Record Linkage

Cristina Zenteno Garcia, Ian Shepherd

Final Project for course CSE 6250: #Team: 15, Team ID: C5

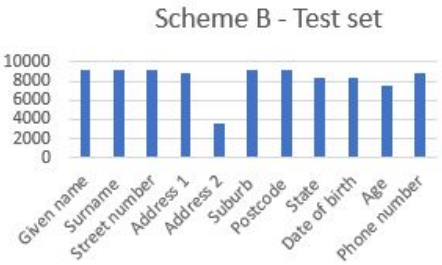
Summary and Scope

Target Paper: Statistical supervised meta-ensemble algorithm for medical record linkage
https://www.sciencedirect.com/science/article/pii/S1532046419301388

- Complex problem that is essential in medical industry
- Links patient data from different sources to provide better healthcare services
- Uses features such as name, birth date, and address
- Paper aims to demonstrate that an ensemble classification approach is better than using single base models such as SVM, Logistic Regression, and Multi-Layer Perceptron
- Tested over two datasets: FEBRL and ePRBN
- Implemented blocking criteria to reduce computational complexity
- Additional hypothesis tested
  - Does adding random forest to meta ensemble improve performance?
  - Does adding phone number to Scheme B improve performance?

Data Descriptions

Datasets	Train	Test
FEBRL: Freely Extensible Biomedical Record Linkage (Scheme A)	2000 originals 3000 duplicates. Total: 5000	5000 originals 5000 duplicates Total: 10000
ePBRN: Electronic Practice Based Research Network (Scheme B)	11100 originals 2993 duplicates. Total: 14093	9250 originals 2493 duplicates Total: 11743



Pipeline

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

String fields: convert to lower case, remove punctuations and tokenized

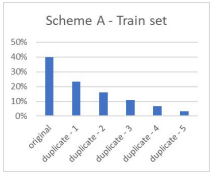
Date of Birth: if recognizable date format then extracted day, month, and year

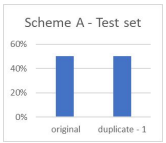
Basic steps such as establishing data types and addressing null values

For the ePRBN sets duplicates were created using 17 types of errors

Each original could have from 1 to up 5 duplicates

A match_ID was assigned to link originals and duplicates





As dataset grows, exponentially more pairs to compare which increases computational costs.

Blocking eliminates pairs unlikely to be matched. Divides records into blocks, or groups, with a higher probability of being linked.

Candidate 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

Use of phonetic algorithms Soundex and Nysiis were used to convert values into codes and generate new attributes

Each scheme has a different set of engineered features using the Jaro–Winkler and Levenshtein similarity metrics.

Grid search performed for hyperparameter tuning on the C value for both SVM and logistic regression, alpha for the MLP, and CCP Alpha for RF

Base models: SVM, Logistic Regression, MLP, Random Forest

Meta Ensemble method: Combination of Bagging and Stacking

The metrics used to evaluate performance of the models are: F-score, recall, precision, and false counts(false positives + false negatives)

Model Descriptions

Tuning hyperparameters on base models using grid search

Algorithm	Tuned Param 1	Tuned Param 2	Other parameters
SVM	Kernel: linear, rbf	C over a range from 0.001 to 5000	degree=3
Logistic Regression	Penalty: L1, L2	C over a range from 0.001 to 5000	solver='liblinear' for l1 and 'lbfgs' for l2 penalty, max_iter=5000
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'
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

	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

Meta Ensemble Method: Bagging & Stacking

Once the best hyper-parameters were selected 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 fourth models to lower bias.

Results

Classification metrics

$$\text{Fscore} = \frac{2 * \text{precision} * \text{recall}}{\text{precision} + \text{recall}} \qquad \text{precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}} \qquad \text{recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}} \qquad \text{false count} = \text{false positives} + \text{false negatives}$$

	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

- 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.

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
  - No documentation on what libraries versions they used
  - Deprecated function was detected
- The entire study was run out of a Jupyter notebook with lines commented and uncommented to run certain parts
  - Original implementation uses commands like np.choice which are based on random seeds that are never set
  - Produces different results depending on how often they are run and what order
  - Even running their scripts produced slightly different results
- 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.