Final Report

#Team: 15, Team ID: C5
Team members: Cristina Zenteno Garcia, Ian Shepherd
GT Usernames: cgarcia66, ishepherd3
GTIDs: 903647123, 903653735

1. Introduction

Record linkage is a complex problem in the medical industry and has become an essential step in the data cleaning process. It allows data from different sources to be linked to the same entity to ensure patient data relates to the correct individual. One of the uses is in the healthcare setting where record linkage can be used to link medical records from different hospitals to create data that can be used to provide better healthcare services for the patient, track the spread of diseases and improve the quality of data for research.

The paper we are reproducing[1] aims to use ensemble classification methods like bagging and stacking over popular base models such as support vector machine, logistic regression and feedforward neural network to evaluate if the ensemble methods could achieve better results than the base models. This hypothesis was tested over two datasets: FEBRL which provided duplicates and links matching original and duplicates, and ePRBN over which random duplicates were created following a Poisson distribution. They also used blocking criteria to reduce computational complexity by removing pairs with low probability to refer to the same entity and created engineered features over which the models were evaluated. As final result, the ensemble method proved to be better than base models on one dataset and on the other it performed almost the same.as the best base model.

On this project we will evaluate the reproducibility of the paper, the methods used and its results. We will be linking different datasets via a variety of techniques, such as blocking, in conjunction with various classification algorithms.

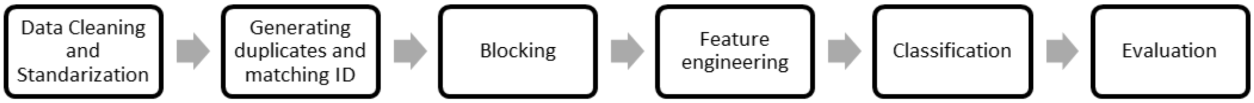
2. Scope of reproducibility

We will test whether an ensemble approach is a better way to address the complex problem of record linkage using three popular supervised machine learning methods: Support Vector Machine, Logistic Regression, Multi-layer Perceptron Neural Network. Then we will test whether the ensemble outperforms base learners in precision, sensitivity, f-score, and number of false matches for both the FEBRL and ePBRN datasets.

3. Methodology

a. Data descriptions

Two synthetic datasets are utilized, Freely Extensible Biomedical Record Linkage (FEBRL) and Electronic Practice Based Research Network (ePBRN). Within FEBRL, we will use FEBRL 3 for training and FEBRL 4 testing. The FEBRL 3 dataset consists of 5000 records, 2000 of which are original and 3000 are duplicate. There is a Zipf distribution of duplicate records ranging from 1 to 5. The FEBRL 4 dataset contains 5000 originals and 5000 duplicates with only 1 duplicate per original. The dataset is designed to be used for testing linkage procedures and thus is an ideal dataset for the study. ePBRN is based on the Australian UNSW Centre for Primary Health Care and Equity. The dataset was built on linkage errors


Data Cleaning and Standarization
Generating duplicates and matching ID
Blocking
Feature engineering
Classification
Evaluation

There are six steps in the pipeline: data cleaning and standardization, generating duplicates and matching ID, blocking, feature engineering, classification, and evaluation. While expansive data cleaning was not required due to some of the work already done on the datasets there are a couple steps. All names needed to be standardized, i.e. lower case and punctuation removed and then tokenized. Next, date of births need to be extracted to day, month, and year for those with a recognizable format with a similar process followed for addresses. Lastly, some basic steps such as establishing data types and addressing null values.

New duplicates were created only for the ePRBN datasets. Diverse types of duplicates were created under diverse scenarios which included swapping of first name and last names, missing or swapping some characters or numbers and more. The selection of the scenario followed a Poisson distribution and an original record can have until 4 duplicates. As improvement, we included the phone number is part of the information given by patients when admitted to a healthcare location. This is only available on ePRBN. We also created duplicates considering wrong area code or swapping of some digit numbers. The final datasets have a match_id column to identify which original and duplicates represent the same patient. As an improvement we fixed the match_id for the FEBRL datasets as we noticed it wasn’t correctly linking original and all duplicates.

Blocking is then used on the clean dataset with the goal of eliminating pairs that are unlikely to be matched. While the study does not use any massive datasets, it correctly points out as the dataset grows there are exponentially more pairs to compare and thus increased computational costs. Essentially blocking divides the records into various blocks, or groups, that have a higher probability of being linked. The blocks used in the study were given name, surname, and postal code (see figure below). Lastly, feature engineering is utilized via phonetic algorithms to convert values into codes.

Number of True Matched Pairs		
Blocking Criteria	Scheme A	Scheme B
Given name	3287/5000	1567/2653
Surname	3325/5000	1480/2653
Postal code	4219/5000	2462/2653

1+ Match	4894/5000	2599/2653

Figure 1: The number of true matched pairs for both schemes. Scheme A represents FEBRL dataset and Scheme B ePBRN

b. Model descriptions

The final output uses an ensemble classification approach utilizing support vector machines, logistic regression, and multi-layer perceptron neural network that then uses bagging and stacking. There is grid search performed for hyperparameter tuning on the C value for both SVM and logistic regression and alpha for the MLP. Once an appropriate set of C and alpha values are established, the algorithms are tuned on the kernel for SVM , penalty for logistic regression, and activation function for MLP.

Implementations			
Algorithm	**Tuned Param 1**	**Tuned Param 2**	**Other parameters**
SVM	Kernel -linear -rbf	C over a range from 0.001 to 5000	degree=3, gamm=’scale’, coef0=0.0, shrinking=True, probability=False, tol=1e-3, cache_size=200, class_weight=None, max_iter=-1, decision_function_shape=’ovr’, break_ties=False
Logistic Regression	Penalty: -l1 -l2	C over a range from 0.001 to 5000	dual=False, tol=1e-4, fit_intercept=True, intercept_scaling=1, class_weight=None, solver=’liblinear’ for l1 and ‘lbfgs’ for l2 penalty, max_iter=5000, multi_class=’ovr’
MLP	Activation: -relu -logistic	alpha over a range from 0.001 to 5000	hidden_layer_sizes=(256,),solver=’lbfgs’, batch_size=’auto’, learning_rate=’constant’, power_t=0.5, max_iter=30000, shuffle=True, tol=0.0001, warm_start=False, momentum=0.9, nesterovs_momentum=True, early_stopping=False, validation_fraction=0.1, beta_1=0.9, beta_2=0.999, epsilon=1e-08, n_iter_no_change=10, max_fun=15000
Random Forest	Criterion: -gini -entropy	ccp_alpha over a range from 0.001 to 5000	n_estimators=100, max_depth=7, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=’sqrt’, max_leaf_nodes=None, bootstrap=True, oob_score=False, warm_start=Fale, class_weight=None, max_samples=None

Figure 2: Parameters of the implementations used. All implementations are from scikit-learn classification algorithms from version 1.1.3

Next, we utilized bagging to address potential overfitting via 10 cross validation folds to lower generation errors on unseen data. The bagged results are then stacked and averaged across the three models to lower bias. The idea being this is one way to approach the bias-variance tradeoff. We tested both a stacked version using SVM, MLP, and Logistic Regression in addition to a stacked implementation that included a Random Forest. We also tested a meta ensemble for scheme B including phone number as we hypothesized this would result in improved results.

c. Computational implementation

All algorithms are from the sklearn package on CPU. Hyperparameters are listed in Figure 2.

d. Code

https://github.com/ian-shepherd/CSE6250_BDH_Project

4. Results

After performing hyper-parameter tuning using parameters in Figure 2, we found the best values for SVM, MLP, LR, and RF on both schemes.

	Scheme A		Scheme B	
Model	**hyper-parameter**	**f-score**	**hyper-parameter**	**f-score**
SVM	Linear kernel with C = 0.002	98.94	Linear kernel with C = 0.001	80.90
LR	Regularization L2 with C = 0.005	99.07	Regularization L2 with C = 1000	81.21
MLP	Relu activation with alpha = 500	99.07	Logistic activation with alpha = 1000	85.37
RF	Entropy criterion with ccp_alpha = 0.5	84.17	Entropy criterion with ccp_alpha = 0.2	6.42

Figure 3: Best f-score of implementations after tuning params 1 and 2 in Figure 2

Then we use those C to run the models by themselves and as base models for bagging and stacking to compare which one achieved a highest matching or lowest false count(false positive + false negative)

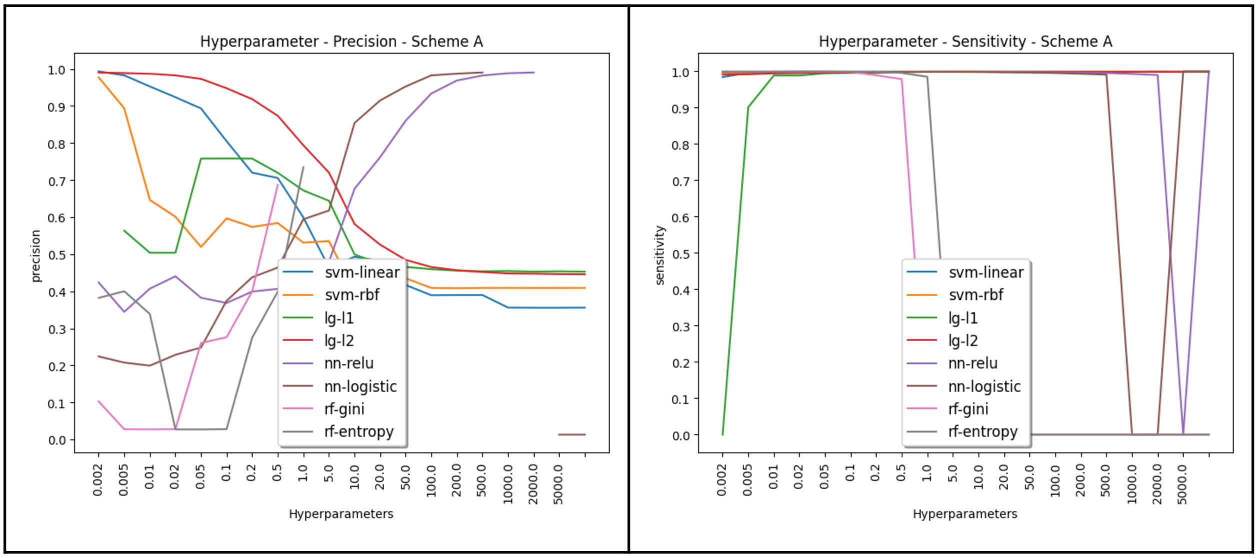
	Scheme A				Scheme B			
	precision	**recall**	**f-score**	**false count**	**precision**	**recall**	**f-score**	**false count**
SVM	98.27	99.63	98.94	597	37.98	99.08	54.91	4266
SVM-bag	98.47	99.63	99.05	94	42.94	98.78	59.86	3474
MLP	98.84	99.31	99.07	91	72.26	97.48	83.01	1047
MLP-bag	98.88	99.20	99.04	94	73.46	97.44	83.77	990
LR	98.70	99.45	99.07	91	72.26	97.48	76.70	1560
LR-bag	98.70	99.45	99.07	91	63.50	97.94	83.77	1530
RF	73.47	98.51	84.17	1814	3.31	99.80	6.29	76,346
RF-bag	67.51	98.51	80.12	2393	0.03	99.81	6.42	76,341

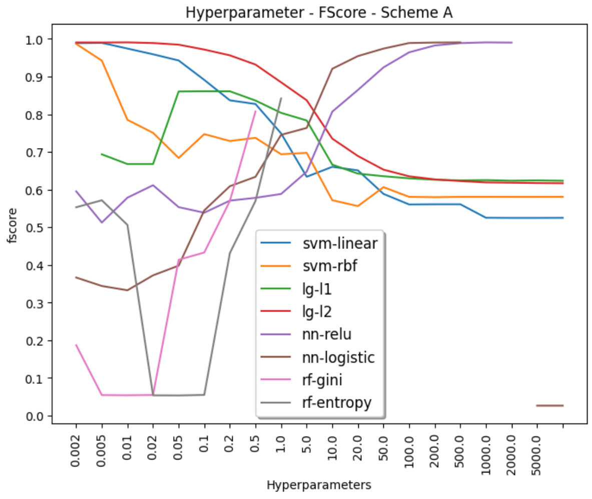
Meta-ensemble	99.16	99.14	99.15	83	76.84	97.33	85.88	839
Meta-ensemble w/ RF	99.81	98.02	98.91	106	78.04	97.33	86.63	788
Meta-ensemble w/ phone number	N/A	N/A	N/A	N/A	92.71	97.80	95.18	265

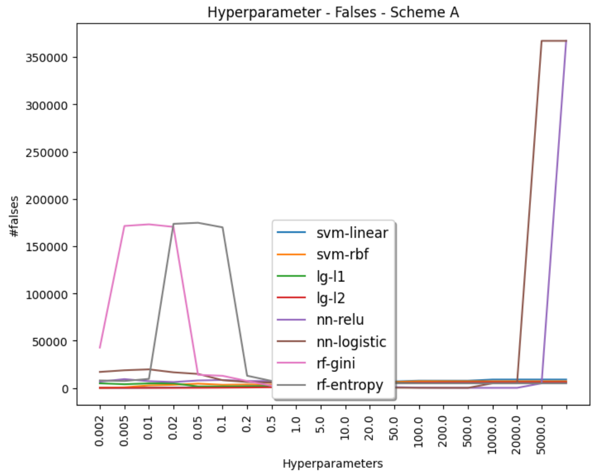
Figure 4: Scores of bagging and stacking implementations

The random forest clearly was the worst performing implementation. Perhaps with more tuning on other parameters the performance could be improved. It is also possible that it is not best suited to the problem at hand. Interestingly enough, the meta ensemble with the random forest performed better than the one without on Scheme B. We did see excellent results across the board on Scheme A, however the meta ensemble both with and without the random forest performed better nonetheless across most metrics. Scheme B did perform better with the meta ensemble, however it is not too dissimilar to the results achieved by the MLP, both bagged and not. However, the meta ensemble is still probably more robust to future data and less likely to be overfit due to the nature of an ensemble.

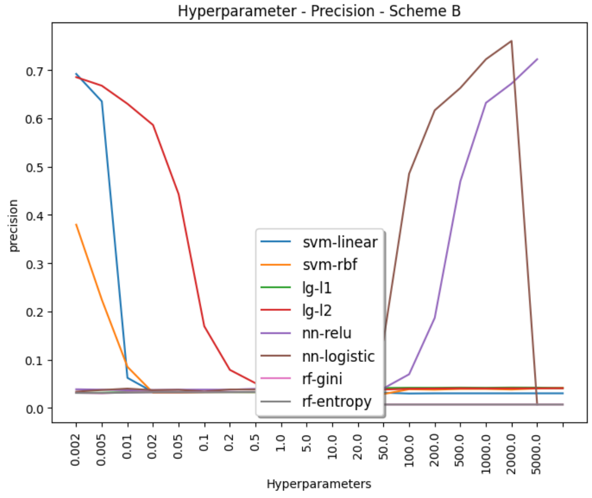
Scheme A

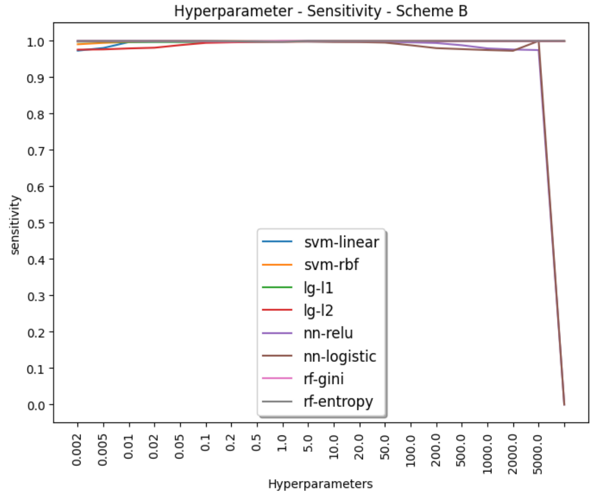


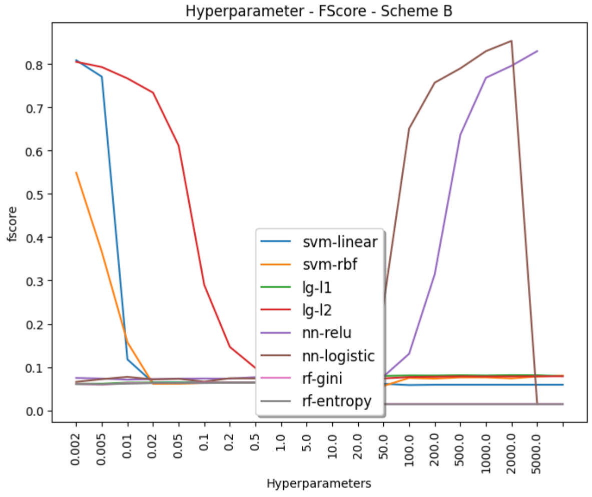

Hyperparameter - FScore - Scheme A

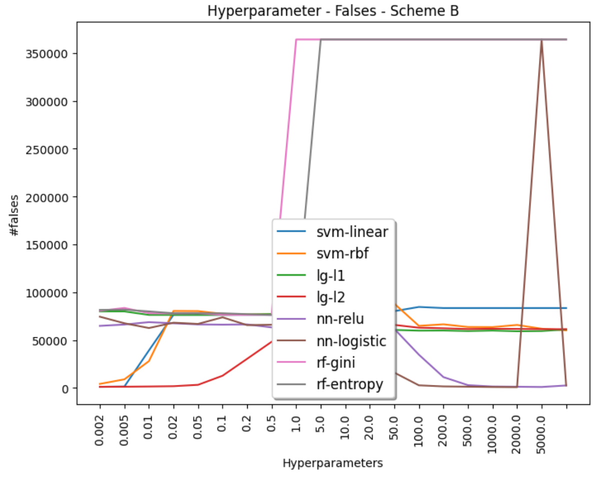

Hyperparameter - Falses - Scheme A

Scheme B


Hyperparameter - Precision - Scheme B


Hyperparameter - Sensitivity - Scheme B


Hyperparameter - FScore - Scheme B


Hyperparameter - Falses - Scheme B

Scheme B with Phone Number

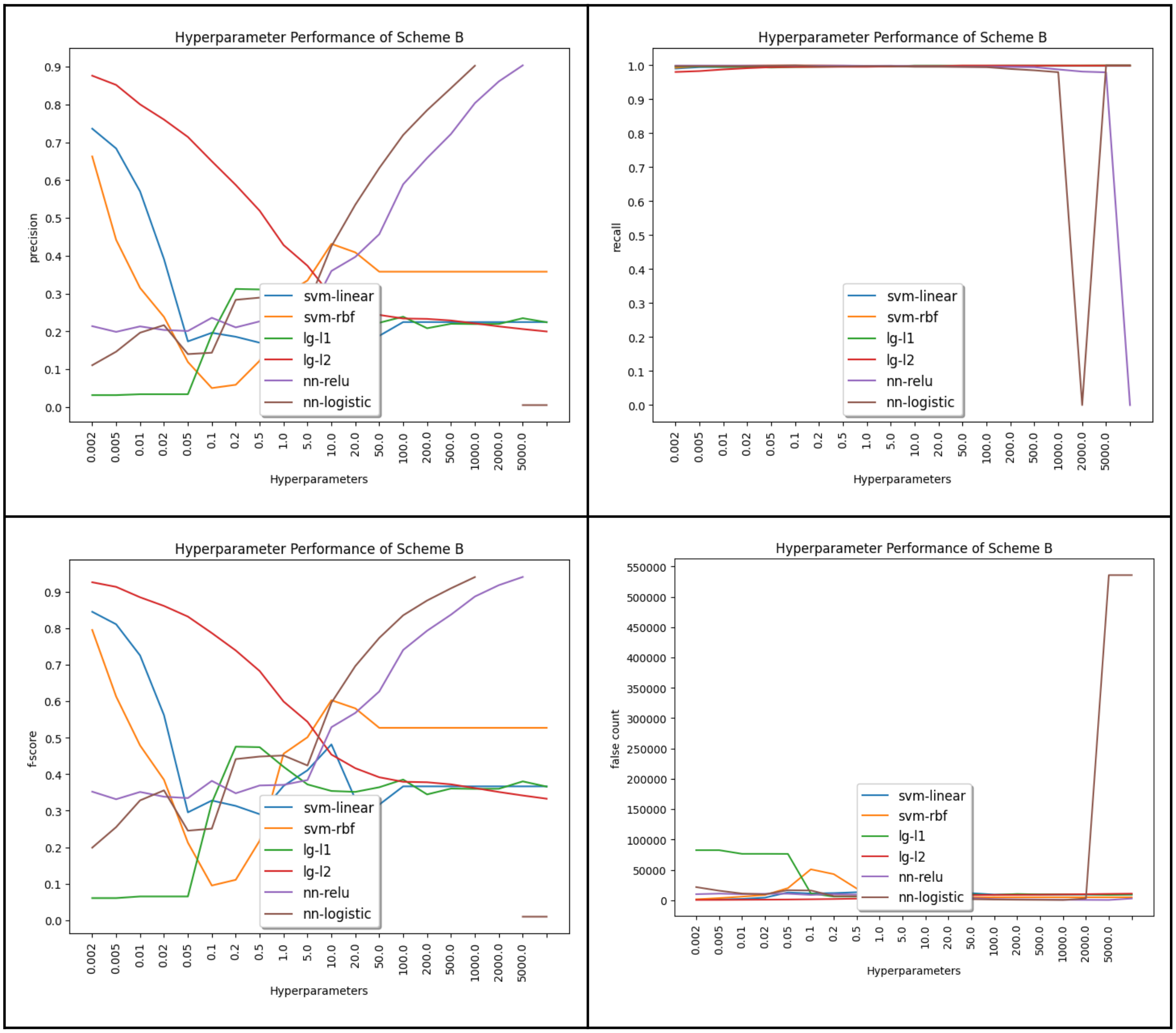


Figure 5: Plots of tuning implementations listed in Figure 2

5. Discussion

We were able to get similar results to the study however we could not exactly match the results. This is likely tied to a few shortcomings in their provided documentation. There are not clearly defined environment requirements. The requirements essentially consist of “install these packages” with no documentation on what versions they used. While not a huge hurdle given the packages used, there was at least one function that has been deprecated which speaks to potential issues with this approach. Secondly, the entire study was run out of a Jupyter notebook with lines commented and uncommented to run certain parts. This matters because the original implementation uses commands like np.choice which are based on random seeds that are never set and can produce different results depending on how often they are run and what order. We even noticed when running their scripts exactly using their data we got slightly different results than their report. Ultimately, the results are similar and the process is reproducible but may not be able to exactly match their results.

The most difficult aspects were tied to the issues above. We spent a lot of time trying to match their results. Even running their provided code did not always match the output of their files. The easier part was we at least had some code to work with. Reading the study without looking at the code first highlighted it would be a little challenging to reproduce their results because they were not always clear what they did and tested. It was a lot easier to spend the time looking over their code and figuring it out from there. Scheme B also had some difficulty due to training time. It was significantly longer than Scheme A due to the blocking on the much larger dataset. That made tuning a more difficult task.

6. Video

https://youtu.be/VY3nSWMaTW4

7. References

[1] Kha Vo, Jitendra Jonnagaddala, Siaw-Teng Liaw, Statistical supervised meta-ensemble algorithm for medical record linkage, Journal of Biomedical Informatics, Volume 95, 2019, 103220, ISSN 1532-0464, https://doi.org/10.1016/j.jbi.2019.103220.

[2] Kevin O'Hare, Anna Jurek-Loughrey, Cassio de Campos, An unsupervised blocking technique for more efficient record linkage, Data & Knowledge Engineering, Volume 122, 2019, Pages 181-195, ISSN 0169-023X, https://doi.org/10.1016/j.datak.2019.06.005.

[3] P. Christen, T. Churches, FEBRL - Freely extensible biomedical record linkage, Available at Febrl - Freely extensible biomedical record linkage (anu.edu.au)

[4] UNSW, The Electronic Practice Based Research Network(ePBRN). Available at The Electronic Practice Based Research Network | Centre for Primary Health Care and Equity (unsw.edu.au)

[5] ReadTheDocs, Python Record Linkage Toolkit. Available at About — Python Record Linkage Toolkit 0.15 documentation