Group 2

04-25-2024

APANPS5210: PYTHON FOR DATA ANALYSIS TEAM PROJECT

Team Members:

Sixuan Li, Xueni Wang, Margaret Ma, Xinyi Yu, Riley Xiong, Shaoze Li

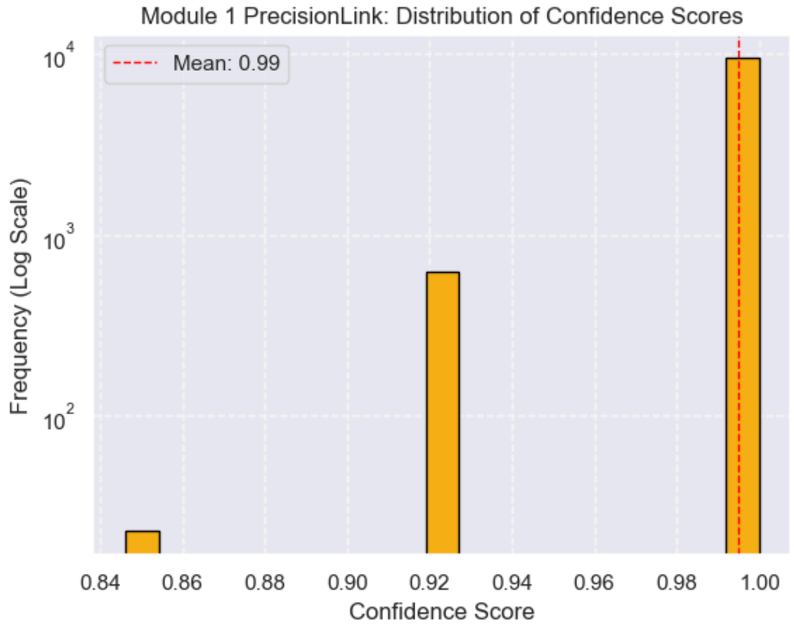
Github Repo:

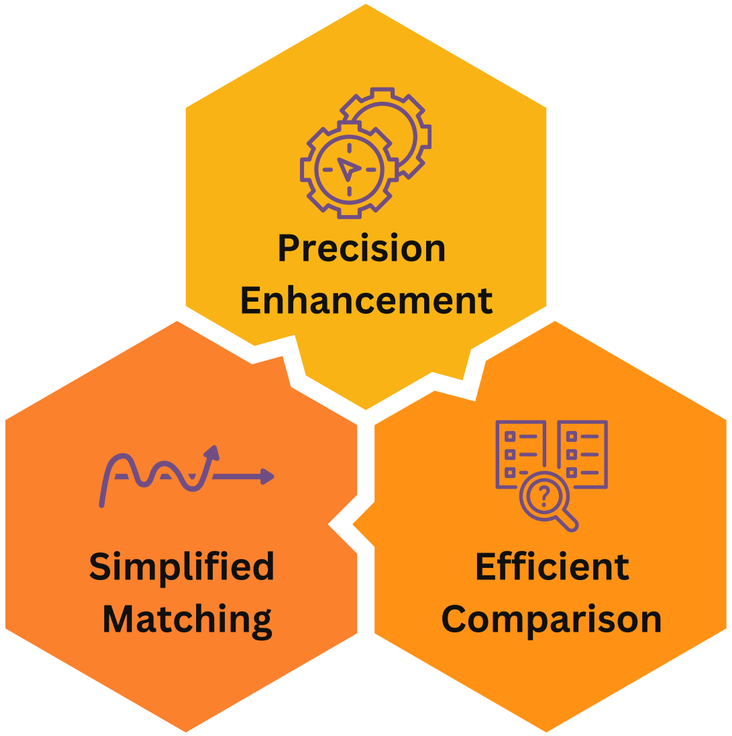
https://github.com/educated-fool/entity-resolution-group2

RecordLinkage

Module 1: PrecisionLink

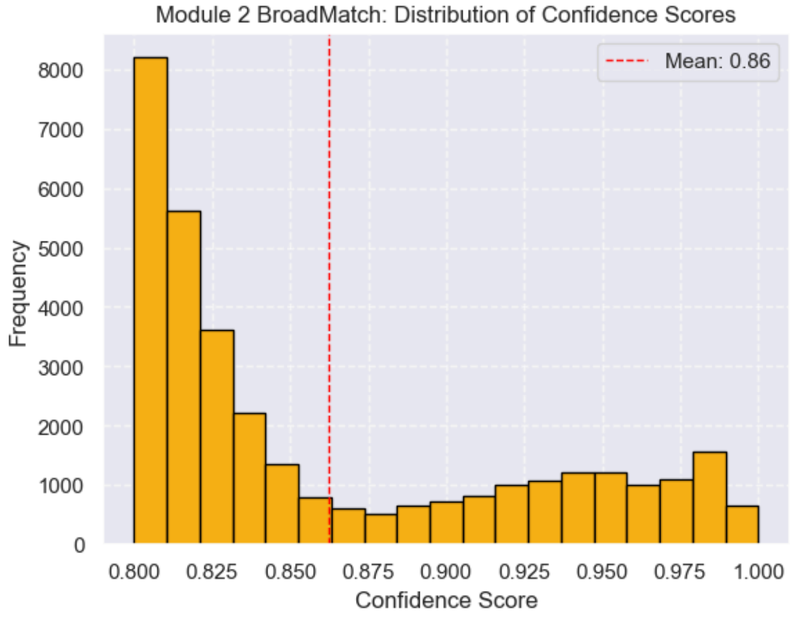
- ***Purpose***: Enhances **precision** in match identification.
- ***Weighted Scoring***:
  - **Different weights** for each field based on importance.
  - Focuses on key fields to improve match accuracy.
- ***Jaro-Winkler Thresholding***:
  - Uses thresholds to refine match quality.
  - **Filters out low-quality matches** effectively.





Module 2: BroadMatch

- ***Purpose***: Increases **recall**, capturing more potential matches.
- ***Full Address Matching***:
  - Consolidates address components into one field.
  - Reduces complexity in comparison process.
- ***Uniform Scoring***:
  - **Equal weighting** across all compared fields.
  - Averages scores for broader inclusivity.
- ***Comprehensive Comparison***:
  - No thresholding, includes more candidates.
  - **Higher match count**, but risk of false positives increased.



Sixuan Li

TheFuzz

It uses Levenshtein Distance to calculate the differences between sequences in a simple-to-use package.

Purpose

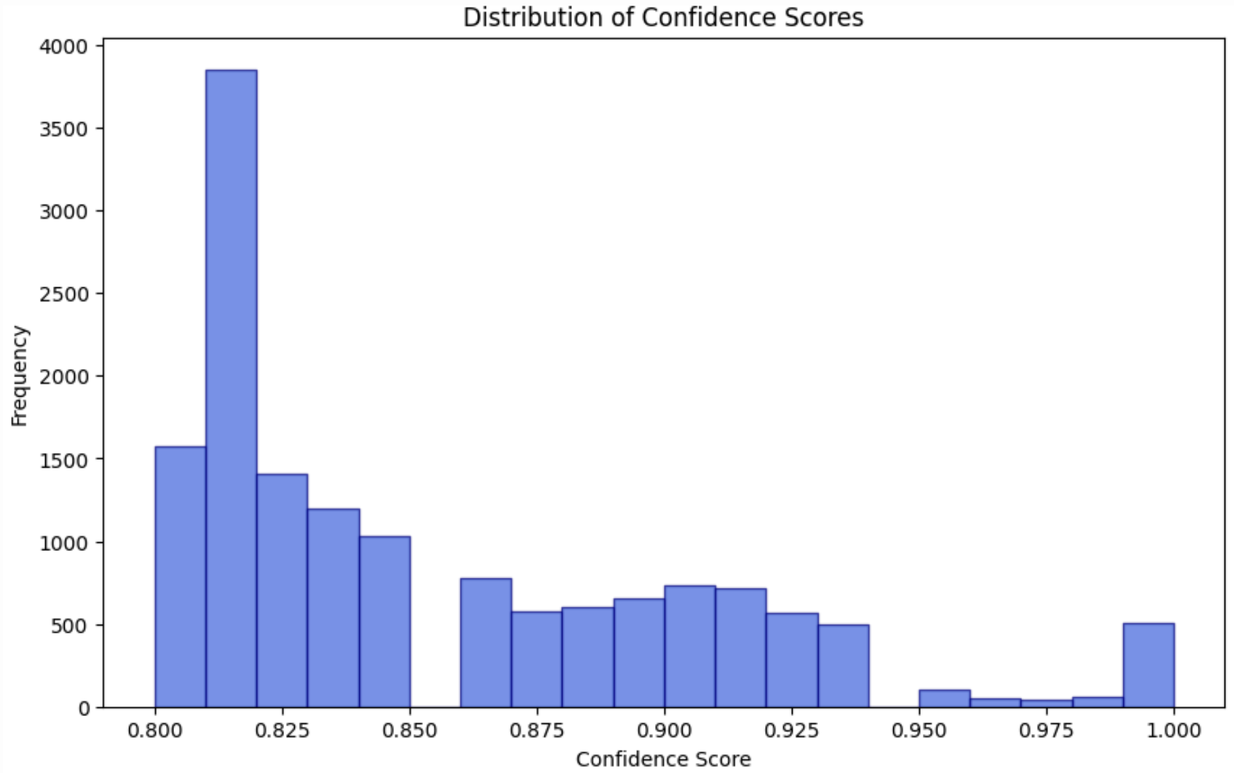
- Maximize the accuracy.
- Practicable size for code running.

Data Preprocessing

- Fill NA with space.
- Transfer all features into string.
- Replace with lowercase.

Matching Method

- Setting block key.
- Combine all features needed.
- process.extractOne()



Output Distribution

Contain 14932 rows in total. Histogram above is a simple distribution for the output.

Xueni Wang

`fnmatch` + `textdistance`

Blocking

Cleaning & Blocking

Reduce the dataset by creating a blocking key column and join the left and right dataframes based on the blocking key.

Exact

Exact Match

name_left = name_right

Score = 1

Wild card

Fnmatch Wildcard Match

fnmatch.fnmatch(name_left, f"*{name_right}*") or fnmatch.fnmatch(name_right, f"*{name_left}*")

Score = text.distance.jaro_winkler * Boost (1.2)

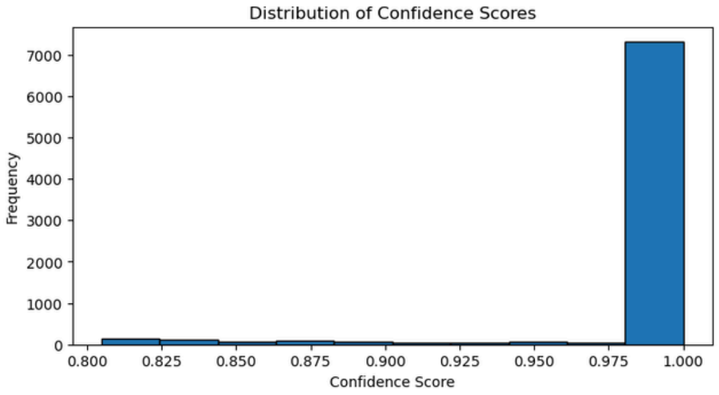
No

No fnmatch Wildcard Match

Score = text.distance.jaccard

Jaro_winkler:
Best for situations with minor typos, but overall high similarity

Jaccard:
Doesn't care about the order, useful for unstructured comparisons



	entity_id	business_id	confidence_score
401	15046	49262	1.0
848	35383	45562	1.0
1965	65788	44707	1.0
2204	74041	39637	1.0
2444	82635	50523	1.0
...	...	...	...
6845645	89251	1044	1.0
6845647	89702	50722	1.0
6845648	89702	50729	1.0
6845653	91826	58043	1.0
6845664	91387	72251	1.0

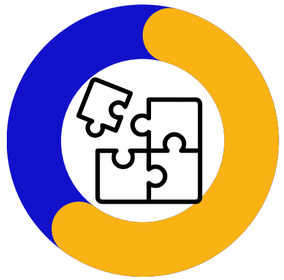
8003 rows × 3 columns

Margaret Ma

Dedupe

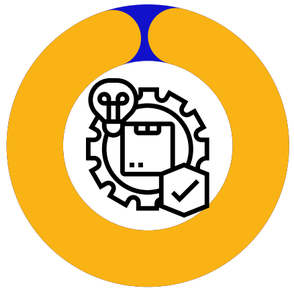
Data Cleaning

- Replace the missing value
- Cleaning and standardizing all columns



Data Matching

- Converting data frames into dictionaries
- Set up Dedupe, including fields, RecordLink
- Training data by manually labeling (Best: 10/10 positive & negative samples)



Scoring and Output

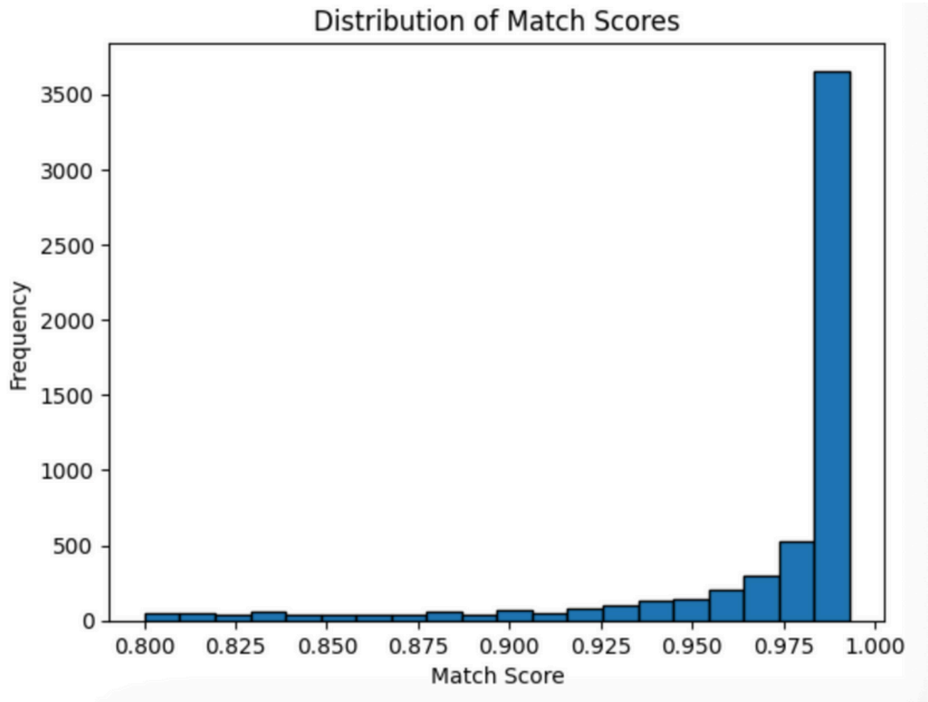
- 5714 records matches (depend on different training)
- High confidence score

Manually Labeling

```
name : new hong kong chinese restaurant
address : 10801 starkey rd ste 5
city : seminole
state : FL
postal_code : 33777

name : newhongkong chinese restaurant
address : 10801 starkey rd 5
city : largo
state : FL
postal_code : 33777

7/10 positive, 3/10 negative
Do these records refer to the same thing?
(y)es / (n)o / (u)nsure / (f)inished / (p)revious
 y
```



Xinyi Yu

Difflib

Step 1:

Get SequenceMatcher from the package.

Step 3

Calculate and filter.

Step 2

Clean data to reduce rows.

Step 4

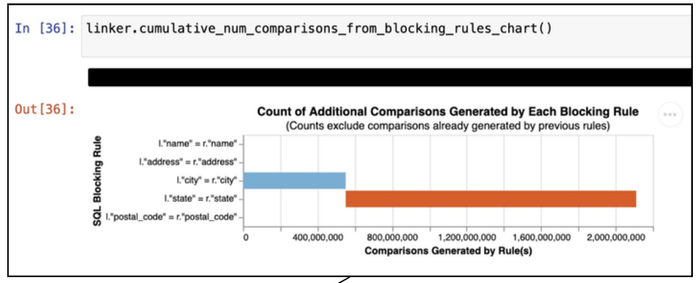
Finally, get the result, show the table.

	entity_id	business_id	confidence
645	82635	50523	0.941860
646	82635	50540	0.931818
648	89051	37027	0.960000
799	24054	78876	0.916667
895	14584	39665	0.867347
...	...	...	...
1133852	93910	18515	0.833333
1133858	94035	72787	0.944444
1133862	94065	81979	0.865385
1133865	94131	27269	0.925000
1133896	94546	83310	0.937500

[7534 rows x 3 columns]

Shaozhe Li

Splink



Advantage:

- Comprehensive Visualization Tools
- **Advanced Probabilistic Matching**
- **Scalable SQL Integration**

Output: The results never ran out

Attempt 1:
Increase Matching Threshold

Increase matching threshold from 0.9 to 0.98:
The system remains overloaded due to the large dataset size.

Attempt 2:
Subset Dataset Size

- Reduced to 5000 rows: too limited for training
- Increased to 10000 rows: overwhelmed the system.

Learnings/Future Actions:

- Complex setup of **comparison functions** and decision rules
- Requires a demanding operational environment, in support **SQL integration** and high computer hardware capabilities

Riley Xiong

THANKS FOR WATCHING