates and the subject. We reorder by similarity and limit the total to a predefined number. At this point we have converted the candidates into a working set of known similarities.

6.4.2 The Complex Case

In the more complex case, when the subject has no date or location information attached to them, the process is similar to the simple case, but the events from the subject cannot be used for selection. Instead, the events from the parents, children, siblings, and spouses of the person are used for selection. This means performing the simple case on all parents, children, siblings, and spouses. Then, once the resulting people are returned, we follow the relationships in reverse. For example, the candidates returned by performing the simple case comparison on the parents of the subject are actually the parents of the individuals that we should consider as candidates for the subject. We have to take one more step and grab any children of those people in order to find the actual candidates. We handle this separately from the simple case because the data necessary for the simple case computation does not exist in the complex case. We perform this reverse relationship traversal for all four categories of relationship. We then combine all results, compute all similarities, reorder, and limit, much as we do at the end of the simple case, to generate our working set.

6.5 Recommended Source Agglomeration

With a working set of known similar individuals to the subject, recommendation computation is trivial.

To compute recommendations, we loop through all individuals in our working set and construct a mapping of sources to scores. When a source is encountered that is not already in our mapping, we put the source into the mapping with a score equal to the similarity for the current individual. When a source is encountered that is already in the mapping, we simply add the current individual's similarity to the source's score.

By the end of the loop, we have a fully-constructed mapping of sources to scores. We order this mapping by score and return the top ten results. At this point, the overall scores for each source could be used to generate an idea of the accuracy of the top ten recommendations. Doing this would require an understanding of the scores that are typically associated with useful sources, as compared to useless sources.

6.6 Validation

From an implementation standpoint, validation requires some modifications. We enhance our system to allow individual fields from any record to be "locked." This is important in being able to establish a measurable testing environment.

We perform validation using a leave-one-out cross-validation method. In the ideal scenario, we could use all of the data at some point in the test. However, our system does not gain many performance improvements when running with multiple test subjects simultaneously. Consequently, leave-one-out cross-validation is the better choice over n-fold cross-validation. Running leave-one-out cross-validation across the entire dataset would take longer than is feasible, so we settle for random selection from the dataset in our validation.

When performing our validation, the test subject always has its sources locked. This means that the system cannot determine on its own which sources are attached to the individual in question. This is an obvious and essential step in the validation.

Additionally, the validation can lock individual components of the test subject's record. For example, perhaps we will want to validate the system when the test subject is a woman for whom we know nothing aside from her relationships. Locking individual components allows us to examine how well our similarity metric is able to locate similar people, even when we know very little or nothing about the subject. We validate each test subject first with only sources locked, then we lock event information, then we lock all known fields on the subject, and finally, we lock all relationships connected to the person except the parental relationships. This process allows us to calculate the loss and the relative performance when subjects are missing vital information. All in all, we compute recommendations four times for each test subject.

For our validation, we select randomly from the database those individuals that have at least one source attached to them. Without at least one source, we cannot know if our recommendations are correct.

After computing recommendations for the test subject, we examine how many of the top ten recommendations matched the actual sources. These results are saved to a CSV file for analysis.

Chapter 7

Results

We have already discussed the necessary adjustments we implement for validation purposes. We ran the validation process for a little over a week. During this time, it ran leave-one-out cross-validation on 3,456 random subjects from the dataset.

For each subject, it first restricted information about the subject by locking the known attached sources. It then requested source recommendations for the subject. It compared the top ten recommendations to the (locked) sources that were actually attached to the subject. We consider the presence of a source in the top ten recommendations a "hit" if that source was actually attached to the subject. The validation process computed the number of hits and stored it with the total number of actual sources attached to the subject.

The choice to return ten recommendations to the user was made for two reasons. First, ten is an easy number of recommendations for a user to evaluate. Providing the user with too many recommendations can be overwhelming, while providing too few may not yield the results we seek. Furthermore, ten is a common number of recommendations for a recommender system to return (likely for similar reasons to our first). We did not, however, tune the number of recommendations to give us the most desirable results in our validation. From a statistical standpoint, such tuning would likely qualify as data snooping.

After computing the number of hits in the top ten recommendations, the validation then performed this process three more times, each time progressively locking more subject data to restrict the recommender's access. It recorded additional hit information after locking

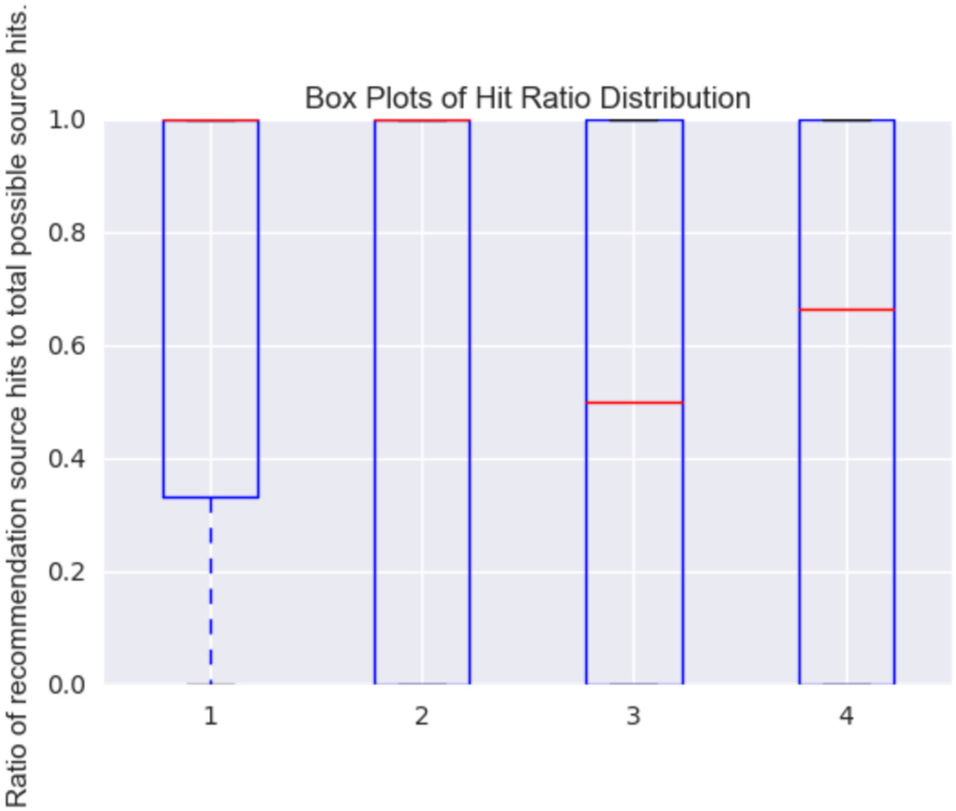

Figure 7.1: 1 = Locked sources. 2 = Locked events. 3 = All subject information locked. 4 = Only parent relationships visible. Notice that as the validation process locks more information to restrict access during the recommendation computation, the hit ratio tends to fall.

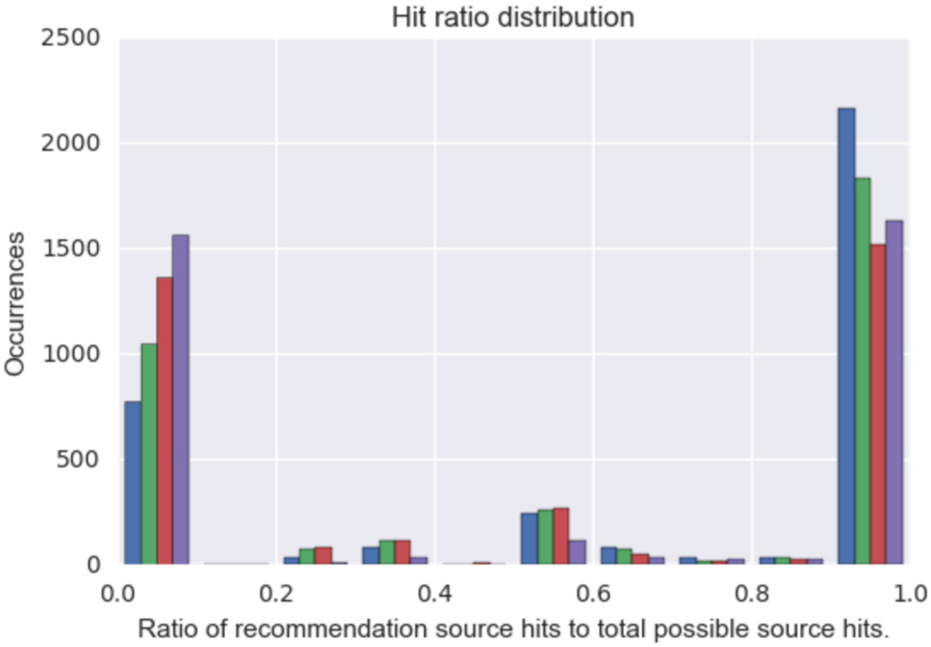

Figure 7.2: Blue = Locked sources. Green = Locked events. Red = All subject information locked. Purple = Only parent relationships visible. Notice that with only parent relationships visible, there is a higher incidence of complete misses, but there is also a higher incidence of success when there is information available.

all events attached to the individual, after locking all known information about the individual (aside from relationships to others), and finally after locking all relationships except parent relationships.

This validation process allows us to see how well the system provides recommendations for sources when the subject has varying levels of known information about him/her. We examine both the ratio of hits to total known sources, as well as the overall cardinality of subjects for whom the recommender successfully provided even one recommendation.

The distribution of these hit ratios helps to explain the success of the recommender. Figure 7.1 shows a box plot of the hit ratios at each level of information restriction. A hit ratio of 1 means the recommender returned all of the known sources for the subject in the top ten recommendations, where a hit ratio of 0 means that none of them were returned. Because of this, the values can range from 0 to 1, with a significant amount of the data being either at 0 or 1. As a result, we see no outliers, and most of the plots fall within the interquartile range. The first two box plots, where only the sources are withheld and where sources and events are withheld, are the only ones that have the median at 1. As more data is withheld or locked, the values tend more towards 0, and we see the median drop below 1. This box plot shows us that, as would be expected, subjects for whom we have some information appear to perform much better than when information is lacking.

A histogram of the hit ratios, as in Figure 7.2, gives us a more complete view of the data than the box plots. Figure 7.2 is color-coded to show the results for the four different test types (with varying levels of information restriction, as discussed previously). The number of subjects for which the system recommended all known sources appear on the right with a hit ratio of 1, while those subjects for which the system recommended none of the known sources appear on the left with a hit ratio of 0. Those in the middle represent some gradation between the system having recommended none and all of the known sources.

A surprising result is that when the only information about the subject is his/her parent relationships, we had more perfect hits (finding all known sources for the subject)

Table 7.1: Validation Results

Restriction Level	Hit Percentage	At-Least-One Hit Percentage
Sources only	68.99%	77.66%
Events locked	58.18%	69.76%
All information locked	49.49%	60.65%
All relationships but parents locked	53.14%	54.83%

than when all relationship information is known. However, this level of restriction also had more complete misses (not finding a single known source for the subject). This behavior may be explained by acknowledging that some individuals don't have any parent relationships that are known to the system, while still having other relationships that are known. Perhaps the parent relationship is the most reliable, and when others exist in the mix, they lower the accuracy. Yet when there is no known parent relationship, we are better off being able to compute similarity using other known relationships. Otherwise, we would invariably come up empty-handed.

Table 7.1 shows both overall hits and at-least-one-hit results from the validation. Overall hit percentage tells the average percentage of known sources that were recommended by the system, while at-least-one-hit percentage tells the percentage of subjects for whom the system recommended at least one known source.

Table 7.2 shows an example of validation for a single individual, chosen randomly from the database. The numbers in the figure indicate the order in which the recommendations were returned, and the asterisks (*) indicate known sources. Elizabeth Laurette Peart only has four sources attached to her, which means that there are only four known sources. The first test, which only hides the known sources from the system, found all four known sources and returned them as the first, third, fifth, and sixth recommendations.

One of the most challenging parts of this process is determining what level of accuracy would be deemed acceptable. Is it adequate for our system to return a valid result for three out of four individuals? Our experience with source discovery in genealogical research

	Utah Marriages, 1887-1935	United States Census, 1930	Salt Lake County Death Records, 1849-1949	England Births and Christenings, 1538-1975	Utah Deaths and Burials, 1888-1946	Utah Death Certificates, 1904-1964	Sweden Baptisms, 1611-1920	Utah, Early Mormon Missionary Database	New Hampshire Births and Christenings, 1714-1904	Massachusetts Deaths and Burials, 1795-1910	Utah, County Marriages, 1887-1940	United States Social Security Death Index	Vermont Vital Records, 1760-1954	England, Dorset, Parish Registers, 1538-1936	Find A Grave Index	Massachusetts Marriages, 1695-1910	United States Passport Applications, 1795-1925
Sources only	*1	2	*3	4	*5	*6	7	8	9	10							
Events locked	*1			3		*4	7				2	5	6	8	9	10	
All information locked	*1		*2	4	*3	*6		9				5		7	8		10
All but parents locked																	

Table 7.2: Validation results from a randomly chosen individual, Elizabeth Laurette Peart (1864-1934). She has no known parents, which results in no recommendations when everything but parents is locked. She has one spouse, 10 children, 4 attached sources, and 9 attached events. 7 of those events occurred in Salt Lake City, Utah, with 2 in Logan, Utah. * indicates a match to a known source.

suggests that the answer would be a resounding “yes!” Furthermore, while the accuracy certainly drops some when restricting subject information, we expected a much larger drop in accuracy. Therefore, perhaps the largest success of this system is its ability to maintain relatively high accuracy, even when confronted with missing or incomplete data, which is typical in family history datasets.

Chapter 8

Discussion

Genealogists often hit walls in their research. Perhaps they encounter a record that suggests that someone had a child, but no further information is forthcoming, and the trail goes dark. Some people spend a lifetime searching in order to learn about such mystery individuals. The surprisingly high accuracy of this system in locating sources for individuals for whom no information is known suggests that this tool may provide a welcome addition to the genealogist's tool belt.

Furthermore, amateur genealogists often lack the knowledge or research skills necessary to find sources themselves, even when the trail hasn't gone dark. In such circumstances, this tool can be an easy starting point in locating sources for individuals.

As described previously, this system makes use of the relatives connected to the subject in order to aid in finding sources for the subject. This practice is common in the world of family history research. Family history expert Elizabeth Shown Mills describes this process:

> Humans cannot be understood when isolated from their environment or their kith and kin. Collaterals—the family members from whom one does not descend—are just as important to research as the direct line...genealogical research is a journey in which one roams through many branches of a family, following a tangle of lines that are likely to be parallel, perpendicular, diagonal, and circular as well. The researcher who explores this maze and watches for significant markers along the way can eventually accumulate enough of them to pave a quite clear path around

> the record gaps, the common names, and the ambiguous identities that plague most frontier research. (Mills, 2003b, p. 28–30).

Mills describes here the importance of exploring the connections surrounding an individual in order to find the missing information. Our recommender system performs a similar exploration. Our results indicate that it does so successfully.

8.1 Limitations

As has been mentioned throughout this thesis, this system has several limitations and biases as it currently stands. While these do not invalidate the system or the findings of this thesis, they are important to understand and consider.

Our current implementation uses only data from FamilySearch. As such, all the biases that already exist in FamilySearch data are present in our data as well. For example, FamilySearch, owned by the Church of Jesus Christ of Latter-day Saints, is likely skewed toward the demographics of its church members. Also, data validity is only as good as the users that enter the data. A system exclusively for professional genealogists would likely provide higher quality data than what we use here.

Sources in FamilySearch are user-edited text fields. This makes reliable parsing difficult. However, FamilySearch automatically populates this field when the source is from a known FamilySearch collection. We limit our recommendations to these collections, which represent a small fraction of what could be recommended were there a more reliable method of parsing and generalizing sources.

Because we retrieve our data relative to a single individual (the author), there is inherent bias in the dataset beyond the bias that exists in FamilySearch as a whole. Although our retrieval algorithm is designed to branch quickly to people who are not related to the originating person, just by nature of the relative retrieval, a random individual from our dataset is more likely to be related to the starting individual than is a random individual from the FamilySearch dataset as a whole.

Our leave-one-out validation involves locking known information and sources on the left-out individuals, generating recommendations for them, then examining the accuracy of the recommended sources relative to the actual sources that are attached to the individuals. However, there may be an inherent difference between individuals who lack sources in practice and those who have sources, but for whom we hide the sources.

8.2 Future Work

Introducing this recommender system into an environment where users can start utilizing its suggestions to help their genealogical research efforts would be a wonderful next step. The results from our validation suggest that it might be ready to provide useful recommendations for source discovery. However, with an active user base, parameter tweaking and optimization become even easier. Additionally, user feedback would provide further validation of the system and its utility. If such further validation is required before releasing the system to an active user base, some intermediary validation could be useful. This might involve controlled tests in which random individuals provide their own ancestors to the system, which would then provide recommendations. They would then manually examine these recommendations to determine their utility.

Recommender systems can only be as powerful as the data behind them. While our current system leverages a decently-sized data set, a production environment will need to use as much data as possible. Providing the system with a much larger data set will be important for ensuring that it provides the best recommendations possible. Such an increase in data set size may require further optimization.

Another possible future area of focus for this system would be to include a name recommender. When name information is missing on an individual, even though the top ten recommendations from the system may include valid sources for the individual, it is very difficult for the user to recognize that without knowing what name to look for in the source. A name recommender may suggest the most likely names for the individual as a starting point for a name to search for in the recommended sources. Such a recommender would include a much more robust name comparison, which might merit increasing the weight given to name comparisons in the overall comparison computation. Such a recommender would probably have low accuracy, but the potential benefit when correct would likely make it a valuable addition.

As mentioned previously, the source fields returned from FamilySearch are primarily user-entered. For this system, we focused solely on those source fields which were automatically entered by FamilySearch's system. In future work, we may be able to develop a method of parsing this user-entered data in a way that opens up a much larger breadth of sources for the system to recommend. There may be some aspects of this user-entered data that would be more easily parsable than others. For example, the field may contain a Family History Library Book or Film Number, an ISBN, or a URL. These are unique and identifiable resources which could be parsed and included in the recommender. Various steps may need to be taken to make sure that such resources are generalizable beyond just the individual.

One simple and potentially beneficial modification would be to allow the user to configure the system to return similar individuals instead of sources. The system already computes similar individuals in the process of creating the source recommendations. In order to do this, the system would simply end at the point where it has collected the top individuals, rather than then gathering the scores for the sources attached to those individuals. As we explained before, it might also be beneficial in this case to remove the requirement that similar individuals have attached, parsable sources. This approach defines the individual the user is searching for in terms of those individuals closest to him or her. This allows the user to examine the surrounding individuals, potentially discovering sources that would not have been automatically parsable by the system, which could be a very valuable addition.

One other adjustment that might provide some benefit would be to gather data on the likelihood of a source being a good source. This would probably best be done by letting users tell the system whether or not a source actually contains their ancestor, although it could also be gathered using a method similar to our validation. As this data is gathered, the system could use it to generate an overall confidence that a particular source contains information on the individual in question. For now, the ranking system shows the relative value of sources, but no real indicator of how confident we are that the individual is contained in that source.

Bibliography

Adomavicius, G. & Tuzhilin, A. (2005). Toward the Next Generation of Recommender Systems: a Survey of the State of the Art and Possible Extensions. *IEEE Transactions on Knowledge and Data Engineering*, *17*(6), 734–749.

Almazro, D., Shahatah, G., Albdulkarim, L., Kherees, M., Martinez, R., & Nzoukou, W. (2010). A Survey Paper on Recommender Systems. *arXiv preprint arXiv:1006.5278.*

Ancestry.com. (2015). Ancestry.com LLC Reports Second Quarter 2015 Financial Results. Accessed October 2015. Retrieved from http://ir.ancestry.com/releasedetail.cfm?releaseid=923333

BCG. (2000). *The BCG Genealogical Standards Manual.* Orem, Utah: Ancestry Publishing.

Brinton, D. (2012). *The Feasibility of Implementing a Collaborative Recommender System for the Brigham Young University Harold B. Lee Library* (Honors Thesis, Brigham Young University).

Duff, W. M. & Johnson, C. A. (2003). Where Is the List with All the Names? Information-Seeking Behavior of Genealogists. *American Archivist*, *66*(1), 79–95.

Eastman, D. (2002). What's in the Future for Genealogy? *Ancestry Magazine*, *20*(6), 14–19.

Freeman, E. A. (1877). Pedigrees and Pedigree Makers. *The Living Age ... 134*(19), 67–85.

Fulton, C. (2009). Quid Pro Quo: Information Sharing in Leisure Activities. *Library Trends*, *57*(4), 753–768.

Grabowski, J. (1992). Keepers, Users, and Funders: Building an Awareness of Archival Value. *The American Archivist*, *55*(3), 464–472.

Herlocker, J. L., Konstan, J. A., Terveen, L. G., & Riedl, J. T. (2004). Evaluating Collaborative Filtering Recommender Systems. *ACM Transactions on Information Systems*, *22*(1), 5–53.

Hershkovitz, A. (2011). Leveraging Genealogy as an Academic Discipline. *Avotaynu*, *27*(3), 18–23.

Jones, T. W. (2007). Post-secondary Study of Genealogy : Curriculum and Its Contexts. *Avotaynu: The International Review of Jewish Genealogy*, *23*, 17–23.

Kembellec, G., Chartron, G., & Saleh, I. (2014). *Recommender Systems*. Wiley.

Mills, E. S. (2003a). Genealogy in the ‘Information Age’: History’s New Frontier? *National Genealogical Society Quarterly*, *91*, 260–277.

Mills, E. S. (2003b, March). Roundabout Research: Pursuing Collateral Lines to Prove Parentage of a Direct Ancestor-Samuel Hansonof Frontier Georgia. *National Genealogical Society Quarterly*, *91*, 19–30.

Ricci, F., Rokach, L., Shapira, B., & Kantor, P. B. (2015). *Recommender Systems Handbook*. Springer.

Sheppard, W. L. (1977). A Bicentennial Look at Genealogy Methods, Performance, Education, and Thinking. *National Genealogical Society Quarterly*, *65*(1), 3–15.

The Pew Internet & American Life Project. (2000). Tracking Online Life: How Women Use the Internet to Cultivate Relationships with Family and Friends. Accessed October 27, 2017. Retrieved from http://www.pewinternet.org/2000/05/10/main-report-29/

U.S. Census Bureau. (2014). *2014 U.S. Census*.

Veale, K. H. (2005). A Doctoral Study of the Use of Internet for Genealogy. *Historia Atual Online*, *7*, 7–14.

Weiss, J., Nolan, J., Hunsinger, J., & Trifonas, P. (2006). *The International Handbook of Virtual Learning Environments*. Springer.

Further Reading

Ancestry.com. (2005). Press Release: Americans' Fascination with Family History is Rapidly Growing. Accessed October 2015. Retrieved from http://www.ancestry.com/corporate/newsroom/press-releases/2005/06/ameri-cans-fascination-with-family-history-is-rapidly-growing

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning.* Springer.

Bishop, R. (2005). The Essential Force of the Clan: Developing a Collecting-inspired Ideology of Genealogy through Textual Analysis. *Journal of Popular Culture*, *38*(6), 990–1010.

Burges, C. J. C. (1998). A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery*, *2*(2), 121–167.

Burke, R. (2002). Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction*, *12*(4), 331–370.

Castellano, G., Jain, L. C., Maria, A., & Eds, F. (2009). *Web Personalization in Intelligent Environments* (G. Castellano, L. C. Jain, & A. M. Fanelli, Editors). Studies in Computational Intelligence. Berlin, Heidelberg: Springer Berlin Heidelberg.

Croom, E. A. (2000). *The Sleuth Book for Genealogists: Strategies for More Successful Family History Research.* Cincinnati, Ohio: Betterway Books.

Drake, P. J. (2001). Findings from the Fullerton Genealogy Study. *Posted Material from Master's Thesis. Fullerton: California State University. 25*, 2009.

Erben, M. (1991). Genealogy and Sociology: A Preliminary Set of Statements and Speculations. *Sociology*, *25*(2), 275–292.

Greenberg, N. S. (1982). *Gerontology and Genealogy: Family Research as a Therapeutic Tool for the Aged.* ERIC Clearinghouse.

Hackstaff, K. B. (2010). Family Genealogy: A Sociological Imagination Reveals Intersectional Relations. *Sociology Compass*, *4*(8), 658–672.

Hill, W., Stead, L., Rosenstein, M., & Furnas, G. (1995). Recommending and Evaluating Choices in a Virtual Community of Use. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Pages 194–201).

Kramer, A.-M. C. (2011). Kinship, Affinity and Connectedness : Exploring the Role of Genealogy in Personal Lives. *Sociology*, *45*(3), 379–395.

Lü, L., Medo, M., Yeung, C. H., Zhang, Y.-C., Zhang, Z.-K., & Zhou, T. (2012). Recommender systems. *Physics Reports*, *519*(1), 1–49.

Lucas, S. A. (2008). *The Information Seeking Process of Genealogists* (Doctoral Dissertation, Emporia State University).

McCain, G. & Ray, N. M. (2003). Legacy Tourism: The Search for Personal Meaning in Heritage Travel. *Tourism Management*, *24*(6), 713–717.

Mills, E. S. (1997). *Evidence!: Citation and Analysis for the Family Historian*. Baltimore, Maryland: Genealogical Publishing Company.

Montaner, M., López, B., & De La Rosa, J. L. (2003). A Taxonomy of Recommender Agents on the Internet. *Artificial Intelligence Review*, *19*(4), 285–330.

Nash, C. (2005). Geographies of Relatedness. *Transactions of the Institute of British Geographers*, *30*(4), 449–462.

Rapp, D. W. & Jones, M. P. (2012). Analyzing the Family Tree. In *Proceedings of RootsTech Technology Workshop, Salt Lake City, UT.*

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994). GroupLens : An Open Architecture for Collaborative Filtering of Netnews. In *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work* (Pages 175–186).

Rojas, J. (2011). Assessment of a Proprietary Online Smart-Family-Matching Tool to Reunite Lost Families. In *Proceedings of AFRICON, 2011* (September, Pages 1–6). IEEE.

Shardanand, U. & Maes, P. (1995). Social Information Filtering. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Pages 210–217).

Shriver, M. D. & Kittles, R. A. (2004). Genetic Ancestry and the Search for Personalized Genetic Histories. *Nature Reviews. Genetics*, *5*(8), 611–618.

The Free Library. (2000). Recent Maritz Poll Shows Explosion in Popularity of Genealogy. Accessed February 2015. (While this website appears to be no longer accessible, the statistic we reference from the site is also referenced by Weiss et al. 2006, Ancestry.com 2005, and Drake 2001.) Retrieved from http://www.genealogy.com/genealogy/press-051600.html

Tutton, R. (2004). They Want to Know Where they Came From: Population Genetics, Identity, and Family Genealogy. *New Genetics and Society*, *23*(1), 105–120.

Tyler, K. (2008). Ethnographic Approaches to Race, Genetics and Genealogy. *Sociology Compass*, *2*(6), 1860–1877.

Walker, T. & Embley, D. W. (2004). Automatic Location and Separation of Records : A Case Study in the Genealogical Domain. In *Proceedings of ER: International Conference on Conceptual Modeling—Conceptual Modeling for Advanced Application Domains* (Pages 302–313). Springer.

Willever-farr, H., Zach, L., & Forte, A. (2012). Tell Me About My Family : A Study of Cooperative Research on Ancestry.com. In *Proceedings of the 2012 iConference* (Pages 303–310). ACM.

Wilson, D. R. (2002). Bidirectional Source Linking: Doing Genealogy 'Once' and 'For All'. In *Proceedings of Family History and Technology Workshop* (Pages 54–60).

Appendix

Lies in Genealogy

The presence of falsified genealogical information exacerbates the already difficult challenge to find good sources in family history research.

> I turn over a peerage or other book of genealogy, and I find that, when a pedigree professes to be traced back to the times of which I know most in detail, it is all but invariably false. As a rule, it is not only false, but impossible...The historical circumstances, when any are introduced, are for the most part not merely fictions, but exactly that kind of fiction which is, in its beginning, deliberate and interested falsehood. (Freeman, 1877, p. 67)

During some time periods, such falsification of genealogical records was not merely present, but widespread. Edward Freeman describes a specific example of one such time period.

> The time of the Norman Conquest is the time to which it became fashionable for people to trace up their pedigrees. To be of the blood of the invaders of England was thought to be something creditable. Some people undoubtedly came of such blood, and could prove that they came of it. And of course there must have been many others who did come of it who could not in the same way prove the fact. It thus became a point of honor with most families to think themselves descended from the companions of the Norman Conqueror. Those who had no real pedigrees to prove it invented false pedigrees, which in a few generations did just as well. (Freeman, 1877, p. 69)

So, even if sources can be found dating back to the time period, perhaps even proven to have been written by the person in question, they may still be wrong. Some select historians may know a time period well enough to recognize inconsistencies in such documents, but the hobbyist cannot be expected to navigate such complexities.

Potential Benefits to Family History

Right now, it is hard to tell if recommender systems are being used in some of the major genealogy software packages. We cannot find any indication of such usage in publicly-disclosed documents. Nevertheless, many of these software packages have automatic suggestions or tips. From the outside, they appear to be generated based on content and not collaborative filtering. Thus, given that family history documents are often littered with alternate misspellings and typos, automatically parsing the content may not turn up all the relevant materials. It is feasible that a collaborative recommender system might overcome this issue and be able to point to relevant materials, even when the supporting content that makes it relevant is not automatically searchable because of typos or alternate spellings.

Source Quality

In the genealogical community, there is a general understanding that certain sources are more reputable than others. For example, in verifying a person's birthdate, a birth certificate is a higher-quality source than a newspaper article. Thus, some sources are more useful than others. The most readily-available sources for our system are those that are already digitized and indexed by FamilySearch, since those are the sources whose source fields are easily parsable. Sources entered by hand may, in fact, be high-quality sources for certain individuals and pieces of information, but free-entry text is not guaranteed to follow any sort of standardized form. Consequently, writing a parser for such free-entry source text would be a very difficult task, although if implemented successfully, it could yield marked improvement for the recommender system.

Date Regex

```
(\W|_)*((\W*((((3[01])|(2\d)|(1\d)|(0?\d))((st)|(nd)|(rd)|(th))?)(\W|_)*((jan|feb|mar|apr|may|jun|jul|aug|sep|oct|nov|dec)[urceytoi]?[auhlsebm
]?[ratmeb]?[yrbe]?[yer]?[r]?|([\W_]((1[012])|(0?\d)))[\W_])(\W|_)*(\d{2,4}))\W*|\W*(\d{4})\W*|\W*(jan|feb|mar|apr|may|jun|jul|aug|sep|oct|
nov|dec)[urceytoi]?[auhlsebm]?[ratmeb]?[yrbe]?[yer]?[r]?(\W|_)*(\d{4})\W*|\W*((((3[01])|(2\d)|(1\d)|(0?\d))((st)|(nd)|(rd)|(th))?)(\W|_)*(jan|
feb|mar|apr|may|jun|jul|aug|sep|oct|nov|dec)[urceytoi]?[auhlsebm]?[ratmeb]?[yrbe]?[yer]?[r]?)\W*|\W*((((3[01])|(2\d)|(1\d)|(0?\d))((st)|(nd)|(rd
)|(th))?)(\W|_)+((1[012])|(0?\d))(\W|_)+(\d{2,4}))\W*|\W*(((1[012])|(0?\d))(\W|_)+(((3[01])|(2\d)|(1\d)|(0?\d))((st)|(nd)|(rd)|(th))?)(\W|_)
+(\d{2,4}))\W*|\W*((jan|feb|mar|apr|may|jun|jul|aug|sep|oct|nov|dec)[urceytoi]?[auhlsebm]?[ratmeb]?[yrbe]?[yer]?[r]?)\W*|\W*((jan|feb|mar|
apr|may|jun|jul|aug|sep|oct|nov|dec)[urceytoi]?[auhlsebm]?[ratmeb]?[yrbe]?[yer]?[r]?(\W|_)*(((3[01])|(2\d)|(1\d)|(0?\d))((st)|(nd)|(rd)|(th))?))\
W*|\W*((jan|feb|mar|apr|may|jun|jul|aug|sep|oct|nov|dec)[urceytoi]?[auhlsebm]?[ratmeb]?[yrbe]?[yer]?[r]?(\W|_)*(((3[01])|(2\d)|(1\d)|(0?\d))((
st)|(nd)|(rd)|(th)|(\W|_)+))(\W|_)*(\d{4}))\W*)|(from(\W|_)*(\W*((((3[01])|(2\d)|(1\d)|(0?\d))((st)|(nd)|(rd)|(th))?)(\W|_)*((jan|feb|mar|
apr|may|jun|jul|aug|sep|oct|nov|dec)[urceytoi]?[auhlsebm]?[ratmeb]?[yrbe]?[yer]?[r]?|([\W_]((1[012])|(0?\d)))[\W_])(\W|_)*(\d{2,4}))\W*|\W*(\
d{4})\W*|\W*(jan|feb|mar|apr|may|jun|jul|aug|sep|oct|nov|dec)[urceytoi]?[auhlsebm]?[ratmeb]?[yrbe]?[yer]?[r]?(\W|_)*(\d{4})\W*|\W
*((((3[01])|(2\d)|(1\d)|(0?\d))((st)|(nd)|(rd)|(th))?)(\W|_)*(jan|feb|mar|apr|may|jun|jul|aug|sep|oct|nov|dec)[urceytoi]?[auhlsebm]?[ratmeb]?[
yrbe]?[yer]?[r]?)\W*|\W*((((3[01])|(2\d)|(1\d)|(0?\d))((st)|(nd)|(rd)|(th))?)(\W|_)+((1[012])|(0?\d))(\W|_)+(\d{2,4}))\W*|\W*(((1[012])|(0?\
d))(\W|_)+(((3[01])|(2\d)|(1\d)|(0?\d))((st)|(nd)|(rd)|(th))?)(\W|_)+(\d{2,4}))\W*|\W*((jan|feb|mar|apr|may|jun|jul|aug|sep|oct|nov|dec)[
urceytoi]?[auhlsebm]?[ratmeb]?[yrbe]?[yer]?[r]?)\W*|\W*((jan|feb|mar|apr|may|jun|jul|aug|sep|oct|nov|dec)[urceytoi]?[auhlsebm]?[ratmeb]?[yrbe
]?[yer]?[r]?(\W|_)*(((3[01])|(2\d)|(1\d)|(0?\d))((st)|(nd)|(rd)|(th))?))\W*|\W*((jan|feb|mar|apr|may|jun|jul|aug|sep|oct|nov|dec)[urceytoi]?[
auhlsebm]?[ratmeb]?[yrbe]?[yer]?[r]?(\W|_)*(((3[01])|(2\d)|(1\d)|(0?\d))((st)|(nd)|(rd)|(th)|(\W|_)+))(\W|_)*(\d{4}))\W*)(\W|_)*to(\W|_)*(\
W*((((3[01])|(2\d)|(1\d)|(0?\d))((st)|(nd)|(rd)|(th))?)(\W|_)*((jan|feb|mar|apr|may|jun|jul|aug|sep|oct|nov|dec)[urceytoi]?[auhlsebm]?[ratmeb]?[
yrbe]?[yer]?[r]?|([\W_]((1[012])|(0?\d)))[\W_])(\W|_)*(\d{2,4}))\W*|\W*(\d{4})\W*|\W*(jan|feb|mar|apr|may|jun|jul|aug|sep|oct|nov|dec)[
urceytoi]?[auhlsebm]?[ratmeb]?[yrbe]?[yer]?[r]?(\W|_)*(\d{4})\W*|\W*((((3[01])|(2\d)|(1\d)|(0?\d))((st)|(nd)|(rd)|(th))?)(\W|_)*(jan|feb|mar|
apr|may|jun|jul|aug|sep|oct|nov|dec)[urceytoi]?[auhlsebm]?[ratmeb]?[yrbe]?[yer]?[r]?)\W*|\W*((((3[01])|(2\d)|(1\d)|(0?\d))((st)|(nd)|(rd)|(th))?)
(\W|_)+((1[012])|(0?\d))(\W|_)+(\d{2,4}))\W*|\W*(((1[012])|(0?\d))(\W|_)+(((3[01])|(2\d)|(1\d)|(0?\d))((st)|(nd)|(rd)|(th))?)(\W|_)+(\d
{2,4}))\W*|\W*((jan|feb|mar|apr|may|jun|jul|aug|sep|oct|nov|dec)[urceytoi]?[auhlsebm]?[ratmeb]?[yrbe]?[yer]?[r]?)\W*|\W*((jan|feb|mar|apr|
may|jun|jul|aug|sep|oct|nov|dec)[urceytoi]?[auhlsebm]?[ratmeb]?[yrbe]?[yer]?[r]?(\W|_)*(((3[01])|(2\d)|(1\d)|(0?\d))((st)|(nd)|(rd)|(th))?))\W*|\
W*((jan|feb|mar|apr|may|jun|jul|aug|sep|oct|nov|dec)[urceytoi]?[auhlsebm]?[ratmeb]?[yrbe]?[yer]?[r]?(\W|_)*(((3[01])|(2\d)|(1\d)|(0?\d))((st)|(
nd)|(rd)|(th)|(\W|_)+))(\W|_)*(\d{4}))\W*))|((\W*((((3[01])|(2\d)|(1\d)|(0?\d))((st)|(nd)|(rd)|(th))?)(\W|_)*((jan|feb|mar|apr|may|jun|jul|
aug|sep|oct|nov|dec)[urceytoi]?[auhlsebm]?[ratmeb]?[yrbe]?[yer]?[r]?|([\W_]((1[012])|(0?\d)))[\W_])(\W|_)*(\d{2,4}))\W*|\W*(\d{4})\W*|\W
*(jan|feb|mar|apr|may|jun|jul|aug|sep|oct|nov|dec)[urceytoi]?[auhlsebm]?[ratmeb]?[yrbe]?[yer]?[r]?(\W|_)*(\d{4})\W*|\W*((((3[01])|(2\d)|(1\d)
|(0?\d))((st)|(nd)|(rd)|(th))?)(\W|_)*(jan|feb|mar|apr|may|jun|jul|aug|sep|oct|nov|dec)[urceytoi]?[auhlsebm]?[ratmeb]?[yrbe]?[yer]?[r]?)\W*|\W
*((((3[01])|(2\d)|(1\d)|(0?\d))((st)|(nd)|(rd)|(th))?)(\W|_)+((1[012])|(0?\d))(\W|_)+(\d{2,4}))\W*|\W*(((1[012])|(0?\d))(\W|_)+(((3[01])|(2\
d)|(1\d)|(0?\d))((st)|(nd)|(rd)|(th))?)(\W|_)+(\d{2,4}))\W*|\W*((jan|feb|mar|apr|may|jun|jul|aug|sep|oct|nov|dec)[urceytoi]?[auhlsebm]?[
ratmeb]?[yrbe]?[yer]?[r]?)\W*|\W*((jan|feb|mar|apr|may|jun|jul|aug|sep|oct|nov|dec)[urceytoi]?[auhlsebm]?[ratmeb]?[yrbe]?[yer]?[r]?(\W|_)
*(((3[01])|(2\d)|(1\d)|(0?\d))((st)|(nd)|(rd)|(th))?))\W*|\W*((jan|feb|mar|apr|may|jun|jul|aug|sep|oct|nov|dec)[urceytoi]?[auhlsebm]?[ratmeb]?[
yrbe]?[yer]?[r]?(\W|_)*(((3[01])|(2\d)|(1\d)|(0?\d))((st)|(nd)|(rd)|(th)|(\W|_)+))(\W|_)*(\d{4}))\W*)(\W|_)*(\/|(through)|(thru))(\W|_)*(\W
*((((3[01])|(2\d)|(1\d)|(0?\d))((st)|(nd)|(rd)|(th))?)(\W|_)*((jan|feb|mar|apr|may|jun|jul|aug|sep|oct|nov|dec)[urceytoi]?[auhlsebm]?[ratmeb]?[
yrbe]?[yer]?[r]?|([\W_]((1[012])|(0?\d)))[\W_])(\W|_)*(\d{2,4}))\W*|\W*(\d{4})\W*|\W*(jan|feb|mar|apr|may|jun|jul|aug|sep|oct|nov|dec)[
urceytoi]?[auhlsebm]?[ratmeb]?[yrbe]?[yer]?[r]?(\W|_)*(\d{4})\W*|\W*((((3[01])|(2\d)|(1\d)|(0?\d))((st)|(nd)|(rd)|(th))?)(\W|_)*(jan|feb|mar|
apr|may|jun|jul|aug|sep|oct|nov|dec)[urceytoi]?[auhlsebm]?[ratmeb]?[yrbe]?[yer]?[r]?)\W*|\W*((((3[01])|(2\d)|(1\d)|(0?\d))((st)|(nd)|(rd)|(th))?)
(\W|_)+((1[012])|(0?\d))(\W|_)+(\d{2,4}))\W*|\W*(((1[012])|(0?\d))(\W|_)+(((3[01])|(2\d)|(1\d)|(0?\d))((st)|(nd)|(rd)|(th))?)(\W|_)+(\d
{2,4}))\W*|\W*((jan|feb|mar|apr|may|jun|jul|aug|sep|oct|nov|dec)[urceytoi]?[auhlsebm]?[ratmeb]?[yrbe]?[yer]?[r]?)\W*|\W*((jan|feb|mar|apr|
may|jun|jul|aug|sep|oct|nov|dec)[urceytoi]?[auhlsebm]?[ratmeb]?[yrbe]?[yer]?[r]?(\W|_)*(((3[01])|(2\d)|(1\d)|(0?\d))((st)|(nd)|(rd)|(th))?))\W*|\
W*((jan|feb|mar|apr|may|jun|jul|aug|sep|oct|nov|dec)[urceytoi]?[auhlsebm]?[ratmeb]?[yrbe]?[yer]?[r]?(\W|_)*(((3[01])|(2\d)|(1\d)|(0?\d))((st)|(
nd)|(rd)|(th)|(\W|_)+))(\W|_)*(\d{4}))\W*))|((about|abt\.?)(\W|_)*(\W*((((3[01])|(2\d)|(1\d)|(0?\d))((st)|(nd)|(rd)|(th))?)(\W|_)*((jan|feb|
mar|apr|may|jun|jul|aug|sep|oct|nov|dec)[urceytoi]?[auhlsebm]?[ratmeb]?[yrbe]?[yer]?[r]?|([\W_]((1[012])|(0?\d)))[\W_])(\W|_)*(\d{2,4}))\W*|\
W*(\d{4})\W*|\W*(jan|feb|mar|apr|may|jun|jul|aug|sep|oct|nov|dec)[urceytoi]?[auhlsebm]?[ratmeb]?[yrbe]?[yer]?[r]?(\W|_)*(\d{4})\W*|\W
*((((3[01])|(2\d)|(1\d)|(0?\d))((st)|(nd)|(rd)|(th))?)(\W|_)*(jan|feb|mar|apr|may|jun|jul|aug|sep|oct|nov|dec)[urceytoi]?[auhlsebm]?[ratmeb]?[
yrbe]?[yer]?[r]?)\W*|\W*((((3[01])|(2\d)|(1\d)|(0?\d))((st)|(nd)|(rd)|(th))?)(\W|_)+((1[012])|(0?\d))(\W|_)+(\d{2,4}))\W*|\W*(((1[012])|(0?\
d))(\W|_)+(((3[01])|(2\d)|(1\d)|(0?\d))((st)|(nd)|(rd)|(th))?)(\W|_)+(\d{2,4}))\W*|\W*((jan|feb|mar|apr|may|jun|jul|aug|sep|oct|nov|dec)[
urceytoi]?[auhlsebm]?[ratmeb]?[yrbe]?[yer]?[r]?)\W*|\W*((jan|feb|mar|apr|may|jun|jul|aug|sep|oct|nov|dec)[urceytoi]?[auhlsebm]?[ratmeb]?[yrbe
]?[yer]?[r]?(\W|_)*(((3[01])|(2\d)|(1\d)|(0?\d))((st)|(nd)|(rd)|(th))?))\W*|\W*((jan|feb|mar|apr|may|jun|jul|aug|sep|oct|nov|dec)[urceytoi]?[
auhlsebm]?[ratmeb]?[yrbe]?[yer]?[r]?(\W|_)*(((3[01])|(2\d)|(1\d)|(0?\d))((st)|(nd)|(rd)|(th)|(\W|_)+))(\W|_)*(\d{4}))\W*)))(\W|_)*
```