Module X: Blocking

Rebecca C. Steorts

Reading

- Binette and Steorts (2020)
- Steorts, Ventura, Sadinle, Fienberg (2014)
- Murray (2016)

Agenda

- Data Cleaning Pipeline
- Blocking
- Traditional Blocking
- Probabilistic Blocking

Load R packages

```
knitr::opts_chunk$set(echo = TRUE, fig.width=4, fig.height=
library(RecordLinkage)
library(blink)
```

Data Cleaning Pipeline

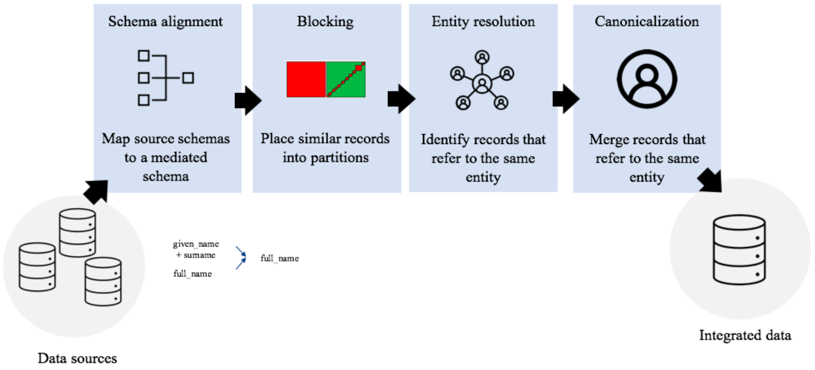


Figure 1: Data cleaning pipeline.

Blocking

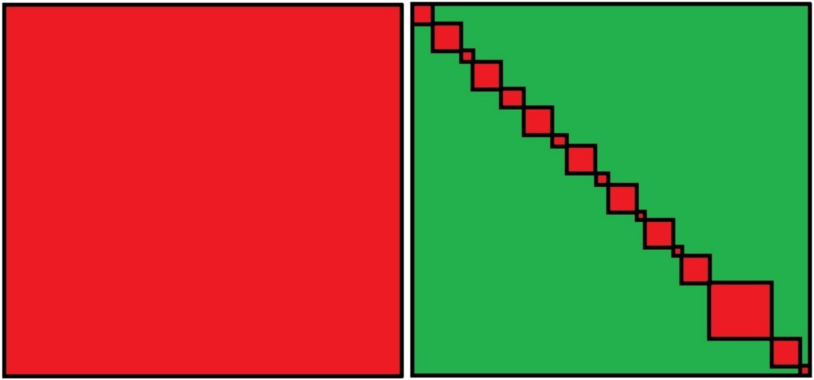


Figure 2: Left: All to all record comparison. Right: Example of resulting blocking partitions.

Blocking

- Blocking partitions similar records into partitions/blocks.
- ER is only performed within each blocks.

Traditional Blocking

- A deterministic (fixed) partition is formed based upon the data.
- A partition is created by treating certain fields that are thought to be nearly error-free as fixed.
- Benefits: simple, easy to understand, and fast to implement.
- Downsides: the blocks are treated as error free, which is not usually accurate and can lead to errors in the ER task that cannot be accounted for.

Example: Blocking on date of birth year.

Probabilistic Blocking

- A probability model is used to cluster the data into blocks/partitions.

Example: Fellegi-Sunter (1969), or Locality Sensitive Hashing

Under both blocking approaches, record pairs that do not meet the blocking criteria are automatically classified as non-matches.

Example: Traditional blocking

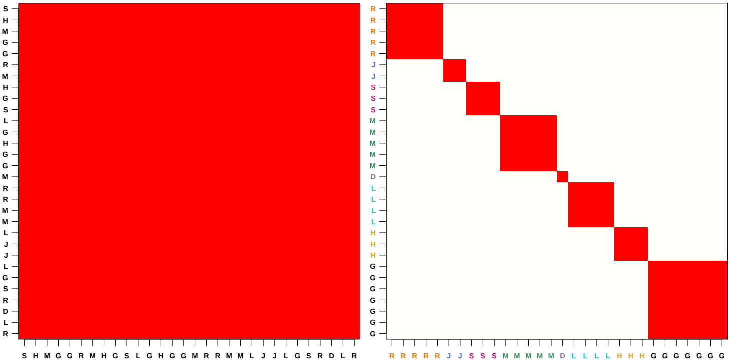


Figure 3: All-to-all record comparisons (left) versus partitioning records into blocks by lastname initial and comparing records only within each partition (right).

Example: RLdata500

```r
library(RecordLinkage)
data(RLdata500)
head(RLdata500)
```

```
##   fname_c1 fname_c2 lname_c1 lname_c2   by bm bd
## 1  CARSTEN     <NA>    MEIER     <NA> 1949  7 22
## 2     GERD     <NA>    BAUER     <NA> 1968  7 27
## 3   ROBERT     <NA> HARTMANN     <NA> 1930  4 30
## 4   STEFAN     <NA>    WOLFF     <NA> 1957  9  2
## 5     RALF     <NA>  KRUEGER     <NA> 1966  1 13
## 6  JUERGEN     <NA>   FRANKE     <NA> 1929  7  4
```

RLdata500 (Continued)

```r
# Total number of all to all record comparisons
choose(500,2)
```

```
## [1] 124750
```

RLdata500 (Continued)

```
# Block by last name initial
last_init <- substr(RLdata500[,"lname_c1"], 1, 1)
head(last_init)
```

```
## [1] "M" "B" "H" "W" "K" "F"
```

```
# Total number of blocks
length(unique(last_init))
```

```
## [1] 20
```

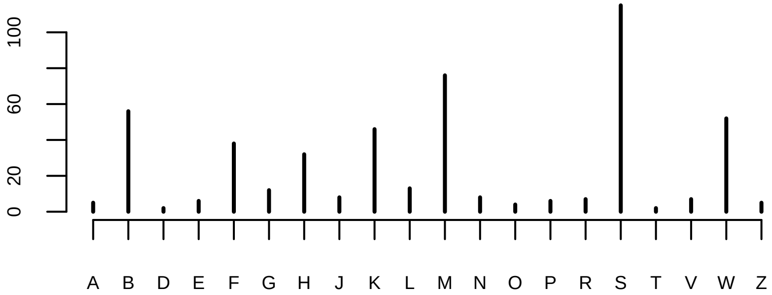
RLdata500 (Continued)

```r
# Total number of records per block
recordsPerBlock <- table(last_init)
head(recordsPerBlock)
```

```
## last_init
##  A  B  D  E  F  G
##  5 56  2  6 38 12
```

RLdata500 (Continued)

```r
# Block sizes can vary
plot(recordsPerBlock,
     cex.axis=0.6, xlab="", ylab="")
```



RLdata500 (Continued)

```
# Total number of records pairs per block
choose(recordsPerBlock, 2)
```

```
## last_init
##    A    B    D    E    F    G    H    J    K    L    M
##   10 1540    1   15  703   66  496   28 1035   78 2850
##    T    V    W    Z
##    1   21 1326   10
```

```
# Reduction on comparison space
sum(choose(recordsPerBlock, 2))
```

```
## [1] 14805
```

RLdata500 (Continued)

What is the overall dimension reduction form the original space to the reduced space induced by blocking?

Recall the original space of comparisons was

```r
choose(500, 2)
```

```
## [1] 124750
```

We have reduced the number of comparisons to

```r
sum(choose(recordsPerBlock, 2))
```

```
## [1] 14805
```

How do we calculate the reducation ratio?

The reduction ratio is

$$\text{RR} = \% \text{ comparisons eliminated by blocking.}$$

```
(choose(500, 2) - sum(choose(recordsPerBlock, 2))) /
  choose(500, 2)
```

```
## [1] 0.8813226
```

How do we calculate the reducation ratio?

In a function:

```
reduction.ratio <- function(block.labels) {
  n_all_comp = choose(length(block.labels), 2)
  n_block_comp = sum(choose(table(block.labels), 2))

  (n_all_comp - n_block_comp) / n_all_comp
}

reduction.ratio(last_init)
```

```
## [1] 0.8813226
```

Pairwise Evaluation Metrics

Precision

```r
labels = unique(last_init)

# Number of matching pairs among blocks
n_matches = sapply(labels, function(label) {
  # Records in a given blocks
  records = which(last_init == label)
  # Number of matches in that block
  sum(duplicated(identity.RLdata500[records]))
})

# Total number of pairs
n_pairs = sum(choose(table(last_init), 2))

sum(n_matches) / n_pairs
```

```
## [1] 0.003377237
```

Pairwise Evaluation Metrics

```r
precision <- function(block.labels, IDs) {
  labels = unique(block.labels)

  # Number of matching pairs among blocks
  n_matches = sapply(labels, function(label) {
    records = which(block.labels == label)
    sum(duplicated(IDs[records]))
  })

  # Total number of pairs
  n_pairs = sum(choose(table(block.labels), 2))

  sum(n_matches) / n_pairs
}

precision(last_init, identity.RLdata500)
```

```
## [1] 0.003377237
```

Pairwise Evaluation Metrics

Recall

```
recall <- function(block.labels, IDs) {
  precision(IDs, block.labels)
}
```

Case Study to El Salvador

We return to the case study on El Salvador, where we will investigate deterministic blocking as done in Sadinle (2014).

Task 1

Implement the blocking procedure from Sadinle (2014), where the blocking criterion is XXX.

Task 2

Explain why you think the author choose this blocking criterion.

Task 3

What is the reduction ratio, precision, and recall assuming that the ground truth is true in this situation?

Task 4

Can you come up with a better blocking criterion for this data set that is deterministic?