UHASSELT
KNOWLEDGE IN ACTION

Maastricht University

Faculteit Wetenschappen
School voor Informatietechnologie

master in de informatica

Masterthesis

Analysis of entity resolution techniques in academic and health data

Emile Peetermans

Scriptie ingediend tot het behalen van de graad van master in de informatica

PROMOTOR :

Prof. dr. Frank NEVEN

2021
2022

De transnationale Universiteit Limburg is een uniek samenwerkingsverband van twee universiteiten in twee landen: de Universiteit Hasselt en Maastricht University.

UHASSELT
KNOWLEDGE IN ACTION

www.uhasselt.be
Universiteit Hasselt
Campus Hasselt:
Martelarenlaan 42 | 3500 Hasselt
Campus Diepenbeek:
Agoralaan Gebouw D | 3590 Diepenbeek

Maastricht University

Faculteit Wetenschappen
School voor Informatietechnologie

master in de informatica

Masterthesis

Analysis of entity resolution techniques in academic and health data

Emile Peetermans

Scriptie ingediend tot het behalen van de graad van master in de informatica

PROMOTOR :

Prof. dr. Frank NEVEN

Universiteit Hasselt

Masterproef voorgedragen tot het behalen van de graad van master in de informatica

Analysis of Entity Resolution techniques in academic and health data

Auteur:

Emile Peetermans

Promotor:

Prof. dr. Frank Neven

Co-promotor:

Prof. dr. Stijn Vansummeren

Academiejaar 2021-2022

Acknowledgements

First and foremost, I'd like to thank my promoters, Prof. dr. Frank Neven and Prof. dr. Stijn Vansummeren. They have been very patient with me and have guided me with good ideas and remarks whenever I felt lost. Without the exceptional amount of trust they had in me, I would never have been able to finish this study.

I would also like to thank the Intego and FRIS teams, and especially Arne Janssens for providing the tools necessary to complete the second half of my thesis and always being available whenever I required assistance or had questions.

Lastly, I'm also very grateful for the support my parents provided.

Abstract

Many digital applications and platforms dealing with data use the concept of profiles. A profile combines data from one or multiple sources that relate to a single real-world entity. A company profile might include the name, address, type of business and the name of the CEO. There can be multiple profiles for a single real-world entity, however, and the activity of discovering these "duplicate" profiles is called Entity Resolution (ER). This umbrella term covers many different techniques and procedures, such as schema alignment, blocking, and creating matching functions. Different kinds of data require different methodologies; ER has no one-size-fits-all approach. In this study, two different types of problems that can arise due to duplicate profiles will be analysed and solved using modern ER techniques. One of these challenges is data deduplication, where a data source contains multiple profiles containing information for a single entity. The goal of ER, in this case, is to merge these duplicate profiles to remove redundant information and fix wrong or missing values. To explore and analyse valuable techniques for solving deduplication, a case study has been made on FRIS, a data set containing information about flemish research institutes. Many researchers and projects have duplicate profiles in this dataset, primarily due to wrong or missing unique identifiers. Given a set of profiles of researchers or projects, we developed an ER pipeline capable of extracting all pairs of profiles that are likely duplicates or matches. For each predicted matching pair, an explanation is given as to why this pair has been selected and with what level of confidence. This explanation is given in the form of a graphical tool that domain experts can use to accept or deny profile merges.

A second big challenge when dealing with data is record linkage. When two companies merge, for example, some of their clients can have an account in the database systems of both organisations. It could be valuable for the newly merged company to combine the data in the profiles relating to the same clients. In this case, one needs to "link" the records of one data source to entries in the other. This process is slightly different from deduplication, as two data sources might have different data schemes and contain distinct kinds of data. The analysis of this ER problem is performed on a dataset provided by Intego. This institute collects anonymous patient data from general practitioners all over Flanders. After a partially unsuccessful migration to a new database system, their data is divided between an old and a new set. Some information overlaps and occurs in old and new profiles, but most data is disjoint. A record linkage tool has been created that will look for the best match for all old patient profiles, and explain each decision using a graphical program based on decision rules.

However, none of the resulting entity resolution models will work perfectly for every possible duplicate pair. That's why, in both case studies, explainability and user feedback play an important role, as incorrect data merging can result in the loss of critical information.

Dutch summary

Veel digitale toepassingen en platforms die met gegevens werken, maken gebruik van het concept profielen. Een profiel combineert gegevens uit een of meer bronnen die betrekking hebben op één enkele reële entiteit. Een bedrijfsprofiel kan bijvoorbeeld de naam, het adres, het soort bedrijf en de naam van de CEO bevatten. Er kunnen echter meerdere profielen bestaan voor één enkele reële entiteit, en de methode om deze 'duplicate' profielen te ontdekken wordt Entity Resolution (ER) genoemd. Deze overkoepelende term omvat veel verschillende technieken en procedures, zoals schema alignment, blocking, en het maken van matching functions. Verschillende soorten data vereisen verschillende methodologieën; ER kent geen one-size-fits-all benadering. In deze studie zullen twee verschillende types van problemen die kunnen ontstaan door duplicate profielen worden geanalyseerd en opgelost met behulp van moderne ER technieken. Een van deze problemen is datadeduplicatie, waarbij een databron meerdere profielen bevat die informatie bevatten voor eenzelfde entiteit. Het doel van ER, in dit geval, is het samenvoegen van deze duplicate profielen om redundante informatie te verwijderen en foute of ontbrekende waarden te herstellen. Om waardevolle technieken voor het oplossen van deduplicatie te verkennen en te analyseren, is een case studie gemaakt van FRIS, een dataset met informatie over Vlaamse onderzoeksinstituten. Veel onderzoekers en projecten hebben duplicate profielen in deze dataset, voornamelijk als gevolg van verkeerde of ontbrekende unieke identifiers. Gegeven een set van profielen van onderzoekers of projecten, ontwikkelden we een ER pipeline die in staat is om alle paren van profielen te extraheren die waarschijnlijk duplicaten zijn. Voor elk voorspeld overeenstemmend paar wordt een verklaring gegeven waarom dit paar is geselecteerd en met welk betrouwbaarheidsniveau. Deze uitleg wordt gegeven in de vorm van een grafisch hulpmiddel dat domeinexperts kunnen gebruiken om samenvoegingen van profielen te aanvaarden of te weigeren.

Een tweede grote uitdaging bij het omgaan met gegevens is record linkage. Wanneer bijvoorbeeld twee bedrijven fuseren, kunnen sommige van hun klanten een account hebben in de databasesystemen van beide organisaties. Het kan voor het nieuwe gefuseerde bedrijf waardevol zijn om de gegevens in de profielen met betrekking tot dezelfde klanten te combineren. In dit geval moet men de records van de ene gegevensbron "linken" aan entries in de andere. Dit proces verschilt enigszins van deduplicatie, aangezien twee gegevensbronnen verschillende dataschema's kunnen hebben en verschillende soorten gegevens kunnen bevatten. De analyse van dit ER-probleem wordt uitgevoerd op een dataset die door Intego wordt verstrekt. Dit instituut verzamelt anonieme patiëntengegevens van huisartsen in heel Vlaanderen. Na een gedeeltelijk mislukte migratie naar een nieuw databasesysteem zijn hun gegevens verdeeld over een oude en een nieuwe set. Sommige informatie overlapt en komt voor in zowel oude als nieuwe profielen, maar de meeste gegevens zijn disjunct. Er is een record linkage tool gemaakt die voor alle oude patiëntprofielen de beste match zoekt, en elke beslissing toelicht met behulp van een grafisch programma op basis van decision rules.

Geen van de resulterende modellen voor het herkennen en samenvoegen van entiteiten zal echter perfect werken voor elk mogelijk duplicaatpaar. Daarom spelen in beide casestudies de uitlegbaarheid en de feedback van de gebruikers een belangrijke rol, aangezien een onjuiste gegevenssamenvoeging kan leiden tot het verlies van waardevolle informatie.

In hoofdstuk 2 worden literatuur en eerdere werken over het onderwerp verkend en beschreven aan de hand van voorbeelden en figuren. Dit hoofdstuk behandelt de algemene workflow van ER toepassingen, alsook vergelijkingen van state-of-the-art technieken voor de behandeling van verschillende situaties in elke processtap. Hoofdstuk 3 omvat de materie over experimentele data deduplicatie, met een case studie over de FRIS dataset. Dit hoofdstuk is onderverdeeld in twee aandachtsgebieden, personen en projecten. Voor beide problemen worden de ER-stappen behandeld met een analyse van de beste technieken. Het laatste deel van het hoofdstuk bespreekt de ontwikkeling en het gebruik van een grafisch hulpmiddel om de beslissingen van het ER model uit te leggen en om domeinexperts te helpen bij het begrijpen en

verbeteren van het model. De focus van hoofdstuk 4 ligt op record linkage, in het bijzonder op de Intego dataset. Dit hoofdstuk beschrijft de gebruikte patientgegevens, hoe deze zijn verwerkt, de technische uitdagingen en uiteindelijk de totstandkoming van een uitlegprogramma en hoe dit in toekomstig werk kan worden verbeterd. In het laatste hoofdstuk wordt de dissertatie afgesloten, worden de bevindingen toegelicht en worden onze oplossingen voor het probleem van FRIS en Intego besproken.

De tool gemaakt voor FRIS is gebaseerd op SN-blokvorming met een logistisch regressie-matchingmodel voor personen en LSH-blokvorming op abstract gecombineerd met een logistisch regressiemodel voor projecten. De resultaten van deze pijplijnen worden worden getoond door gelijksoortige gevallen te groeperen, zoals matches van personen zonder publicaties of projectparen met verschillende titels. Duplicaten zijn gesorteerd in aflopende volgorde op basis van de waarschijnlijkheid die het model toekend. Voor elk van deze gevonden duplicaten wordt een verklaring gegeven in de vorm van een tabel met similariteitswaarden en hun belangrijkheid. De belangrijkheid van elke feature-waarde wordt berekend door de SHAP-score voor dit sample te nemen en deze te normaliseren aan de hand van de globale maximale en minimale SHAP-waarden. Dit geeft een idee van hoeveel invloed deze waarde heeft op de uiteindelijke beslissing van het logistische regressiemodel. Basis gegevens over de profielen in de duplicaatparen worden ook gegeven, waaronder een reeks publicatienamen en onderzoeksdisciplines in het voorbeeld van personen. Een handmatige beoordelaar kan door deze resultaten scrollen om de paren te labelen als duplicaat, niet-duplicaat of onbekend. Deze informatie wordt gebruikt om de trainingsgegevens te verbeteren, vergelijkbaar met active learning, om het model efficiënter te maken.

Voor gebruik in de context van Intego is een soortgelijke tool gemaakt als die welke voor de FRIS data is gemaakt. Het belangrijkste idee is nog steeds om dubbele patiëntenparen zo duidelijk mogelijk weer te geven en aan te geven waarom het model denkt dat het om een duplicaat gaat. Dit stelt de medewerkers van Intego in staat de resultaten te interpreteren en overeenkomende gevallen te identificeren die nog niet zijn gevonden met behulp van de strikte hiërarchische matching. Een extra uitdaging bij deze gegevens is dat de hoeveelheid gegevens te groot is om alle patiënten in één keer te verwerken. Het instrument moet in staat zijn een voortdurend groeiende pool van potentiële matches te verwerken en de hertraining van matchingmodellen te verwerken terwijl het algoritme nog loopt. Een andere complicatie is het feit dat de relaties tussen profielen moeilijker te interpreteren zijn. Een 10%-overeenkomst in termen van diagnosecodes is minder betekenisvol dan een familienaam die op één teken na overeenkomt. Voor de eindgebruiker is het dus van nog groter belang om het belang van kenmerken zo te visualiseren dat de relaties tussen twee profielen duidelijk worden. Dit project op Intego maakt geen gebruik van een blackbox-model zoals de logistische regressie die voor FRIS-gegevens wordt gebruikt, zodat het visualiseren van voorspellingen eenvoudiger wordt. Voor elk overeenstemmend paar geeft het instrument een verklaring van het gevolgde beslissingsboompad met drempelwaarden bij elk beslissingsknooppunt. Voorbeeldwaarden in het geval van overlappende lijsten worden getoond door codes om te zetten in beschrijvingen en een maat van zeldzaamheid te tonen, die weergeeft hoe vaak zo'n match voorkomt.

Voor elke voorspelling wordt een tabel gegenereerd op basis van het beslissingspad van het getrainde boommodel. Op elk knooppunt dat voor een bepaalde beslissing wordt doorgegeven, wordt de voorwaarde van het knooppunt (b.v. "Bloeddrukverschil" < 2 kg) getoond met een kleur die het belang aangeeft om als overeenkomst te worden beschouwd, evenals de werkelijke waarde en voorbeelden in het geval van een op overlap gebaseerd similariteitskenmerk. De effectkleur wordt bepaald door vergelijking van het aantal niet-overeenkomende en overeenkomende paren in de trainingsgegevens die dezelfde uitkomst op de voorwaarde hebben als het huidige voorbeeldpaar. Beschouw een voorbeeldpaar met een zeer hoge overlap in diagnosecodes met data, het overlappingskenmerk "ICD10-date" is 0,7. De node in de uitleg tabel met de conditie "ICD10-date > 0.05" zal bijvoorbeeld erg groen zijn, omdat het aantal overeenkomende paren in de training data die ook aan deze conditie voldoen ver uitstijgen boven het aantal niet-overeenkomende paren die dat wel doen. In dit voorbeeld betekent dit dat deze voorwaarde zeer belangrijk wordt geacht voor de beslissing dat het specifieke samplepaar als een match zal worden geclassificeerd. Evenzo zullen voorwaarden waaraan ook meestal wordt voldaan door niet-matchende paren, ertoe leiden dat het conditieknooppunt rood wordt gekleurd. Bij een gelijke verdeling in matches en non-matches wordt de node grijs gekleurd om aan te geven dat het voldoen aan de voorwaarde relatief onbelangrijk is voor de uiteindelijke beslissing.

Een van de belangrijkste beperkende factoren voor ER-prestaties is de kwaliteit en de hoeveelheid van de beschikbare gegevens. In de twee gevallen die in deze studie zijn geanalyseerd, was een set gelabelde samples beschikbaar om supervised modellen te trainen. Zonder deze trainingsgegevens zouden andere benaderingen moeten worden overwogen, en zou de focus worden verlegd naar clustering en actief leren.

Vergelijken hoe goed een op clustering gebaseerde techniek zou presteren op onze gegevens, in tegenstelling tot de huidige supervised modellen, zou een boeiende studie zijn in toekomstig werk. Niet alleen is er behoefte aan een voldoende grote pool van vooraf gematchte profielen, maar de gegevens moeten ook attributen bevatten die op de een of andere manier kunnen worden omgezet in nuttige similariteitskenmerken om duplicaatparen te onderscheiden van niet-duplicaatparen. Met FRIS-projecten, bijvoorbeeld, waren de behaalde prestaties lager dan met de persoonsdata. Dit was vooral te wijten aan het feit dat er minder waardevolle attributen waren om te gebruiken, aangezien zelfs projecten met een overeenstemmende samenvatting en titel toch een verschillend contractID konden hebben. Machine-learning hulpmiddelen waren in dit geval nuttig bij het identificeren van gelijkeniskenmerken die een kleine of zelfs nadelige invloed hadden op de efficiëntie van het model. Men zou bijvoorbeeld kunnen verwachten dat overlappende projecten geassocieerde onderzoekers delen of overlappende tijdspannes hebben. Beide kenmerken hebben een negatieve correlatie met de uitkomstvariabele, wat betekent dat projectparen met een verschillend contractID vaker een grote overeenkomst vertonen dan projecten met overeenstemmende ID's. Ook de tijdspanne waarin een onderzoeker actief heeft gepubliceerd, zou als nuttig kunnen worden beschouwd om de gelijkenis tussen twee personen te vergelijken. Dit bleek een misvatting te zijn omdat dubbele vermeldingen voor onderzoekers vaak worden veroorzaakt doordat mensen van het ene instituut naar het andere overstappen. Het ene profiel heeft dan een actieve periode tot aan de overstap, terwijl het andere profiel een publikatiebereik heeft dat na de overstap begint. Deze tegenstrijdigheden tussen verwachting en uitkomst komen vaak voor bij machine learning problemen, maar kunnen de besluitvorming foutgevoeliger maken wanneer er onvoldoende belangrijke features zijn.

Als antwoord op de belangrijkste onderzoeksvraag uit de inleiding wordt aangetoond dat het met moderne technieken mogelijk is duplicaten nauwkeurig te voorspellen. Er moeten echter inspanningen worden geleverd om het vertrouwen van de gebruikers in het systeem op te bouwen, willen deze modellen bruikbaar zijn. Deze uitdaging is aangegaan door hulpmiddelen te introduceren die in staat zijn het beslissingsproces bloot te leggen, een evenwicht te vinden tussen de mate van controle tussen mens en machine en, op grond daarvan, te voldoen aan de eis van betrouwbaarheid. Zelfs met beperkte data is het mogelijk een behulpzaam programma te ontwikkelen om de waarschijnlijkheid te voorspellen dat twee profielen naar dezelfde reële entiteit verwijzen. Tijdens het hele ontwikkelingsproces moet de nadruk liggen op transparantie en goed gefundeerde voorspellingen om het effect van menselijke veronderstellingen in de fasen van feature engineering en modelselectie te beperken.

Contents

Chapter 1

Introduction

Entity resolution is an umbrella term for techniques covering the process of comparing *profiles* of information within one or multiple datasets and identifying data that pertain to the same real-world entity. Use cases for these algorithms include removing duplicate entries, merging information from multiple datasets as well as different kinds of data mining.

This study will analyse and implement two cases of entity resolution. One is based on the Flanders Research Information Space (FRIS), an organisation that possesses data on all registered Flemish institutes, researchers and their publications. FRIS stores information about researcher names, addresses, research disciplines, and links to projects and institutes. Each researcher is assigned a globally unique ID to which all data about this person is linked. This so called "ORCID" is helpful as different institutes often publish their only data about a researcher that is already in the dataset, resulting in multiple entries with the same unique researcher ID. A complication is that some entries with different IDs relate to the same researcher, where the ID in one of the profiles has been entered wrongly either through human error or simply unavailability of information. This issue causes some researchers to lose data and could create unnecessary redundancy. The organisation also keeps track of information on research projects, such as funding institutes, publication abstracts and participants. Similar to researchers, projects have a globally unique ID. When multiple institutes participate in the same project, they often create separate entries, each referencing their researchers and sometimes slightly different abstract and research disciplines. Both entries should be linked through the project contract ID, but this ID is sometimes wrong or missing. This leads to a problem similar to the previous one with researchers, with some project entries being unreachable using the correct ID. In this case, the aim is to discover and merge duplicate profiles for both people and projects. To aid in this goal, a visual tool has been developed that automatically suggests possible duplicates, which an expert can individually approve.

The other case revolves around Intego, an academic centre for general practice. This institute has access to a large pool of information on patients made available to them by individual Flemish practices for conducting research. These data are entirely anonymised but include entries about diagnoses, lab results, measurements, prescriptions and vaccines. The organisation has since 2014 gradually moved from an old database system to a new one, but during the transfer of data, the mapping between old- and new patient IDs was lost. The data collection on the new database system started before a complete merge of the old system was completed. As a result, data are divided over two separate sets with their own data- and ID schemes. In this instance, the problem exists in merging two data sources' profiles while preserving common entities. Through the years, some profiles have been matched based on a strict overlap between profiles in important fields or social security numbers made available for a small portion of patients. This collection of pairs of duplicate old and new patient IDs is used for the training and verification of ER techniques. An adaptation of the tool created for FRIS has been made to accommodate new challenges introduced by the Intego dataset, such as data volume and increased uncertainty. Together, these two situations cover two essential cases of entity resolution.

In Chapter 2, literature and previous works on the subject matter will be explored and described using examples and figures. This chapter will cover the general workflow of ER applications, as well as comparisons of state-of-the-art techniques for handling different situations at each process step. Chapter 3 comprises the matter on experimental data deduplication, with a case study on the FRIS dataset. This

Chapter is divided into two main focus entities, persons and projects. For both problems, the ER steps are covered with an analysis of the best techniques. The final section of the chapter discusses the development and usage of a graphical tool for explaining the decisions of the ER model and aiding domain experts in understanding and improving the model. The focus of Chapter 4 lies on record linkage, particularly on the Intego dataset. This chapter describes the used patient data, how it was processed, technical challenges and ultimately, the creation of an explanation program and how this can be improved in future work. The final chapter concludes the thesis, explaining the findings and discussing our solutions to the problem posed by FRIS and Intego. This concluding chapter will also provide answers to the following primary research questions:

1. What techniques work best in the example cases of FRIS and Intego for:
    - Blocking
    - Matching
    - Explainability
2. Which similarity features achieve the best results for predicting duplicates on FRIS and Intego data?
3. Can a trustworthy ER application be created using machine learning techniques to accurately predict whether two profiles are duplicates based on the data provided?

The same basic process for building the ER application is followed in the record linkage and deduplication chapters. To train a model capable of discerning whether two sets of information share the same object, pairs of possible duplicate profiles need to be converted into vectors of *similarity features*. These values indicate the similarity of specific values or combinations of values in two profiles. For example, a similarity feature vector might include the result of an edit-distance metric comparing family names of persons. When many different metrics are used and the results are placed together in these vectors, they give a general numerical representation of similarity for a particular pair of profiles. These representations can then be used in typical machine learning models when supplemented with labels indicating the pair is a duplicate or a non-duplicate. In this study, many different kinds of similarity metrics will be explored and analysed in order to find the most potent combination of features for training an ER model. To this extent, any possible source or representation of data will be considered valuable until proven otherwise. This process, called "feature engineering" from this point, is one of the most crucial parts of this project. A sufficiently accurate entity resolution model might not be realised without a sufficiently large and influential set of similarity metrics.

A second critical stage is training and evaluating the model used for matching similarity feature vectors. This model ultimately decides whether a pair of profiles describe the same entity or not. This stage heavily relies on the features generated from the chosen similarity metrics but also depends on requirements such as the necessary interpretability and the most important performance metric. Both simple linear models and complex neural networks will be empirically analysed, resulting in the models that best meet the demands becoming an integral part of the end products.

Finally, the third crucial part of this study is making the decisions of produced deduplication and record-linkage models understandable, adaptable, and valuable when the probability of predictions is low. To this end, visualising tools will be created that graphically explain why the trained model comes to the conclusions it makes. Users then can correct the models' mistakes or accept correct classifications. This feedback will be used in the next iteration of training, making the model ever-evolving.

Chapter 2

Entity Resolution

This chapter will give formal definitions and thorough explanations for all ER concepts relevant to this study. The main component for all ER concepts is the *profile*, containing information on a real-world entity.

Many sources of data deal with structures called profiles. These profiles contain information for a particular real-world entity such as humans, companies or events. Suppose a grocery store uses a database containing information about the store's product types. A profile, in this case, could contain information about the name of the product, manufacturer, category, and description, for example: {`name: "Spaghetti 500 g", manufacturer: "Barilla", categories: ["pasta", "wheat products"], description: "500-gram pack of Barilla spaghetti"`}. Imagine this grocery store growing and merging with a smaller store. The company wants to merge their product database with the one used by the other store. In this process, duplicate products will need to be identified based on the profile information. The profiles in the other dataset could contain different information, such as an extra field containing nutritional information. On the other hand, data about the same product could be entered differently in both data sources because of mistakes or conventions. In addition, the structure of the data itself could be utterly divergent with alternative field names and data formats. The task of combining these databases and recognising shared products along with new ones is called Entity Resolution (ER). An illustrative example of profiles with possibly related entities is shown in Table 2.1. Here the profile IDs 8314, 8317 and 8319 seem to refer to the same person, likewise with 8315 and 8316. An ER algorithm should be capable of recognising this similarity and producing the clusters of "matching" profiles like in Table 2.2.

Many techniques have been designed over the years specifically for solving this crucial task in data management, ranging from rudimentary static algorithms to advanced deep neural network-based solutions. Some techniques work better for certain types of data or specific volume levels; some require training data while others do not, while some are more suited for continuously evolving data. There is, however, a general workflow that is shared by all ER techniques, consisting of the following basic steps: Schema alignment, Blocking, Matching and Clustering. The first and last steps of this workflow are not always required.

Schema alignment is necessary when the datasets to be merged have distinct schemes. The goal is to align attributes based on relatedness. Identifying these semantically identical fields creates a schema-aware context necessary for the following steps. *Blocking* is a technique used for reducing time complexity. In the most naive ER approach, any pair of profiles is considered to be a possible duplicate and will be processed in a costly comparison operation. In an effort to make ER more scalable, blocking techniques attempt to reduce the number of pairs to compare by finding potential matches based on an efficient function. The result is a set of "blocks" containing profiles to be compared. Blocking increases time efficiency at the cost of possibly finding fewer matches (reduced recall). The *matching* step involves comparing profiles within one block using a matching function to decide the degree of similarity. This similarity score can then be used to classify pairs of profiles as either match or non-match. *Clustering* is an optional step in which individual matching results are collected to infer indirect matches. The following sections will cover these steps in detail, including examples with state-of-the-art techniques.

Any pair of profiles from an entity is considered a "match" or "duplicate". Data sources containing

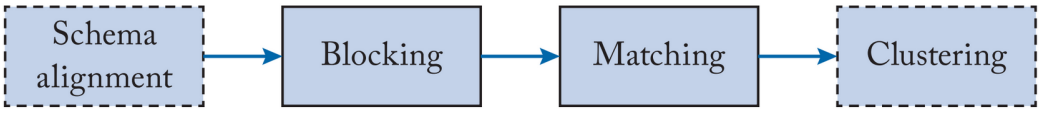

Figure 2.1: Schematic visualisation of the basic ER-workflow. Optional steps are indicated with dashed lines.

ID	Name	Address	Purchases
8314	Robert M. Jordan	22 Acacia avenue, London	8413, 43122, 1431
8315	Caroline Dubois	30 Rue Lecourbe, Paris	55, 134
8316	Carry S. Dubois		134, 14777
8317	Bob Jordan	82C S Croxted Rd, London	3331
8318	Jozef Smith	38 Rosendale Rd, London	1431
8319	Robert Martin Jordan	82C S Croxted Rd, London	43122, 1431, 3331

Table 2.1: Example of a set of user profiles containing profiles potentially referencing the same entities.

Blocks	
Robert	**Caroline**
8314	8315
8317	8316
8319	

Table 2.2: Sets of IDs of person profiles that most likely correspond to the entity "Robert Martin Jordan" or "Caroline Dubois" respectively.

multiple profiles per entity are called "dirty". The entity resolution task on a single dirty datasource is called *deduplication*. When there is at most one profile for each entity, the datasource is "clean". In the case of two or more clean data sources, entity resolution will focus on finding shared entities across different sets of data and attempting to merge profiles, called *record linkage*. While deduplication and record linkage have different use cases and slightly different workflows, the fundamentals of ER are still used in both.

2.1 Schema alignment

In general, this step in the default ER workflow is unnecessary when doing deduplication on a single datasource, or when the schemas of two datasets are entirely homogenous. Schema alignment aims to generate mappings between attributes of the different data sources. For example, one source might have the field "Surname", while another dataset includes a field called "Last name". These attributes both contain information about an entity's family name and and should be considered as describing the same information. Another example is the mapping between a combination of attributes. This example is visualised in Figure 2.2. Consider a field "Address" consists of information about a person's residence location. This information could be stored in multiple separate fields such as "Street" and "Number" in some other data sources.

In the above examples, the mapping is relatively straightforward. Automated techniques are necessary when schemata are very complex and difficult to compare by humans. An example is learning schema alignment rules using basic transformations, like abbreviations or acronyms. An example is TranScm, where a set of extendable built-in rules define a possible matching between two schema components and provide a means for translating an instance of the first schema to the second [MZ98]. More advanced techniques analyse attribute values to find appropriate mappings. Zhang et al. propose a two-step approach, consisting of first partitioning columns into clusters based on the data distribution of their values and then refining these partitions by computing the similarity of the distributions [Zha+11; Zha+10].

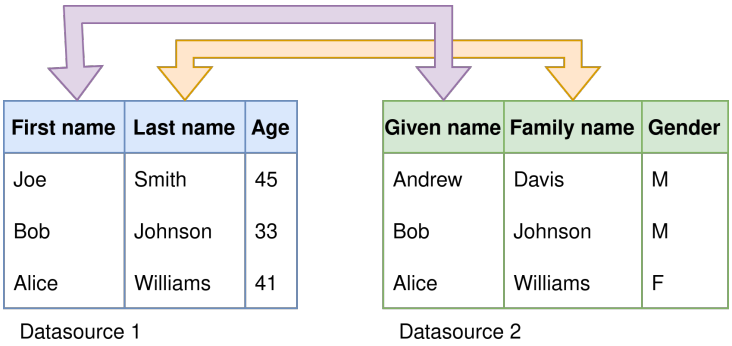

Figure 2.2: Simple example of aligning columns in two data sources. In this case alignment could be done manually, but also by comparing values between columns of both sources as multiple first- and last name values are equal to those in the given and family name columns.

2.2 Blocking

Blocking encapsulates the process of creating groups (blocks) of profiles that are likely matches. This step is necessary to reduce the otherwise quadratic time complexity of the matching algorithm. It is generally not possible to use a costly matching function on the entire Cartesian product of all profiles in the dataset, so a selection of blocks is usually made in order to improve time efficiency vastly. However, this comes at the cost of perhaps not finding all existing profiles for a particular entity due to inadequate blocking techniques. The goal is to find a suitable blocking scheme to minimise the loss of recall while reducing the matching set as much as possible, but this depends heavily on the data used. There might be no universal best technique, but this section will cover some of the most common schemes and explain their workings. Evaluating blocking techniques is done using the *pair-completeness* (P-C) and *pair-quality* (P-Q). These are defined as the fraction of true positive profile pairs assigned the same block compared to the total number of true positive pairs, and the fraction of pairs sharing a block that are true positive duplicates, respectively. Blocking techniques all work by selecting the most informative attributes based on domain knowledge or machine learning and then creating signatures or keys of these fields that are representative of the entire profile. These so called "blocking keys" can then be compared in an efficient manner. In the example given in Figure 2.3, SN achieves a P-C of 1, as all true duplicates share a block, and a P-Q of 0.5, because two in four profile combinations produced by the blocks are not duplicate profiles.

The most basic cornerstone non-learning method is called *Standard Blocking* (SB) [FS69]. A blocking key is manually defined through combinations of transformations of attributes. For example, in a database containing information about people, a blocking key could consist of the first three characters of the last name, the last number in the postal code and the first five characters in their telephone number. Profiles that produce the same blocking key using the same transformations are put in the same block. This technique is susceptible to noise, as the slightest difference in blocking keys results in two profiles never being compared. In order to address this issue, so-called q-gram blocking can be used [Chr11; Pap+15]. SB-keys are split up into parts of length q, where a separate block is made for each possible part, and profiles containing the same part in their keys are put together.

Blocking techniques based on supervised learning can be trained to yield complex sets of combinations of transformation functions. These algorithms try to optimise an objective function, such as maximising the ratio of previously uncovered positive pairs (matches) over the covered negative pairs. An example is ApproxDNF, which learns a set of k attribute transformation combinations in Disjunctive Normal Form (DNF) [BKM06].

Other blocking methods rely on global similarity evidence extracted from the blocking keys of profile pairs, the entire input DS, or even from the Matching step. One such example is Sorted Neighbourhood (SN), where blocking keys are put in an alphabetically sorted list, and profiles within a certain range in the list are put together in blocks [HS95; PWN06]. SN is more robust than SB, with a higher tolerance for noise when the range or window is set high enough. Canopy clustering attempts to find blocks by iteratively selecting a random profile, creating a new block for it and using a cheap string similarity measure to place all profiles with a similarity higher than a predefined threshold [Chr11; MNU00]. Profiles with a similarity higher than a second stricter threshold are removed from the selection list. They do not

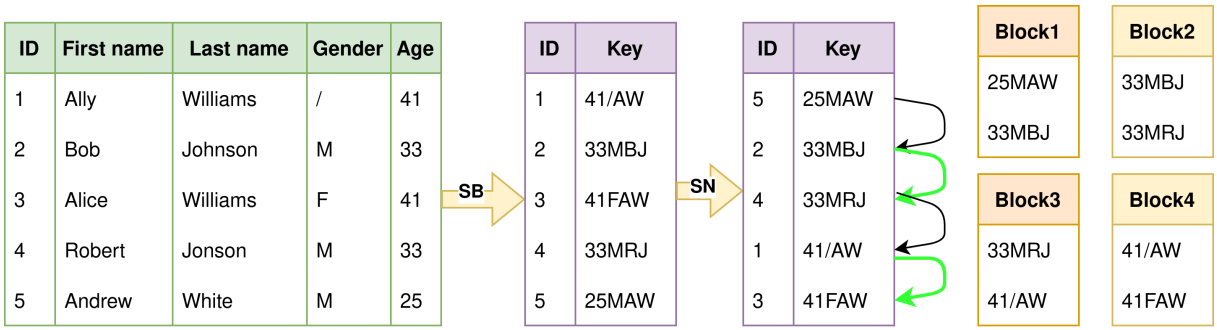

Figure 2.3: Example of standard blocking (SB) and sorted neighbourhood (SN) executed on standard blocks. SB keys are generated using age, gender, first character of first name and first character of last name. None of the keys overlap in this case so SB would yield zero blocks. When ordering the keys using SN with a window of one, four blocks are created of which two are most likely duplicate profiles.

participate in any of the subsequent blocks, as this would result in duplicate blocks.

Locality Sensitive Hashing (LSH), an algorithmic technique that hashes similar input items into the same "buckets" with high probability, is another technique incorporating global information often used by blocking schemes [RU11]. Given a similarity function $Sim : U \times U \rightarrow [0, 1]$, an LSH scheme with a set of hashing functions H has the probability $Prob_{h \in H}[h(A) = h(B)] = Sim(A, B)$ for two objects $A, B \in U$ [Cha02]. This characteristic allows for efficient similarity calculation without using a costly function. LSH for blocking works by first dividing every blocking key, for example, a person's first name, in a set of shingles or n-grams. The name "Karel" becomes {"kar", "are", "rel"} when $n = 3$. These small fixed-size parts of the string are used to create a vocabulary where each "shingle" represents an index. A string can thus be one-hot encoded into a vector with a length equal to the size of the vocabulary. The one hot-encoded vector is then hashed into a much smaller size while preserving similarities between pairs. These signatures are placed into b buckets in order to group possible duplicate pairs.

2.3 Matching

The goal of matching is to calculate the similarity between all pairs of profiles selected as candidate matches by the blocking step. This similarity score is based on a *matching function*. A similarity graph is created with edges identifying the probability that the connected profile nodes belong to the same entity. Matching schemes usually rely on the combination of similarity metrics calculated for each corresponding attribute pair. Numerical field similarity should generally be dependent on the distance between the two values. An example function comparing the numerical attribute A is the following: $NumSim(v1, v2) = 1.0 - \frac{|v1-v2|}{\max(A)-\min(A)}$. A metric often used to calculate the measure of similarity between sets of items is called the Jaccard similarity coefficient. This index defines the similarity between two sets as the intersection's size over the union's size: $J(A, B) = \frac{|A \cap B|}{|A \cup B|}$ [Jac12].

When comparing string attributes, however, there is no best way of defining similarity as it largely depends on the data. Selecting the best technique for comparing these values requires domain knowledge and extensive testing. Basic string similarity metrics are based on the edit-distance between two strings, signifying the number of manipulations required to transform one string to another. A well-known example is the Levenshtein distance, first described by Soviet mathematician Vladimir Levenshtein [Lev66]. Essentially, it represents the number of deletions, insertions and substitutions necessary to go from string a to string b. Edit-distances can precisely reveal the amount of difference between two strings on a character level but is also quite time-consuming on more extensive texts. On a content level, edit-distance metrics do not say much as two texts can be structured entirely different while still expressing the same concepts. Vector-based matching schemes, such as WHIRL, derive the similarity of two profiles by utilising numerical statistics on tokens used in field values [Coh00]. WHIRL calculates the cosine similarity between the TF-IDF vectors of specific string attributes. Every dimension in these vectors corresponds to a distinct word token t, with TF standing for its Term Frequency or the number of times it appears in the attribute value and IDF for its Inverse Document Frequency which is the number of profiles that contain t in the specified attribute value. The cosine similarity returns a measure between 1 (equal) and

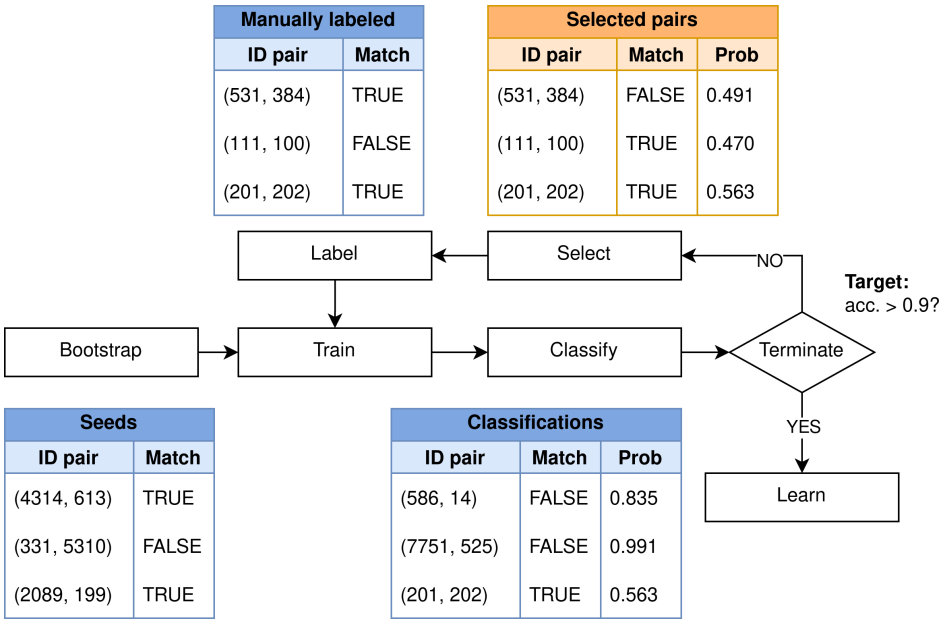

Figure 2.4: Schematic example of self learning entity resolution workflow. The process starts with a set of seeds in the bootstrap step, trains a supervised classification model on this small set and tests if classification results of model are accurate enough based on initial seeds/manual samples. If not, the program proceeds with the select step to retrieve ambiguous profile pairs and requires an expert to manually assign labels for the next training step.

0 (no similar tokens) while taking into account words that are frequent in most profiles, such as "the" or "in".

In order to predict whether a pair of profiles match, multiple similarity metrics on different attributes often have to be combined into a single decision function. This function can give either binary output (`match` or `non-match`) or return a number between 0 and 1 indicating the likelihood that the pair relates to one single entity. This decision function can be constructed manually by leveraging domain knowledge to specify thresholds for each attribute's similarity measure(s), or a prediction model can be created that learns to distinguish the two types of pairs from the characteristics of the data. Algorithms to achieve the latter can be divided into two classes, supervised and unsupervised methods.

Supervised techniques are often the most accurate, with the caveat of needing a large enough set of labelled duplicate and non-duplicate pairs. Every instance or pair of profiles is represented by a feature vector where every dimension corresponds to a similarity score for a specific attribute value. For example, threshold-based ordinal regression can be used on these labelled feature vectors to learn to map profile pairs into one of five categories: `hard-conflict`, `non-conflict`, `weak-match`, `moderate-match`, and `strong-match` [Yan+20]. In practice, any classification algorithm can be used for building a supervised matching method.

Due to the rarity of a sufficiently large amount of labelled data when dealing with entity resolution, many approaches have been designed to circumvent this requirement. One major class, in this case, are the unsupervised matching techniques, which are generally based on clustering, *self-learning* or graph theory. One of the earliest unsupervised matching techniques, "Clustering Record Linkage Model", is based on applying k-means clustering to the similarity feature vectors of candidate matching pairs [EVE02]. Using $k = 3$, the cluster with a centre closest to the point [1,1, ..., 1], which in this case is the point of complete similarity in all attributes, is considered the cluster of matches, while datapoints in the cluster closest to [0, 0, ..., 0] are considered non-matches. Pairs assigned to the third cluster are potential matches to be reviewed manually by a domain expert.

Self-learning methods attempt to create a *weakly* labelled dataset, using strict thresholds on similarity metrics for defining high-probability matches and non-matches. This automatically generated set of example instances are called *seeds* and can be used in supervised techniques. Hou et al. use self-learning iteratively in order to classify more complex samples in each round using an evolving classification model [Hou+20]. This is comparable to active learning approaches where the need for labelled data

is minimised through continuous classifying and retraining based on the initial seeds. Primpeli et al. introduce a way to generate seeds to start the active learning process without having to specify a set of arbitrary thresholds [PBK20]. They summarise the set of feature similarities to a single score and pick the "elbow" of the plotted distribution of these scores to divide matches and non-matches. After training on the available labelled samples and classifying the unlabelled pairs, a *termination criterion* is evaluated, such as converging to a performance target over the unlabelled instances as proposed by Meduri et al. [Vam+20]. If the criterion is not yet reached, the algorithm proceeds with the `select` step, in which the most ambiguous instances with the lowest predicted probability are selected for manual labeling. The algorithm then starts again, using the manually extended set of samples and repeats the training step.

The disadvantage of learning-based classification techniques is interpretability. Most of these algorithms produce black-box models whose predictions are difficult to analyse. Intelligibility is, however, a feature that is crucial to improving and maintaining ER solutions. Some classification schemes, such as decision trees, can be converted to sets of matching rules in order to make understanding model decisions easier to understand. Techniques like Explainable Boosting Machines attempt to build a classification model with interpretability in mind, by learning coefficients signifying the magnitude for each feature in a predication [Lou+13]. These models achieve results comparable to state-of-the-art techniques like XGBoost while being easy to understand [Nor+19]. By using Shapley Additive exPlanations (SHAP values), traditional blackbox models can also be made more explainable [LL17b]. SHAP can visualise individual contribution of each feature on the model output. For numerical output, an example is the force plot. A base value for the output is established by calculating the mean predicted value for each observation, while each feature is represented with their SHAP value "pushing" the output number to a higher or lower value according to their contribution.

2.4 Clustering

The output of the matching step can be seen as a similarity graph. A node is created for every profile where each pair of potentially matching profiles are connected by an edge depicting the similarity value or duplicate probability. A profile can have edges with multiple other profiles, indicating a possible entity consisting of three or more profiles. Clustering aims to convert this graph into a set of clusters or final entities. The most suitable clustering technique depends on the type of ER task. For record linkage, the graph is bipartite as there is a one-on-one mapping between the profiles in the two input data sources. An example of a clustering technique for these problems is called Unique Mapping Clustering, where edges are sorted in decreasing weight and iteratively. The profile pairs at the top of the list are considered duplicates if the similarity exceeds a threshold and none of the adjacent nodes has been matched yet [Lac+13]. In the case of deduplication, a popular choice is Center Clustering [HGI00]. This technique builds clusters around centre nodes, defined as the nodes with the highest average similarity score of connecting edges. Clustering algorithms are generally not required as entities can be found by simply setting a threshold value for the similarity score produced by the matching step and selecting all pairs that score above this threshold as duplicates. It becomes more valuable in cases where most entities consist of three profiles or more, as the algorithms can then use the transitive property of profile duplicates.

Chapter 3

FRIS: deduplication

The FRIS (Flanders Research Information Space) dataset contains information about researchers and projects in Flemish universities distributed by the Flemish government. Researchers are represented by name, a list of physical addresses, a list of electronic addresses (email), and in most cases, an ORCID ID. Open Researcher and Contributor ID (ORCID) is a persistent alphanumeric code to uniquely identify authors and contributors of scholarly communication. Apart from this, a person is linked to a set of publications, projects, organisations and research disciplines. When a researcher moves from one university to another, the new university will often create a new entry for the person with the university's address. This should not be a problem as the ORCID of the person should still be identical to the one in the previous entry, making any publications or projects that reference this record linkable to the other previous profile. This is not always the case, though, as some researchers do not yet have an ORCID before the university move, for example.

Another type of duplicates occurs in projects. These profiles consist of project title, abstract, a set of research disciplines, members, funding organisations and a start- and end date. Projects get a unique contractID by which they are identifiable. In some cases where a project receives funding from multiple sources, there will be duplicate records with a different contractID.

The ER task in this context is deduplication, as the execution domain includes a single dirty data source. This chapter will focus on finding the optimal technique for extracting the highest percentage of high-quality duplicates. The recall of ER techniques can be estimated using ORCID and contractIDs. However, the goal is to find duplicates that do not match in terms of these unique IDs, so a manual evaluation is necessary to measure precision accurately.

3.1 Persons

Person data were extracted in the format illustrated in Table 3.1. Research disciplines and publications include both a set of IDs and a list of strings, the latter of which was used to determine similarity in a more content-based manner. As there is only one data source, schema alignment is not relevant.

Attribute name	Value
first_name	Karel
last_name	Arnaut
university_city	Leuven
orcid	0000-0003-1872-9281
research_disciplines	'Anthropological theory', 'Linguistic anthropology', ...
research_discipline_codes	05040101, 05040108, ...
publications_title_keywords	"important analytic processes 'ethnicization' flemish cities ..."
publications_id_list	6fee21-423d-112, 344a4-5531f-431, ...

Table 3.1: Person attributes extracted from FRIS data.

Blocking scheme	P-C	P-Q
Exact first name	0.930	0.00171
Exact last name	0.963	0.01258
SN first name	0.943	0.00645
SN last name	*0.972*	0.00665
LSH last name	0.970	0.00412
LSH first + last name	0.961	0.01560
Union SN + LSH	0.972	0.00265
Intersection SN + LSH	0.970	*0.01880*
Union SN first + SN last	0.972	0.00334

Table 3.2: Results of different blocking schemes on Person data. P-C: pair completeness, percentage of duplicates found. P-Q: pair-quality, percentage of returned pairs that are duplicates.

3.1.1 Blocking

The fact that most duplicates are due to researchers moving to another university helps us choose an optimal blocking scheme. The attribute `university_city` will not be helpful as this will always be different, just like `fris_id`. A researcher will usually have different `research_disciplines` and `publications`, making fields related to these unusable for blocking as well. The most suitable attributes to block on are `first_name` and `last_name`. We will review some of the most common blocking techniques and evaluate them by measuring pair-completeness and pair-quality based on persons with the same ORCID.

Possibly the most naive technique is to put persons with equal names in the same block. This results in reasonably low Pair-completeness as some profiles contain spelling mistakes or omissions of second and third names. A more advanced approach is Sorted Neighbourhood, as illustrated in Section 2.2. In this study, a comparison is made between primarily sorting on first name or on last name. The window size is fixed at 4, as a bigger window does not immediately lead to a higher pair-completeness. However, some duplicates cannot be found by simply sorting in alphabetical order, as names might include a prefix in one of the two duplicate profiles. In this case, an order-independent technique such as Locality Sensitive Hashing (LSH) might improve results.

From the results of these blocking schemes shown in Table 3.2, it can be deduced that Sorted Neighbourhood (SN) on last name and then first name achieves the highest pair-completeness for person data. Pair-quality in this context is of little concern as the dataset size allows for any blocking scheme with a P-Q above 0.0001 still to achieve good time performance in the matching step. LSH can be adjusted by reducing the hashing resolution to reach a similar P-C as SN, but this brings with it a cost in P-Q and increases time complexity. A combined set of blocks was also tested to leverage the difference in results between the LSH and SN techniques. The Jaccard index between these two techniques is 0.14, indicating a fair difference between block composition. The results in Table 3.2 show however that this difference in blocking pairs only relates to false positives and that the correct pairs in LSH blocks are a subset of those in the SN results. Interestingly, pair-quality increases significantly when taking the intersection of these techniques.

3.1.2 Matching

As the output of the blocking procedure, a list of possible duplicate couples in the form of index couples is generated. This set consists of 19771 profile pairs, of which 541 are duplicates. The class of non-matching pairs is 35.5 times larger than the class of matches, so the data can be considered imbalanced. This imbalance between duplicates and non-duplicates is intrinsic to deduplication datasets because duplicates are usually a rare occurrence. The techniques used for matching thus need to be

In the matching step, each of these couples will be compared in order to decide whether the pair is a duplicate. For each one of these, a similarity feature vector will be produced for use in machine learning techniques. These similarity vectors consist of results from multiple similarity measures for different attributes.

Level 1	Level 2	Level 3	Level 4
Natural sciences	Physical sciences	Classical physics	Thermodynamics
Engineering and technology	Computer engineering	Computer theory	Systems theory and modelling
Medical and health sciences	Clinical sciences	Immunology	Vaccinology

Table 3.3: This table shows three examples of different levels in the flemish research discipline tree.

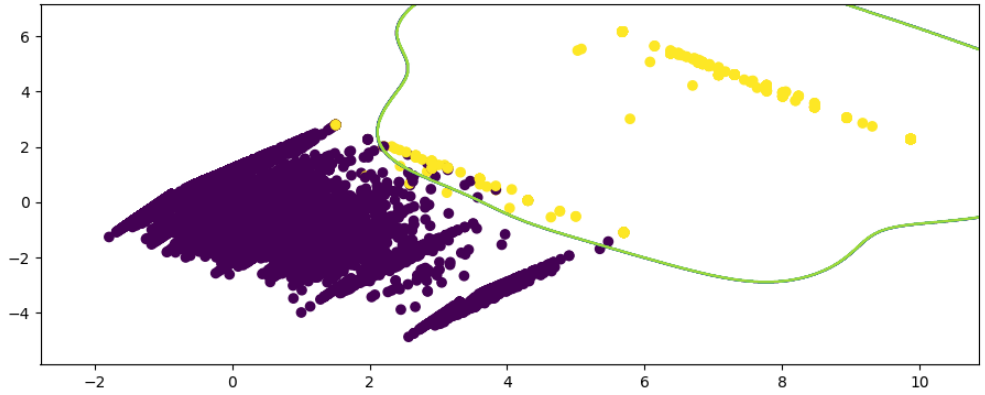

Figure 3.1: Scatterplot of first 2 PCA components for similarity vectors of blocking pairs. Non-matches are coloured in purple, and matches in are yellow. The green line represents an SVM trained for dividing these two classes.

Feature engineering

In the case of persons, a similarity vector includes the Levenshtein-distance for `first_name`, `last_name` and `city`. Flemish research disciplines are classified in a tree structure with four levels. An example of this structure can be observed in Table 3.3. The first two levels are meant for reporting. FWO mainly uses level 3 to decide on selection panel distribution of project applications. Level 4 provides a detailed typology of people, organisations, and projects [Uni]. Research disciplines in the FRIS dataset are all at level 3 or 4. This allows for comparisons of disciplines in lower tree levels for more generalisation. A Jaccard index feature was added for both level 2 and level 3 nodes for all entries in the `research_discipline_codes` fields. Publications ids are likewise compared using the Jaccard index between the sets of publications. Another feature that was tested are the dates for a persons publications. The minimum and maximum date for all publications connected to each profile were calculated in order to create a time-range in which the person

In order to calculate similarity based on string context instead of exact comparisons, vector embeddings for research disciplines and publications have been added. These are context-aware feature vector representations generated by a language model. This project uses a miniature version of the pre-trained BERT model [San+19]. Embeddings are generated for the list of research discipline names combined in a single string and for the titles of publications which are first converted to a list of keywords to increase performance. These vectors are compared between profiles using the cosine similarity measure $cos(x, y) = \frac{x \cdot y}{||x|| \cdot ||y||}$.

A visualisation was made to detect whether the features are descriptive enough to allow for separation by a linear model. In Figure 3.1 a version of the data reduced to 2 dimensions using PCA is shown along with a trained support vector dividing the two classes. This image makes it clear that there are two groups of positive matches; one is very distinct from non-matches, while the other group is much closer and more challenging to isolate. Through examination, it became evident that matching pairs belonging to the latter group usually have slightly differing first- or last names and publications are often not similar. The non-matches in close proximity to the bottom group usually have discipline and publication codes similar to their paired profile. The chosen model for classification will need to be able to learn a prioritisation for features that can handle these kinds of anomalies.

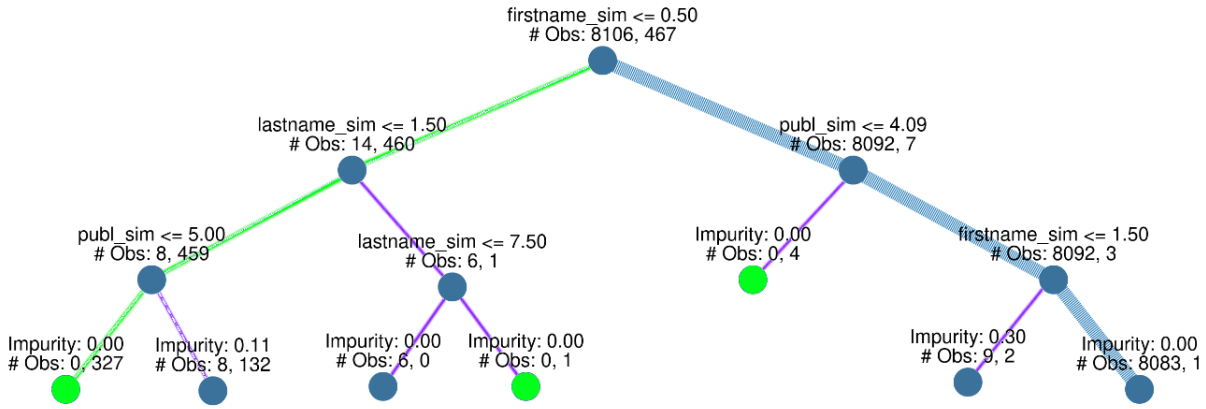

Figure 3.2: Decision tree visualisation with a max depth of 3 and count of positive and negative samples on each node (#Obs). Branches to the left mean the condition is true (lower values mean higher similarity). The thick blue branch represents the most common path for non-matches, while the green branch represents the path for duplicates. Features are adjusted such that smaller similarity values denote a higher similarity. Leaf nodes representing a match are coloured in green.

Model selection

A comparative study was conducted to determine the best model for deciding whether a pair of person profiles represented by a similarity feature vector should be considered a duplicate. The first tested model is a decision tree to maximise interpretability because the set of features is relatively small. Using InterpretML, a library that helps make machine learning implementations more explicable to humans, a plot was made of the decision tree to aid in analysing the results [Nor+19]. For this example, the maximum tree depth was limited to three. A higher depth increases precision but reduces intelligibility. This decision tree visualisation in Figure 3.2 shows that most duplicates can be filtered out using just first name, last name and publication codes. The vast majority of non-duplicate pairs differ in all three of these fields. For duplicates without matching publications, discipline matches become relevant. By inspecting the impurity in each leaf node, it becomes clear that the decision tree is not yet optimal. The second node from the left still includes 132 positive matches with an impurity of 0.13, for example.

Explainable Boosting Machines (EBM) are another very explainable model with accuracy comparable to state-of-the-art blackbox model techniques [Lou+13]. These so called *glassbox* models can show the overall importance of a feature in a decision compared to the other features. The resulting EBM trained on 60% of the person dataset, including 287 positive matches and 5428 negative pairs. A test set with 239 positive and 3572 negative cases was used in order to define model performance. On average, the EBM achieved 0.998% accuracy on the test set. There were five false positives and no false negatives. Using the EBM, it is possible to inspect why these false positives were considered duplicates. An example of this is two profiles for the entity "Gilles De Meester". One works in Ghent and the other in Antwerp. The Gilles that works in Antwerp does not yet have publications to his name but works in "Bacteriology" and "Infectious diseases". These are disciplines in the branch of "Microbiology", one of the research disciplines of the other Gilles. As illustrated in Figure 3.3, the uncertainty about publications and the slight overlap of research disciplines and complete name similarity make this a problematic sample to decide on, even for humans.

Another technique that was tested is logistic regression. This is a classic statistical technique in classification that achieves high performance with identified independent variables. In order to increase the transparency of the model, SHAP values were calculated and analysed [LL17a]. The significance of features on the resulting model was plotted using SHAP values in Figure 3.4. This visualisation again reaffirms the belief that name similarity is essential. Another interesting point to note is that the values for `dis_level2_sim` and `dis_level3_sim` have almost the same shape, except for the latter being shifted more to the right. This indicates that low research discipline similarity on a more specific level has a lower impact, while a high similarity is more significant than at the less descriptive level. An example illustrating the high significance of publication similarity is shown in the force plot of Figure 3.5, where a single matching publication would be enough to make the model consider the pair a duplicate.

The performance of these models has been compared in terms of the average recall and precision of 5-fold

Model	Precision	Recall
Decision tree	0.930	0.911
Explainable boosting machine	0.953	0.911
Logistic regression	*0.972*	0.915
Feedforward neural network	0.953	*0.926*

Table 3.4: Performance of matching models trained on the FRIS person dataset. Precision and recall are calculated based on a 5-fold cross-validation on the resulting pairs of profiles produced by blocking. It is assumed that actual duplicate profiles always have matching ORCID fields.

Sampling technique	Sample counts positive/negative	Recall (duplicates)	Precision (duplicates)
Standard	358/52621	0.9107	*0.9808*
Oversampling	52621/52621	*0.9940*	0.7767
Undersampling	358/358	*0.9940*	0.7952
Cleaning undersampling	358/52576	0.9405	0.9518

Table 3.5: Recall and precision scores using a logistic regression model on person duplicates for different sampling strategies on the same test dataset.

cross-validation on the entire set of blocking results, shown in Table 3.4. As stated above, the precision score may be misleading as it is not yet verified if all true duplicates have the same ORCID. Also, note that this recall is not calculated on the entire dataset but rather on the data presented to the matching model by the blocking algorithm. A feedforward neural network seems to achieve the highest benchmark when considering recall as the most critical metric. However, logistic regression is the best choice in terms of overall performance, as its recall is only marginally lower.

Dealing with imbalance

In order to address the class imbalance caused by the fact that duplicates are very rare in the dataset, over- and undersampling techniques have been explored. Imbalanced learn provides a set of algorithms for these kinds of problems. Oversampling attempts to handle imbalances by producing more samples for the underrepresented classes. This can be done naively by randomly sampling with replacement from the currently available data. More advanced techniques like SMOTE and ADASYN generate new samples by interpolation [Cha+02; He+08]. Undersampling is focused on reducing the samples in larger classes to create balance. This is usually done using prototype generation or selection, where data is removed apart from a number of specific "prototypes" with the most useful samples. Another method in undersampling is cleaning, where data points that do not agree enough with their neighbourhoods are removed. In order to compare these different strategies, a single training set was created and resampled for using each different technique. Logistic regression models were then trained on these resampled datasets and compared on the same testing dataset, by calculating recall and precision in terms of the duplicates or positive cases. For oversampling SMOTE was used, the undersampling technique uses cluster centroids as prototypes and the cleaning method removes samples that do not have the same class as the datapoints in their nearest-neighbourhood. The results are shown in table 3.5. While oversampling and undersampling increase recall, the precision of the resulting model for both of these techniques is quite low. Cleaning undersampling achieves a higher recall than standard sampling while still retaining a decent precision, and was thus selected as the main sampling strategy going forward.

3.1.3 Results

The resulting pipeline used to create a deduplication algorithm for person data is as follows:

1. Blocking based on sorted neighbourhood on last- and first name
2. Nearest-neighbourhood cleaning as a sampling strategy for dealing with imbalances
3. Similarity features for first name, last name, publications, specific- and high-level disciplines
4. Training a matching model using a logistic regression classifier

Algorithm 1: FRIS Person ER pipeline

Data: $n \geq 0$
Result: $y = x^n$
$B \leftarrow$ SN blocking on last- and first name;
; /* Creates a list of pairs of possible duplicate profiles */
$B_c \leftarrow$ nearestNeighbourhoodUndersampling(B);
$Y \leftarrow$ labels(B_c);
; /* Retrieve labels based on ORCIDs of profile pairs in the cleaned blocking set */
$X \leftarrow$ generateSimilarityVector(B_c);
; /* Creates a feature vector for each pair in the list of cleaned blocking pairs, based on first name, last name, publications, specific- and high-level disciplines */
$m \leftarrow$ LogisticRegression().fit(X, Y);

Technique	Recall	Precision
Manual (ours)	0.938	0.972
Dedupe	0.771	0.977
Recordlinkage	0.850	0.969

Table 3.6: Comparison of our manual technique with Dedupe and Recordlinkage. Recall refers in this case, to the fraction of all profile pairs with equal ORCIDs, that are detected as duplicates by the model. Precision indicates the fraction of pairs presented as duplicates that have matching ORCIDs.

When the algorithm is finished, the blocking function and matching model can be reused on unseen and unlabelled data (no available ORCID) in order to detect the most likely duplicates. The results of this final deduplication algorithm were evaluated against the state-of-the-art entity resolution software *Dedupe* and *Recordlinkage* [GE22; De 19]. Dedupe takes a set of attribute names to automatically create blocking rules. Then it clusters the data into a set of duplicates and non-duplicates and selects the most problematic pairs as the samples for manual labelling. These are found by comparing the results of different automatically learned blocking rules and the outcome of similarity metrics over all attributes. By default, the algorithm asks the user for feedback until ten positive (duplicate pairs) and ten negative cases are found, although a higher or lower number of training samples can also be provided. After this manual labelling session Dedupe will use these pairs to improve it's matching model and generates a final set of resulting duplicate clusters along with probability scores.

Recordlinkage is a modular record linkage toolkit to link records in or between data sources. The toolkit provides most of the tools needed for record linkage and deduplication. This program requires the user to input an attribute used for blocking, unlike Dedupe manually, and then automatically computes candidate pairs. Comparison vectors are then generated for these blocking pairs based on comparison metrics defined by the user, such as the *Jarowinkler* edit-distance on the "Name" field. A classifier is then trained on the resulting feature vectors, much like in our manual technique described above. The result is again a set of duplicate pairs, including probabilities that can be manually validated.

Dedupe was tested using the fields first and last name, research disciplines and publications. A manually labelled set of 10 positive and 85 negative cases was generated based on the pairs that conflict in terms of automatically produced blocking rules or similarity metrics to train the classifier. This classification model produced by Dedupe has been tested on the full dataset and compared with our manual logistic regression model. The performance comparison of all three techniques is shown in Table 3.6.

3.2 Projects

Project data consist of information about assignments connecting researchers to institutions, funding organisations and papers. Duplicates in this data are often the result of different organisations collaborating on the same project, each creating a different profile for the project, adding only the researchers associated with their institute. An approach similar to the one used on person profiles was followed for project data. In this case, duplicates are identified based on matching `contract_ids`. Table 3.7 shows a summary of the fields used in feature engineering.

Figure 3.3: False-positive duplicate prediction for Gilles De Meester. The weight the model gives to feature values in order to make the decision is shown as bars. Orange bars are a positive correlation to duplicates, while blue is negative. Remarkably, even though there is an overlap in level3 research disciplines, this is still not high enough to have a positive relationship with the pair being a match. Note also that publication similarity has a value of 10, a value reserved for occasions when one of the two profiles has an empty list of publications. This situation is slightly more common in duplicate pairs, as indicated by the positive relationship.

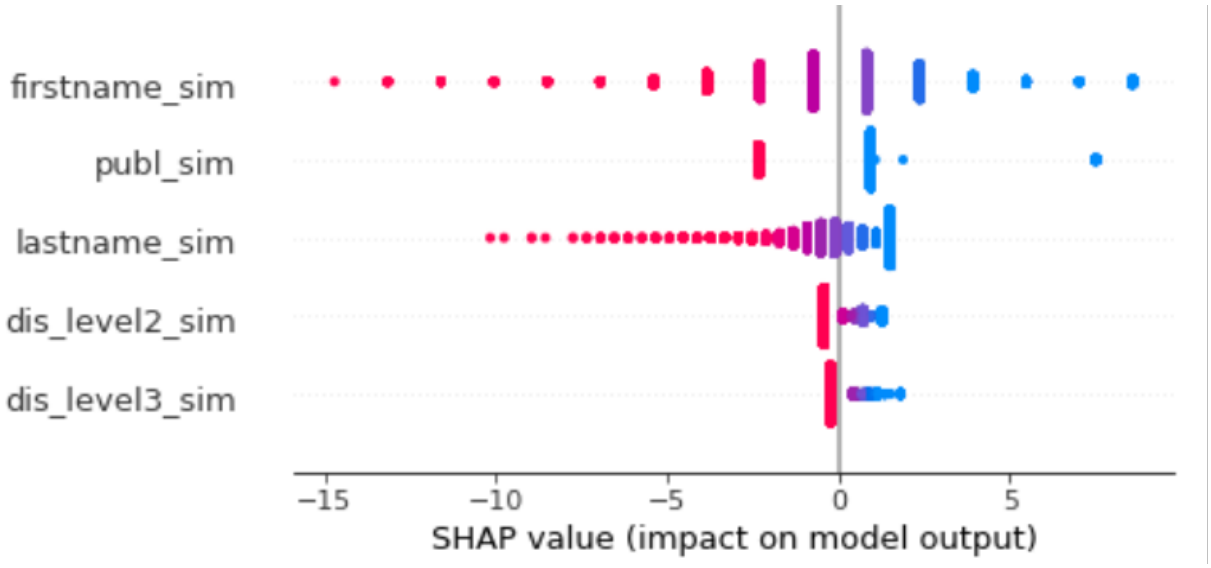

Figure 3.4: Beeswarm plot of SHAP values for a logistic regression model to detect person duplicates. The X-axis shows the impact of a feature, while the blue or red colour indicates lower and higher values, respectively. The size of a line represents the number of occurrences. For example, very high `lastname_sim` values indicate a significant disparity between profile family names. As a similar family name is essential for being a duplicate, high values in this feature are far to the left, indicating a negative SHAP contribution.

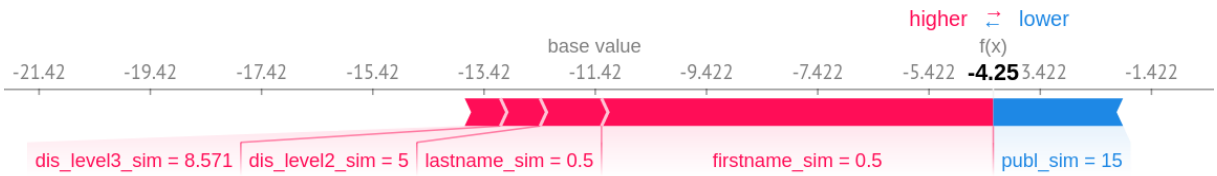

Figure 3.5: A force plot of the false-negative classification for the two profiles of Saïd Hachimi-Idrissi. This sample illustrates the importance of common publications, as the lack thereof negates the effects of a similar last name and set of research disciplines.

Attribute name	Value
title	Trpv4 as a key driver of branch motility and ...
abstract	Microglia are the immune cells permanently ...
start_date	2019-10-01
end_date	2022-09-30
institute	UHasselt
contract_id	12H8220N
research_disciplines	'Innate immunity', 'Cell movement', 'Cell signalling' ...
participants	76693082, 308272719, 76676766, ...

Table 3.7: Project attributes extracted from FRIS data.

Blocking scheme	P-C	P-Q
SN start_date	0.409	0.00334
SN title	0.892	0.00728
SN abstract	0.247	0.00202
LSH title	0.932	0.09379
LSH abstract	*0.965*	0.00175
Union LSH abstract & title	0.965	0.00174

Table 3.8: Results of different blocking schemes on Person data. P-C: pair completeness, percentage of duplicates found. P-Q: pair-quality, percentage of returned pairs that are duplicates.

3.2.1 Blocking

In this case, blocking is a bit more tricky as manual data exploration indicates no field is entirely the same in most duplicates.

Sorted neighbourhood and LSH blocking were compared with different parameters on different fields. Again, recall is, in this case, the most critical metric. Sorted neighbourhood was executed on `title`, `abstract` and `start_date`, as χ^2 tests show that the label (match/non-match) is very dependent on these attributes. LSH was used on both `title` and `abstract`, using different values for shingle size. The results are shown in Table 3.8. As opposed to the data for persons, LSH achieves the best results regarding recall for projects. When used on project abstracts, LSH achieves a recall of close to 97%, leaving only nine duplicate profile pairs.

3.2.2 Matching

Similar to person data, matching is based on the possible pairs of duplicate projects produced by the blocking step.

Feature engineering

Since publication abstracts are long pieces of text and are costly to compare using edit-distance metrics, embeddings were calculated for each profile in order to find the similarity between these long strings in a time-efficient manner. Both BERT-embeddings based on the first 500 characters and TFIDF-vectors were

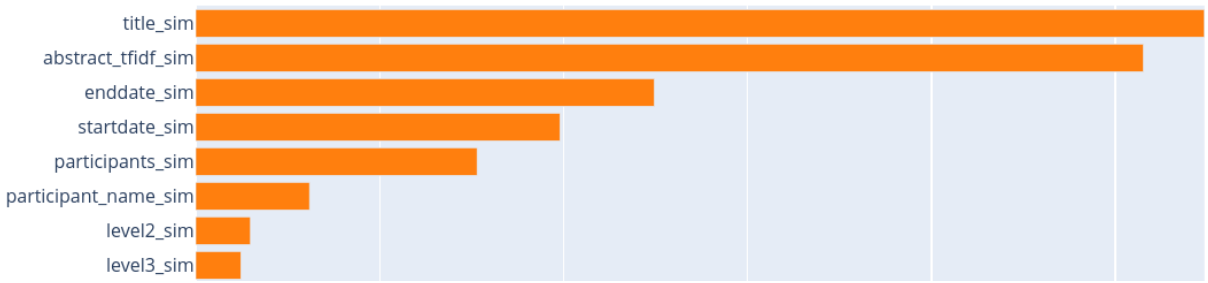

Figure 3.6: Explainable boosting machine mean learned importance of selected similarity features for project profiles.

Model	Precision	Recall
Decision tree	*0.882*	0.891
Explainable boosting machine	0.847	0.954
Logistic regression	0.843	0.968
Feedforward neural network	0.611	*1.000*

Table 3.9: Performance of matching models trained on the FRIS project dataset. Precision and recall are calculated based on a 5-fold cross-validation on the resulting pairs of profiles produced by blocking. It is assumed that actual duplicate profiles always have matching `Contract_id` fields.

Sampling technique	Sample counts positive/negative	Recall (duplicates)	Precision (duplicates)
Standard	2041/1125	0.860	*0.975*
Oversampling	2041/2041	0.875	0.951
Undersampling	1125/1125	0.863	0.968
Cleaning undersampling	997/1125	*0.880*	0.928

Table 3.10: Recall and precision scores using a logistic regression model on project duplicates for different sampling strategies on the same test dataset.

used. Title similarity is calculated using the Levenshtein distance. Research discipline and participant similarity are calculated similarly using the Jaccard coefficient. The start- and end date similarity are calculated using the absolute distance in days. Figure 3.6 shows the relative importance the EBM model gives to each feature in predicting if the pair is a duplicate or not. Interestingly, participant similarity is relatively unimportant. One could imagine a project's participants reaching the same level of importance as publication similarity in finding person duplicates. While examining verified duplicates with the same `contract_id`, it becomes evident that most duplicate projects have different participants. One reasonable explanation is that different funding institutions assign different researchers and only include these people when providing information about the project.

A real example supporting this belief found by using our ER model is the project "Vector embeddings as database views". This project is represented by two profiles in the FRIS database, with a slight difference in Contract_IDs: "G.0192.22N" and "G019222N". This pair had been picked up as a false negative. One version was uploaded by the University of Antwerp, associating their researcher Floris Geerts with the project as the promoter. The other version created by Hasselt University includes a reference to the person Stijn Vansummeren, who was appointed as co-promoter. Neither of these profiles referenced the other institute. Cases like these make it difficult for a supervised model to create the correct association with similarity scores. In fact, when checking the training data, only two project pairs that are labelled as matches have corresponding participants. In comparison, 50 pairs of project profiles with mismatching Contract_IDs have precisely the same participants. Upon closer inspection, these cases are believed to be primarily false negatives and have errors in the Contract_ID field, flagging them as non-matches. Nevertheless, this makes the trained model correlate matching participants with non-duplicate profiles, which is opposite to what we expect.

Model selection

The results of 5-fold cross-validation on labelled project data are shown in Table 3.9. Neural networks achieve an impressive recall of 100%, although it comes at the cost of having an exceptionally low precision as well. Recall is considered the most important metric in this study. However, a precision that is too low will result in the model being unreliable and an increased amount of necessary manual labour. In this spirit, the logistic regression model was chosen for its simplicity and combination of high recall and precision. After applying under- and oversampling techniques, model performance is consistently lower than when the data is used "as-is". This is shown in Table 3.10. When combined with SHAP values, this model provides good interpretability and estimation of match-probability.

3.2.3 Results

The resulting model for project entity resolution consists of the following pipeline:

Technique	Recall	Precision
Manual (ours)	0.843	0.968
Dedupe	0.503	0.980
Recordlinkage	0.817	0.932

Table 3.11: Comparison of our manual technique with Dedupe and Recordlinkage for project data. Recall refers in this case to the fraction of all profile pairs with equal contractIDs, that are detected as duplicates by the model. Precision indicates the fraction of pairs presented as duplicates that have matching contractIDs.

1. LSH blocking on abstracts
2. Standard sampling
3. Similarity features:
   (a) Title: edit-distance
   (b) Abstract: TFIDF cosine-similarity
   (c) Participants: Jaccard-similarity
   (d) Specific- and high-level disciplines: Jaccard-similarity
4. Matching model using logistic regression classifier

The differences between this pipeline and the one used for person data (Section 3.1.3) highlight the notion that different types of data require different approaches, when it comes to producing high-performance entity resolution applications. Comparing this model to existing machine learning techniques yields the results in Table 3.11.

3.3 Evaluation tool

In order to maximise the performance of future models, a tool was created for manual labelling of the most controversial samples in the resultset. These samples consist mainly of profile pairs with differing `ORCID` or `contract_id` fields classified as duplicates with a high probability but also include extremely low probability matches and non-matches. Assigning ground-truth labels to these edge cases will improve the decision boundary and make it easier to interpret precision metrics based on exact `ORCID`/`contract_id` matches.

The tool is based on SN blocking with a logistic regression matching model for persons and LSH blocking on abstract combined with the logistic regression model for projects. The results of these pipelines will be shown to group similar cases together, such as person matches with no publications or project pairs with differing titles. Duplicates are sorted in descending order based on model probability. For each of these duplicates that are found, an explanation is given in the form of a table illustrating similarity values and their importance. The importance of each feature value is calculated by taking the SHAP score for this sample and normalising it using the global maximum and minimum SHAP values. This gives an idea of how much of an effect the feature has on the eventual decision of the logistic regression model. Basic information about the profiles in the duplicate pairs are also given, including a set of publication names and research disciplines in the example of persons. A manual evaluator can scroll through these results in order to label the pairs as either duplicate, non-duplicate or unknown. This information is used to edit the training data, similar to active learning, to make the model more efficient. This entire process is illustrated in Figure 3.7.

The tool allows for matches to be sorted in terms of probability. An example of a very low probability match generated by the tool is shown in Figure 3.8. The two profiles for “Pieter Martens”, of which one lacks an ORCID ID, have somewhat similar research disciplines but lack any common publications. Publication similarity has a higher weight than discipline similarity, as indicated by the “Effect” column, tilting the model towards non-duplicate. The combination of exact first- and last name matches negates this effect resulting in the decision of selecting this pair as a duplicate.

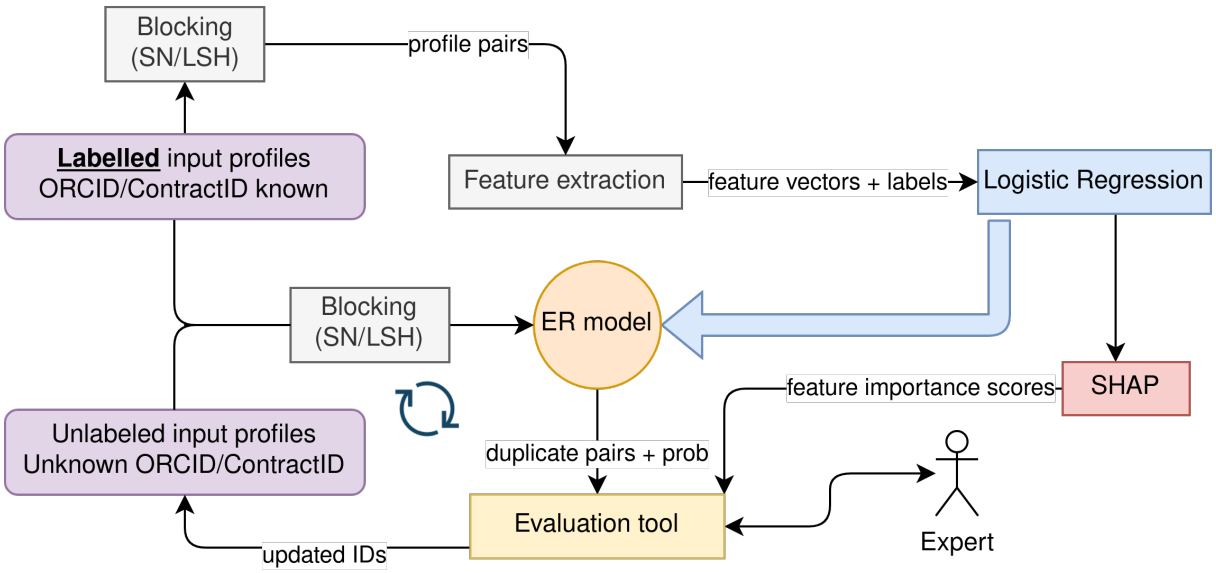

Figure 3.7: Schematic visualisation of the ER pipeline and evaluation tool created for FRIS data. Input is divided into labelled, including unique identifier information and unlabelled profiles. Initially, only the labelled data is used in order to generate a matching classification model based on logistic regression. This model is then turned into an ER model that can determine the probability of a pair of profiles being duplicates. Profile pairs the model is uncertain about, thus resulting in a low probability, are selected and explained to users in the evaluation tool. Manual labelling done by users is converted into new IDs assigned to the dataset profiles. Once user evaluation is finished, the updated set of labelled profiles is used as training data for creating the next version of the model.

Similarity feature	Value	Effect
firstname similarity	Very high	Very positive
lastname similarity	Very high	Positive
publication similarity	Very low	Negative
specific disciplines similarity	Low	Neutral
high-level disciplines similariy	Moderate	Neutral

FRIS ID: 90eed7f2-2ca5-4d36-9be9-946e2fbfead9

Full name: Pieter Martens

City: Heverlee

Research disciplines: Architectural engineering, Architecture, Interior architecture, Architectural design, Architecture, Art studies and sciences

First 5 publications (34):

- The reverse remodeling response to sacubitril/valsartan therapy in heart failure with reduced ejection fraction
- A New View on the World. The Cartographic and Chorographic Publications of Hieronymus Cock
- Virtual Palaces, Part I. Digitizing and Modelling Palaces
- Siege warfare (early modern)
- Une estampe inédite du siège d'Hesdin en 1553

ORCID: 0000-0002-8743-755X

FRIS ID: 9a851119-54ce-4d6b-915b-983f75ff5400

Full name: Pieter Martens

City: Brussels

Research disciplines: Architectural history and theory, Architectural design history and theory , Architectural heritage and conservation, History of art, Early mod

history, Algebra not elsewhere classified

First 5 publications (8):

- Occasions of State: Early Modern European Festivals and the Negotiation of Power
- Le città fortificate nei domini spagnoli delle Fiandre

Figure 3.8: Example of a person profile pair selected as duplicate by the ER model with a low probability. The table above visualises the importance of different feature similarity scores. Below, a summary of the data contained in both profiles is given.

Chapter 4

Intego Data: Record Linkage

Intego is a Flemish general practitioner's registration network led by the Academic Center for General Practice of the KU Leuven and financed by the Flemish government. The institution possesses electronic medical record data from 1994 and uses this data for research purposes. Intego has data about almost 2% of the Flemish population, including 4.4 million diagnoses, 16.7 million prescriptions and 45 million lab results. At some point in their data collection, Intego completely restructured the way they store data. During this change, data in the old scheme were meant to be converted to the new structure. In reality, this conversion was incomplete, and many old medical data were not transferred into the new records. As data are anonymized, and patient_ids use a different technology between the old and new datasets, finding out which parts of the data still have to be merged and linking the old id to the new one poses a difficult challenge. An effort has been made to create a set of verified matches using an exact overlap of information on a small number of patients, but this set is far from complete.

4.1 Available data

The old data that has to be merged consist of five sets:

- Patient information: This dataset includes `patient_id`, `birth_year` and `gender`.
- Diagnoses: `patient_id`, `date`, `ICPC_code` and `ICD10_code`.
- Lab file: `patient_id`, `date`, `lab_type` and `lab_result`.
- Parameters: `patient_id`, `date`, `parameter_type` and `parameter_value`.
- Prescriptions: `patient_id`, `date` and `ATC_code`.
- Vaccine file: `patient_id`, `date` and `vaccine_code`.

The schema of the new data can be loosely mapped on the old data based on these fields, although it generally includes more information. Along with these two datasets, a set of already matched patient IDs is also available. These patients have been linked by comparing their diagnoses, lab results, parameters and prescription codes and dates. If a patient in the old set has exact matches in each of these sets, the probability of being a match is considered to be 1. Another set of 100k patient mappings from old to new is also available, consisting of matches based on SSN data. These patient profiles have been combined regardless of overlap of values in their attributes. They might prove more useful than the biased set of "conservative" matches, as those only include matches where overlap is very clear. As no labelled data are available, the matching requirements for these patients are rigorous to ensure their accuracy. This provides a suitable initial set of records to use as validation data regarding recall. Just as was the case in Chapter 3, precision is difficult to define.

4.2 Blocking

Blocking is, in this case, based on patient birth year and gender, as these are the only available constants between patient matches. Based on 100 randomly sampled profiles in the old dataset, the mean number

id_old	id_new	match	icd10_day	icd10_list	icd10_cosine	atc_day	atc_list
15431525	0070-2205	True	1.00	1.00	1.00	0.05	0.20
42983064	0038-5109	True	0.00	0.20	0.51	0.10	0.15
78666201	0082-9618	False	0.00	0.06	0.10	0.01	0.01

Table 4.1: Example samples from the feature dataset that was generated on matched patient pairs and random combinations of patients.

	Non-duplicate pairs	Duplicate pairs
1	Influenza	Influenza
2	Panniculitis	Gastro-enteritis
3	Laryngopharyngitis	Panniculitis
4	Nasopharyngitis	Laryngopharyngitis
5	Gastro-enteritis	Nasopharyngitis

Table 4.2: Most common overlapping diagnoses for non-duplicate patient profile pairs and duplicate patient profile pairs, respectively.

of profiles in each block is calculated to be 33647. Assuming no noise exists in the birth-year and gender attribute values, the blocking scheme has a pair-completeness of 1 and a pair-quality of about 0.0000297. This is a low score considering each one of the 2,8 million patients in the old dataset will have to be compared with all potential matches in their blocking set, resulting in approximately 94 billion different comparisons. A single prediction takes approximately 0.002 seconds to complete. For an entire block, this would give a duration of 67.3 seconds in order to link a single profile. This means that the record linkage algorithm would take six years to complete when running on the entire dataset. A better blocking scheme is thus mandatory in order for the algorithm to finish in a reasonable time.

As illustrated by the decision tree representation in Figure 4.3, many matches have at least some overlapping diagnoses (ICD10 + date). It might then be possible to create a blocking scheme leveraging this knowledge that uses LSH to put patients with similar diagnoses signatures in the same blocks. This will inevitably result in a lower recall metric, so a multi-pass technique using both precise LSH blocking and wide year+gender blocking is required to maximise the number of found matches. This ensemble blocking technique would work as follows. First, a set of blocks for both the standard blocking on birthyear+gender and LSH blocking on ICD10 signatures will be produced. In the record linkage process, the smaller LSH block, which is a subset of the standard block, will be selected first for analysis. If a suitable match was found in this block, meaning the probability of matching exceeds a predefined threshold, the matching for this patient is considered complete, and the algorithm will continue with the next patient. If no new patient in the LSH block receives a matching probability higher than the threshold, the algorithm will proceed to check all patients in the standard block, excluding the ones already tested. This method has not been tested, but it can be an interesting approach to blocking for future implementations.

4.3 Matching

4.3.1 Feature engineering

A dataset containing feature vectors and labels was created in order to find the most suitable model for predicting matching patients. These data are based on the patients that Intego already matched for the positive cases and an equal amount of random pairs of old and new patients for the negative cases. Each of the available datasets mentioned in section 4.1 was used to generate one or more similarity measures for use in the feature vectors. For diagnoses, one similarity measure is based on the Jaccard similarity of the combination of `ICD10_code` and `date`, one based on the Jaccard similarity of just the list of `ICD10_code` of two patients and one taking the cosine similarity of a TFIDF-vectorized version of the `ICD10_code` list represented as a string. This last feature was included in order to handle missing data more intelligently, as two patients might not have the same diagnoses in the old and new datasets. However, there might be a relationship between the two lists of codes. With this, we mean, for example, that certain diseases or ailments may typically occur due to one another.

Another important trait of diagnosis codes is the difference in frequency of diseases. A widespread

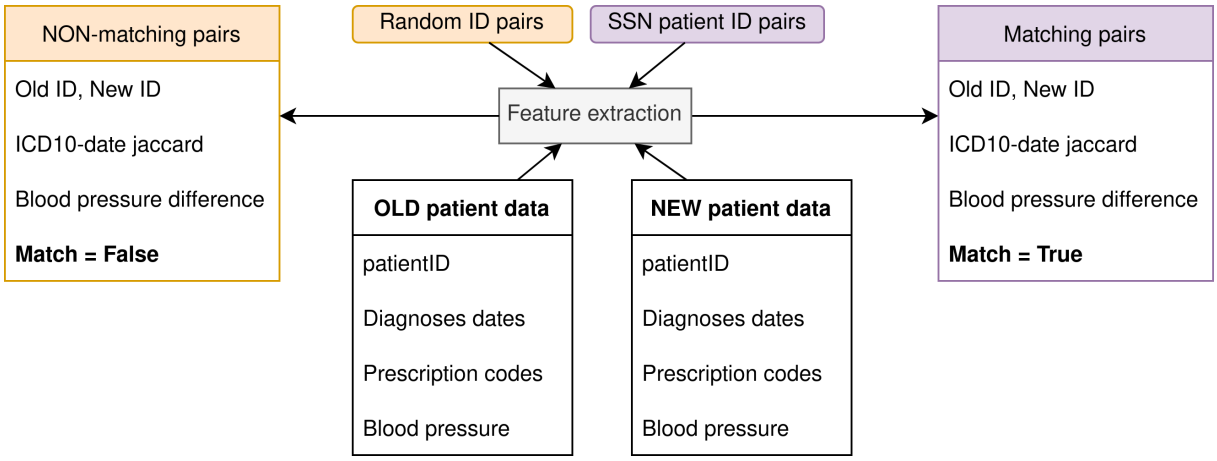

Figure 4.1: Visualisation of the data generation and feature extraction pipeline. SSN verified patient profile pairs are combined with similarity features of randomly selected profiles to generate a labelled dataset suitable for training a machine learning model. Only a subset of the features that were used is shown.

Model	Recall	Precision
Quadratic discrimination	0.1675	0.9105
Neural network	0.9805	*0.9238*
Decision tree	*0.9838*	0.9238
SVM	0.9454	0.9892
Logistic regression	0.0010	0.0002

Table 4.3: 5-fold cross-validation scores on Intego feature vectors.

diagnosis like "influenza" can occur as a shared diagnoses code both for matching and non-matching patient pairs (see Table 4.2), simply because of its prevalence. We attempt to mitigate this problem by modelling code overlap importance using an `icd10_overlap_weight` feature. This is based on the average code_weight for overlapping ICD10 codes in a particular profile pair. These `code_weights` are in turn calculated for each possible diagnoses code, by counting the number of occurrences in overlapping codes of non-matches and dividing this by the total number of non-matching pairs.

Features based on prescription and vaccine data are generated similarly, with one feature for each dataset based on the Jaccard coefficient of `ATC_code` combined with `date`, one based on the Jaccard of just the code lists for prescriptions and one based on just the vaccine dates. Parameter similarity is modelled by comparing the lists of type-value-date combinations but also by calculating the difference of mean values for the parameter types "blood pressure", "length", "weight", "BMI", and "heart rate". These features are added based on the idea that matching patients will have similar values for these measurements as they are essentially the same person, especially in length.

Combining these similarity measures results in a data structure similar to the one in Table 4.1. Feature importance was calculated based on χ^2 tests, resulting in the highest scores achieved by the `ICD10_day`, `ICD10_list`, `ATC_list`, `ATC_day` and `param_day` fields, respectively. A similar importance score is reflected in by plotting the SHAP values of a trained decision tree, as visualised in Figure 4.2.

4.3.2 Model selection

The machine learning model chosen for further analysis was based on high recall in 5-fold cross-validation (Table 4.3) and general interpretability of the model. Combining these two requirements leads to the selection of decision trees, as these achieve near maximum recall while still retaining excellent explainability.

Training a decision tree on the data using a depth limit of 3 already achieves a recall of 0.9840 and a precision score of 0.9034 on the annotated patient matching dataset. The visualisation in Figure 4.3 shows that this performance is already achieved by only using the attributes `ATC_day` and `ICD10_day`.

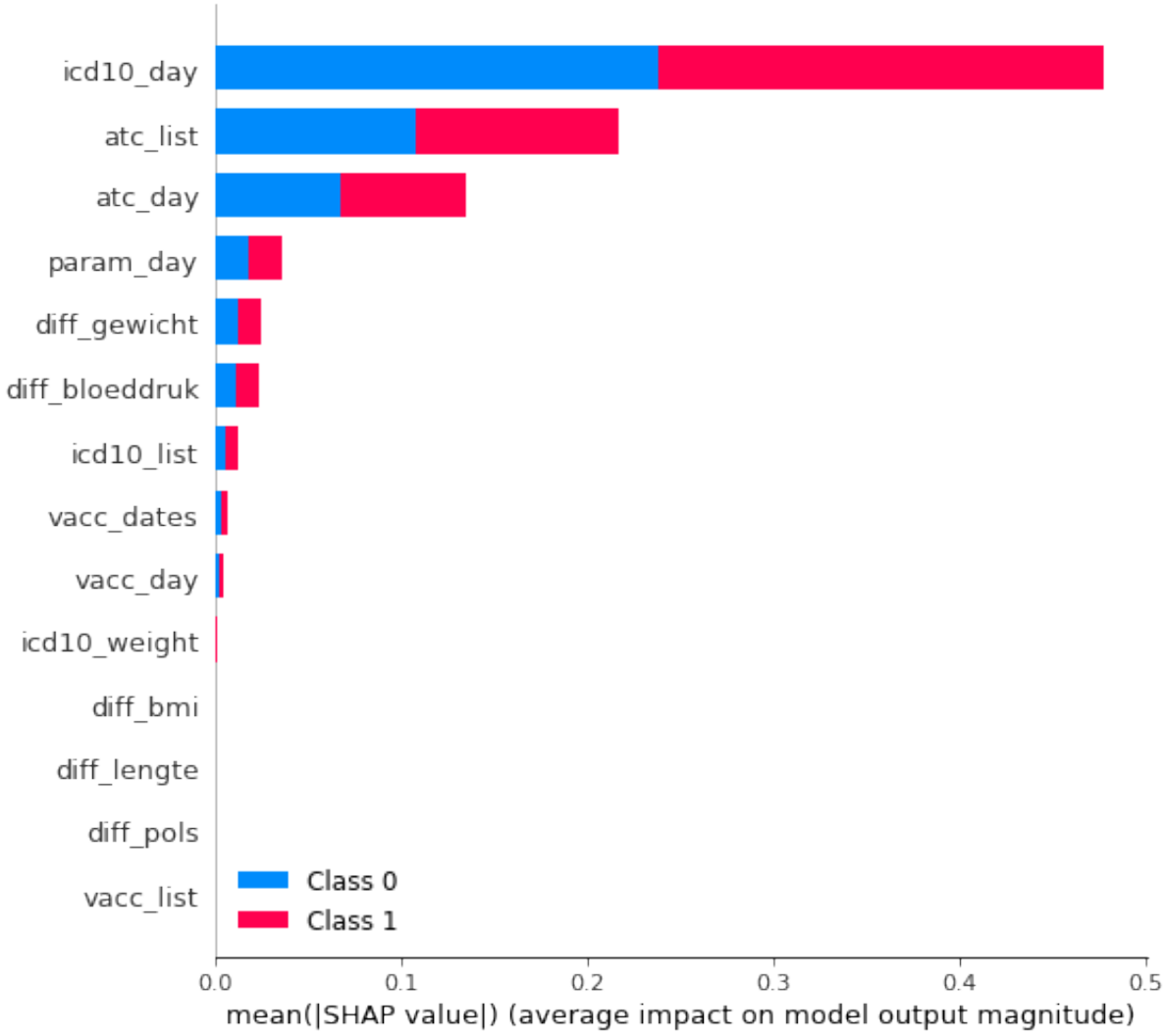

Figure 4.2: Bar chart describing the importance of the features used in the Intego tree using SHAP values. Similarity features with a higher impact in the resulting decision tree have a higher average SHAP value and can be used more effectively by the model to separate matches from non-matches.

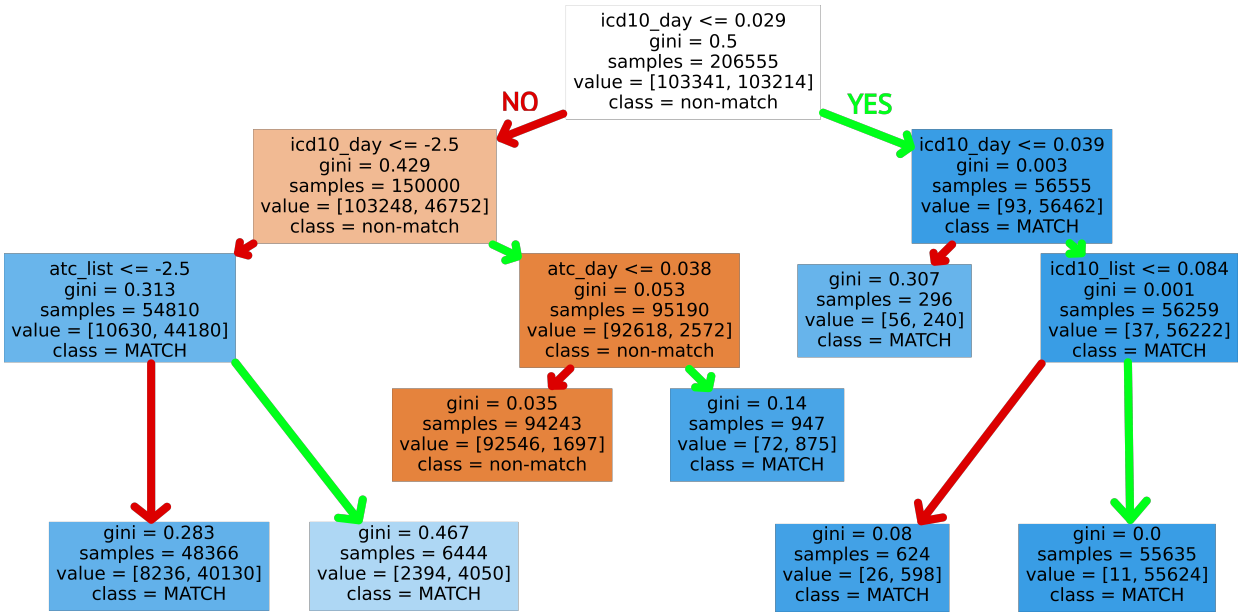

Figure 4.3: A decision tree with `max_depth=3` trained on labelled Intego pair similarity features. The colours blue and orange indicate a node is assigned more to matching and non-matching pairs, respectively.

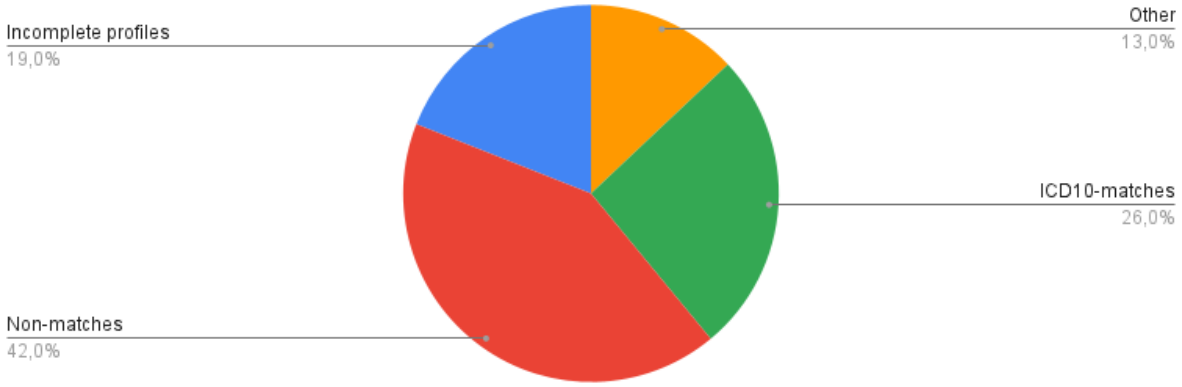

Figure 4.4: Chart visualising the population of the most common leaves of a decision tree trained on 100k matches extracted from SSN data and 100k random pairs of the old- and new patients considered non-matches.

4.4 Results

Upon analysis of the resulting decision tree model, it becomes apparent that 87% of all training profile pairs are assigned to one of three significant leaves. This separation of results is visualised in Figure 4.4. Right most leaf, accounting for 26% of all training samples, contains pairs with the highest possible matching probability, i.e. pairs of patients with very high overlap in ICD10 diagnosis code and date pairs. The largest leaf is made up mostly of non-matching pairs with little to no diagnoses or prescriptions in common. This leaf is assigned to 42% of the total sample pairs, which is a logical number as 50% of the data were random combinations of profiles added as non-matches. The third of these major leaves, representing 19% of the samples, is comprises pairs where at least one profile consists of mostly incomplete data. In this case, the model does not have enough information to make an informed decision. Interestingly, $\frac{3}{4}$ of this leaf is made up of matching profile pairs, leading to the tree deciding that similar samples must be classified as duplicates. This poses a problem for future predictions on unseen data, as an incomplete profile will be able to match with *any* other profile. To tackle this misconception, largely incomplete profiles will be filtered out for the model's training and eventual classification phases. The other 13% of the training data consists of matches and non-matches assigned to less common leaves of the

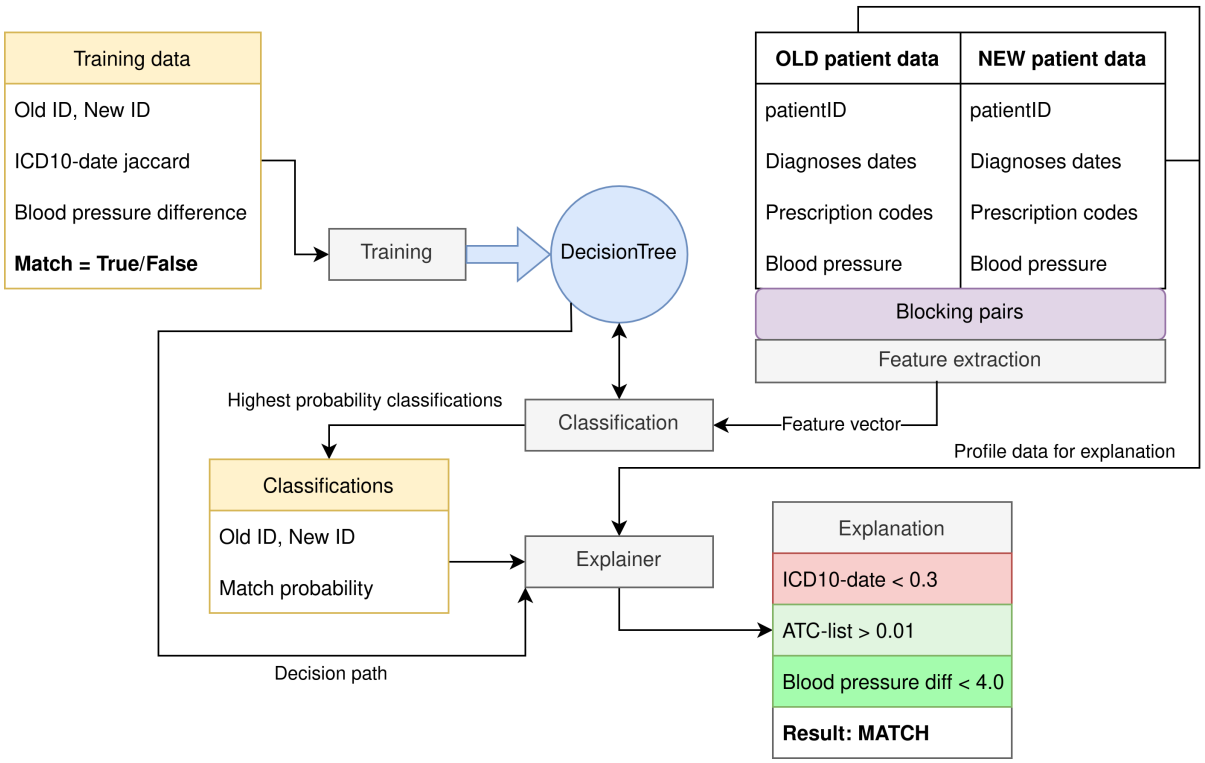

Figure 4.5: Schematic visualisation of the classification and explainer tool created for Intego patient data. First, labelled training data is used to build a decision tree. The classification algorithm then uses this tree to compute the matching probability for pairs of possible duplicate profiles defined in the pre-calculated blocking set. For each old patientID, the new patientID with the highest matching probability is chosen, and the pair of IDs is added to a set of classifications along with its probability. This set of classifications is then passed through to an explainer algorithm that produces a table visualising the decision path a given classification sample takes, including example values and indications of classification effect.

tree, such as pairs that only match in terms of prescription ATC codes and not in ICD10-codes.

The main objective of this project concerning Intego is to find a correct and complete mapping between old- and new patient id's. To this effect, an algorithm has been designed to find the best matching new patient profile for each profile in the old dataset. Performance can be increased by determining a stringent matching probability threshold used to stop searching. Once a match satisfies this sufficiently high threshold, such as a branch in the tree with a zero gini score, the algorithm will consider this as the best match and continue with another patient. Old patient profiles that have been matched are saved to a separate dataset that can be read and used by an evaluation tool.

4.5 Tool

A similar tool as the one created in Chapter 3 has been made for use in this context. The main idea is still to represent duplicate patient pairs as clearly as possible and indicate why the model thinks it is a duplicate. This will allow workers of Intego to interpret results and identify matches that have not yet been found using the strict hierarchical matching. An added challenge for these data is that the volume of data is too large to work through all patients in one go. The tool needs to be capable of handling a continuously growing pool of potential matches and handle the retraining of matching models while the algorithm is still running. Another complication is the fact that the relationships between profiles are more difficult to interpret. A 10% match in terms of diagnoses codes is less meaningful than a family name that matches apart from one character. It is thus of even greater importance to the end-user to visualise feature importance in a way that makes the relationships between two profiles clear. This project on Intego does not make use of a blackbox model like the logistic regression used for FRIS data, so visualising predictions becomes more straightforward. For each matching pair, the tool will give an explanation of

the decision tree path followed with threshold values at each decision node. Example values in the case for overlapping lists are shown by converting codes to descriptions and showing a measure of "rareness", which conveys how often a match like this occurs.

For each prediction, a table is generated based on the decision path of the trained tree model. On every node that is passed for a particular decision, the condition of the node (e.g. "Blood pressure difference" < 2 kg) is shown with a colour indicating the importance of being considered a match, as well as the actual value and examples in the case of an overlap-based similarity feature. The effect colour is decided by comparing the amount of non-matching and matching pairs in the training data that have the same outcome on the condition as the current sample pair. Consider an example pair with a very high overlap in diagnoses codes with dates, the *ICD10-date* overlap feature is 0.7. The node in the explanation table with the condition "ICD10-date > 0.05" will be very green for example, because the amount of matching pairs in the training data that also meet this condition vastly outnumber the amount of non-matches that do. In this example, it means that this condition is considered very important for deciding that the particular sample pair will be classified as a match. Similarly, conditions which are also mostly met by non-matching pairs will result in the condition node being coloured red. An even split in matches and non-matches will make the node grey coloured to indicate that meeting the condition is relatively unimportant for the eventual decision.

The flow of this tool is schematically explained in Figure 4.5. An example of this tool used on a pair of patients is shown in Figure 4.6. This example pair, in this case, has zero ICD10 matches as the new patient profile does not include diagnoses. The model will then check for the second most important feature, which is based on prescription codes. Even though the two profiles only share a single matching prescription and date combination, the pair is considered a match as many prescribed medication codes overlap. When looking at the other variables included in the profiles, a reasonably similar height can be observed, with the older profile having a slightly smaller value. This is a good indicator of the pair being a duplicate, as height rarely changes in adults. The older profile being somewhat shorter can be caused due to measurements before the subject finished growing.

4.6 Future work

Intego workers are equipped with the tools to evaluate matchings between old and new patient data using this classification model and explainer.

Using this classification model and explainer, Intego workers are equipped with the tools necessary for evaluating matchings between old and new patient data. ID pairs of evaluated profiles can be added to a dataset of matches extending the one created in the SSN matching process. However, manually checking all of these classifications would be a very lengthy endeavour. As mentioned in Section 4.2, when matching old profiles, 2,8 million classifications must be manually looked at in the worst case. Instead of doing all of these evaluations manually, an extension of the tool can be made which evaluates classifications for each unique node only once. This approach takes advantage of the tree structure of the classification model and treats each output leaf as a classification "case". In our example model with a `max_depth` of 8, this results in only 35 manual evaluations that need to be done to classify the entire remainder of the dataset automatically.

A shortcoming of the current tool is the representation of the sample explainer, an aspect that could be improved upon in a future study. Feature importance could for example be calculated based on the average of the rule effects that are explained in Section 4.5. Along with these representations of importance, values for each similarity feature can be normalised to give users insight in the relative magnitude of each attribute. Currently, values used in the conditions have little meaning. It is unclear to a normal user that a value of 0.2 for the `ICD10-date` field is exceptionally high, while it is a normal value for `ICD10-list`. Understandability of the model would improve when similarity values are normalised to a scale that is the same for all features.

VERY common for non-matches	Common for non-matches	Neutral	Common for matches	VERY common for matches

Tree explanation for 619028011 and INTEGO_PATIENTID_0085-6109

Rule	Value	Examples
Diagnoses code and date overlap <= **0.029**	undefined	/
Diagnoses code and date overlap IS **NOT defined**	undefined	/
Prescription ATC code and date overlap IS **defined**	0.003	N05B
Prescription ATC code and date overlap <= **0.01**	0.003	N05B
Prescription ATC code overlap > **0.308**	0.333	D06A,N06A,G01A,A03A,N05B

Result: **MATCH** with probability 0.931

Full profiles:

Attribute	Value
Patient ID	619028011.0
Diagnoses ICD10 codes	S93, M77, G43, N10, E78, R42, ...
Prescription ATC codes	M01A, C07A, N06A, C07A, N06A, J07B, ...
Vaccine dates	undefined
Avg. blood pressure	121.0
Avg. weight	74.0
Avg. heartrate	72.0
Avg. height	163.0
Avg. BMI	undefined

Attribute	Value
Patient ID	INTEGO_PATIENTID_0085-6109
Diagnoses ICD10 codes	undefined
Prescription ATC codes	A03F, R05C, N05B, R01A, H02B, G01A, ...
Vaccine dates	1994-06-30, 1994-07-26, 1995-07-05, 2001-11-27, ...
Avg. blood pressure	129.0
Avg. weight	55.0
Avg. heartrate	undefined
Avg. height	165.0
Avg. BMI	undefined

Figure 4.6: Result of running the intego classification explainer tool on a patient profile pair with a matching probability of 0.931. In the *Rule* column, conditions of tree nodes in the decision path are represented with a colour that shows how the value affects the model result. For similarity features based on list overlap, some examples of the intersection of lists are shown in the *Examples* column. Below this explanation a representation of the essential features of both old and new profiles is also visualised.

Chapter 5

Conclusions

Many applications, organisations or companies aggregate information about objects or people. Through human error, bugs or data merges, this information can contain multiple entries for the same subject, resulting in information loss or inefficient redundancy. It has been shown in this study that machine learning techniques are effective at detecting these corresponding profiles. Through careful analysis of the traits and features of the data, tools can be built to aid humans in cleaning or merging datasets of this kind. Using modern methods such as SHAP values, it is possible to create extensive explanations for matching decisions that can make ER applications more interpretable and tweakable by users. Entity resolution techniques can help experts in identifying duplicate entries in many different domains.

One of the main limiting factors in ER performance is the quality and amount of available data. In the two cases analysed in this study, a set of labelled samples was available to train supervised models. Without this training data, different approaches would have to be considered, and the focus would be shifted to clustering and active learning. Comparing how well a clustering-based technique would perform on our data as opposed to the current supervised models would be an exciting study in future work. Not only is there a need for a sufficiently large pool of pre-matched profiles, but the data also needs to contain attributes that can somehow be transformed into helpful similarity features for separating duplicate pairs from non-duplicates. With FRIS projects, for example, the achieved performance was lower than with the person data. This has been mainly due to the fact that there were fewer valuable attributes to make use of, as even projects with a matching abstract and title could still have a different contractID. Machine learning tools were, in this case, helpful in identifying similarity features that had a small or even adverse impact on the efficiency of the model. For example, one could expect duplicate projects to share associated researchers or have overlapping time ranges. Both of these features have a negative correlation with the outcome variable, meaning that more often, project pairs with a different contractID have high similarity than projects with matching IDs. Similarly, the time range in which a researcher has actively made publications could be considered helpful in comparing the similarity between two persons. This proved to be a misconception because duplicate entries for researchers are often caused due to people changing from one institute to another. One profile will have an active period until the switch, while the other will have a publication range starting after the change. These contradictions between expectation and outcome are common in machine learning problems but can make the decision-making more prone to errors when there are insufficient important features.

Another critical factor that played a significant role when designing tools and techniques to solve these two problems has been building the level of trust users can have in the models. In FRIS and Intego data, the primary use of the ER applications has been in classifying profile pairs into duplicate or non-duplicate with zero ground truth knowledge. No verification could be made, apart from manually checking a fraction of the results. With this lack of certainty arises the need for ways to prove that the trained models base their decisions on correct assumptions. This need for decision clarity has been addressed by implementing graphical programs that indicate the effects of features important in making the eventual decision of duplicate or non-duplicate.

In the FRIS case, this has been made possible through training a SHAP model and normalising SHAP values to a scale of 0 to 100 for all similarity features of each sample pair. Based on thresholds, this value determines the effect of the feature on the outcome from “very negative” to “very positive”. Negative

and positive in this case refer to the decision variable of the sample pair being a number between 0 and 1, with 0 being a definite non-match and 1 being a match. A negative effect would thus lower this variable closer to the non-match outcome, while a positive effect would raise the value to promote the matching outcome. The feature itself is also given a normalised value, from "very low" to "very high" in order to make these features easier to evaluate. Using a feedback mechanism, a user can then compare the similarity value to the actual data and decide if the correlation with the effect was correct.

In the case of Intego, the chosen model is a decision tree. This model type is based on rules, where certain meeting conditions determine the outcome. This type of model is based on a set of rules, where meeting certain conditions determines the outcome. A model like this is inherently interpretable, as each sample outcome can be explained using the set of rules that led to a specific outcome. To make these rules more meaningful, each condition that was or was not met has been indicated with a colour representing the distribution of training samples with the same outcome for the particular rule. The green colour shows that most of the training samples with the same result as the sample pair on a condition node were matching pairs. Through this representation of rules, the general effect of the fulfilment of a particular condition on the final outcome becomes clear.

We can conclude that for FRIS, the most efficient blocking techniques were Sorted Neighbourhood and LSH. The matching techniques that achieved the best balance between performance and explainability are Logistic Regression in the case of FRIS, and Decision Trees in the case of Intego. Explainability has been increased through SHAP values and graphical examples In general, the most crucial similarity features are based on fields with well-defined, complete and straightforward values, such as names and publications in the case of FRIS persons and ICD10-date overlap in the case of Intego. As an answer to the final research question posed in the introduction, it is demonstrated that accurately predicting duplicates using modern techniques is possible. However, effort must be made to build users' trust in the system for these models to be usable. This challenge was tackled by introducing tools capable of exposing the decision process, balancing the level of control between man and machine and, in turn, addressing the requirement of certitude. Even with limited data, creating a helpful tool for predicting the probability that two profiles refer to the same real-world entity is possible. During the entire process of development, the focus must be on transparency and well-founded predictions to limit the effect of human assumptions in the feature engineering and model selection stages.

Bibliography

[BKM06] Mikhail Bilenko, Beena Kamath, and Raymond J Mooney. "Adaptive blocking: Learning to scale up record linkage". In: *Sixth International Conference on Data Mining (ICDM'06)*. IEEE. 2006, pp. 87–96.

[Cha+02] Nitesh V Chawla et al. "SMOTE: synthetic minority over-sampling technique". In: *Journal of artificial intelligence research* 16 (2002), pp. 321–357.

[Cha02] Moses S Charikar. "Similarity estimation techniques from rounding algorithms". In: *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*. 2002, pp. 380–388.

[Chr11] Peter Christen. "A survey of indexing techniques for scalable record linkage and deduplication". In: *IEEE transactions on knowledge and data engineering* 24.9 (2011), pp. 1537–1555.

[Coh00] William W Cohen. "Data integration using similarity joins and a word-based information representation language". In: *ACM Transactions on Information Systems (TOIS)* 18.3 (2000), pp. 288–321.

[De 19] J De Bruin. *Python Record Linkage Toolkit: A toolkit for record linkage and duplicate detection in Python*. Version v0.14. Dec. 2019. DOI: 10.5281/zenodo.3559043. URL: https://doi.org/10.5281/zenodo.3559043.

[EVE02] Mohamed G Elfeky, Vassilios S Verykios, and Ahmed K Elmagarmid. "TAILOR: A record linkage toolbox". In: *Proceedings 18th International Conference on Data Engineering*. IEEE. 2002, pp. 17–28.

[FS69] Ivan P Fellegi and Alan B Sunter. "A theory for record linkage". In: *Journal of the American Statistical Association* 64.328 (1969), pp. 1183–1210.

[GE22] Forest Gregg and Derek Eder. *Dedupe*. https://github.com/dedupeio/dedupe. 2022.

[He+08] Haibo He et al. "ADASYN: Adaptive synthetic sampling approach for imbalanced learning". In: *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*. IEEE. 2008, pp. 1322–1328.

[HGI00] Taher Haveliwala, Aristides Gionis, and Piotr Indyk. "Scalable techniques for clustering the web". In: (2000).

[Hou+20] Boyi Hou et al. "Gradual machine learning for entity resolution". In: *IEEE Transactions on Knowledge and Data Engineering* (2020).

[HS95] Mauricio A Hernández and Salvatore J Stolfo. "The merge/purge problem for large databases". In: *ACM Sigmod Record* 24.2 (1995), pp. 127–138.

[Jac12] Paul Jaccard. "The distribution of the flora in the alpine zone. 1". In: *New phytologist* 11.2 (1912), pp. 37–50.

[Lac+13] Simon Lacoste-Julien et al. "Sigma: Simple greedy matching for aligning large knowledge bases". In: *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2013, pp. 572–580.

[Lev66] Vladimir Iosifovich Levenshtein. "Binary codes capable of correcting deletions, insertions and reversals." In: *Soviet Physics Doklady* 10.8 (1966). Doklady Akademii Nauk SSSR, V163 No4 845-848 1965, pp. 707–710.

[LL17a] Scott M Lundberg and Su-In Lee. "A Unified Approach to Interpreting Model Predictions". In: *Advances in Neural Information Processing Systems 30*. Ed. by I. Guyon et al. Curran Associates, Inc., 2017, pp. 4765–4774. URL: http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf.

[LL17b] Scott M Lundberg and Su-In Lee. "A unified approach to interpreting model predictions". In: *Advances in neural information processing systems* 30 (2017).

[Lou+13] Yin Lou et al. "Accurate intelligible models with pairwise interactions". In: *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2013, pp. 623–631.

[MNU00] Andrew McCallum, Kamal Nigam, and Lyle H Ungar. "Efficient clustering of high-dimensional data sets with application to reference matching". In: *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*. 2000, pp. 169–178.

[MZ98] Tova Milo and Sagit Zohar. "Using schema matching to simplify heterogeneous data translation". In: *vldb*. Vol. 98. Citeseer. 1998, pp. 24–27.

[Nor+19] Harsha Nori et al. "InterpretML: A Unified Framework for Machine Learning Interpretability". In: *arXiv preprint arXiv:1909.09223* (2019).

[Pap+15] George Papadakis et al. "Schema-agnostic vs schema-based configurations for blocking methods on homogeneous data". In: *Proceedings of the VLDB Endowment* 9.4 (2015), pp. 312–323.

[PBK20] Anna Primpeli, Christian Bizer, and Margret Keuper. "Unsupervised bootstrapping of active learning for entity resolution". In: *European Semantic Web Conference*. Springer. 2020, pp. 215–231.

[PWN06] Sven Puhlmann, Melanie Weis, and Felix Naumann. "XML duplicate detection using sorted neighborhoods". In: *International Conference on Extending Database Technology*. Springer. 2006, pp. 773–791.

[RU11] Anand Rajaraman and Jeffrey David Ullman. *Mining of massive datasets*. Cambridge University Press, 2011.

[San+19] Victor Sanh et al. "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter". In: *arXiv preprint arXiv:1910.01108* (2019).

[Uni] Ghent University. *Flemish research disciplines*. URL: `https://www.ugent.be/en/research/research-ugent/research-discipline.htm`.

[Vam+20] Venkata Vamsikrishna Meduri et al. "A Comprehensive Benchmark Framework for Active Learning Methods in Entity Matching". In: *arXiv e-prints* (2020), arXiv–2003.

[Yan+20] Yan Yan et al. "Entity matching in the wild: A consistent and versatile framework to unify data in industrial applications". In: *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. 2020, pp. 2287–2301.

[Zha+10] Meihui Zhang et al. "On multi-column foreign key discovery". In: *Proceedings of the VLDB Endowment* 3.1-2 (2010), pp. 805–814.

[Zha+11] Meihui Zhang et al. "Automatic discovery of attributes in relational databases". In: *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*. 2011, pp. 109–120.