

# ACM SIGMOD Programming Contest 2021

Team: BoomBoomChicken

Data Curation Lab - Database Group @ Rutgers University

Chaoji Zuo      Advisor: Dong Deng

chaoji.zuo@rutgers.edu, dong.deng@rutgers.edu



## 1. Task Overview

**Task:** Identifying instances which representing the same real-world entities.

**Training Dataset:**

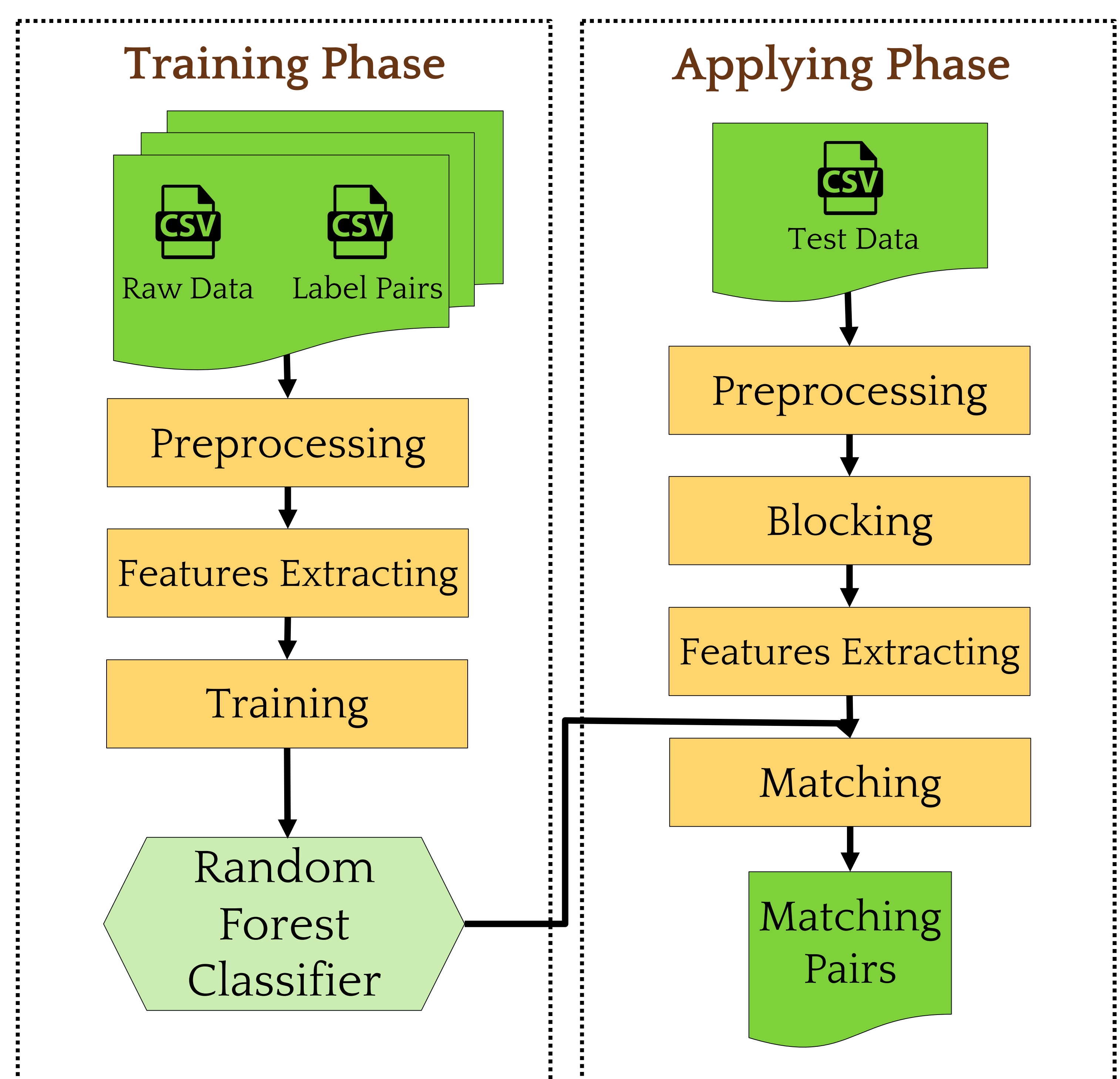
| #         | Entity Type   | Instances | Attributes | Entities |
|-----------|---------------|-----------|------------|----------|
| Dataset 1 | Notebook      | 538       | 14         | 100      |
| Dataset 2 |               | 605       |            | 153      |
| Dataset 3 | Storage Media | 1356      | 5          | 193      |

Dataset 1 & 2 come from the same source with no overlap.

**Measurement:** Average F-measure over 3 datasets.

**Evaluation Environment:** 32 CPU x 2.1 Ghz Processors, 64 Gb Main Memory, 2TB Storage.

## 2. System Architecture



Our system consists of two phases: **Training** phase trains binary random forest classifiers using the label data and **Applying** phase applies matchers on testing dataset.

Among each phase, we have similar **Preprocessing** and **Features Extracting** components. Besides, for Dataset 3, it includes another **Blocking** component in the training phase to augment training data.

## 3. Preprocessing

**Data Cleaning:** Fix typos, synonyms substitution. (e.g., “clase” → “class”)

**Formatting:** Adjust the text to the same format, lowercase letters, replace punctuations. (e.g., “USB 3” → “usb3”)

**Attributes Refining:** Extract keywords, refine strings to attributes. (e.g., “core i5 560m / 2.66 ghz” → “i5 560m”).

**Keywords Extraction:** Leverage TF-IDF and regular expression to filter keywords for attribute refining.

## 4. Blocking & Features Extracting

**Blocking:** Group the instances by specific features. (e.g., “Brand” for dataset 1 & 2 and “(Brand, Size)” for dataset 3.) Use the blocked groups to generate candidate pairs.

**Features Extracting:** Calculate several similarities (e.g., Jaccard similarity, Levenshtein distance) of corresponding attributes between candidate pairs.

| #         | Attributes | Refined Attributes | Features | Groups |
|-----------|------------|--------------------|----------|--------|
| Dataset 1 | 14         | 9                  | 50       | 14     |
| Dataset 2 |            |                    |          |        |
| Dataset 3 | 5          | 8                  | 42       | 56     |

## 5. Matcher

The core model of our system is a **Binary Random Forest Classifier**, which is trained by labeled pairs of instances using corresponding extracted features.

For dataset 1 & 2, we train the model directly using the label data and the classifiers contain 120 trees. For dataset 3, we train several classifiers for different brands independently to increase the granularity of classifying, each of classifier contains 50 trees.

## 6. Result

| #         | Precision | Recall | F1    |
|-----------|-----------|--------|-------|
| Dataset 1 | 0.925     | 0.976  | 0.950 |
| Dataset 2 | 0.987     | 0.976  | 0.982 |
| Dataset 3 | 0.932     | 0.822  | 0.874 |

Average F-measure over 3 dataset: **0.935**

## 7. Conclusion & Acknowledgement

- Random Forest is an effective model for structured entity matching. However, when the training dataset is limited, it might get overfitting. Data augmentation is a valid way to help train a robust model.
- Preprocessing is vital for key textual attributes like Model and Brand. These attributes influence the matching heavily.
- We leverage python package py\_entitymatching from Magellan project to extract features and train random forests.