

# Module 4: Deterministic Blocking

Rebecca C. Steorts

joint with Olivier Binette

# 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=3,  
                        fig.align="center")  
  
library(RecordLinkage)  
library(blink)  
library(italy)  
library(tidyverse)
```

# Data Cleaning Pipeline



Figure 1: Data cleaning pipeline.

# Blocking

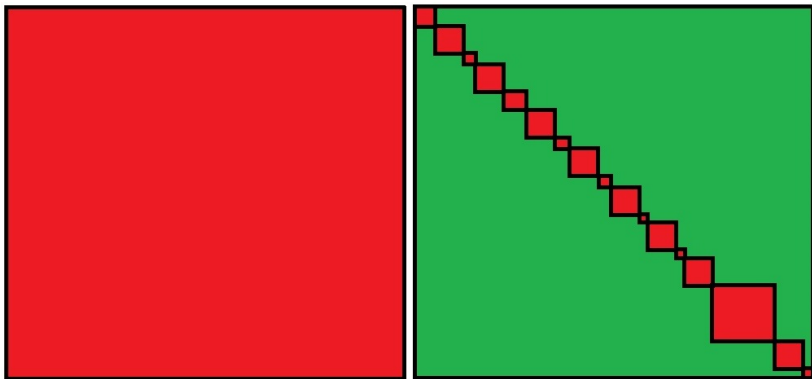


Figure 2: Left: All to all record comparison. Right: Example of resulting blocking partitions.

# 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

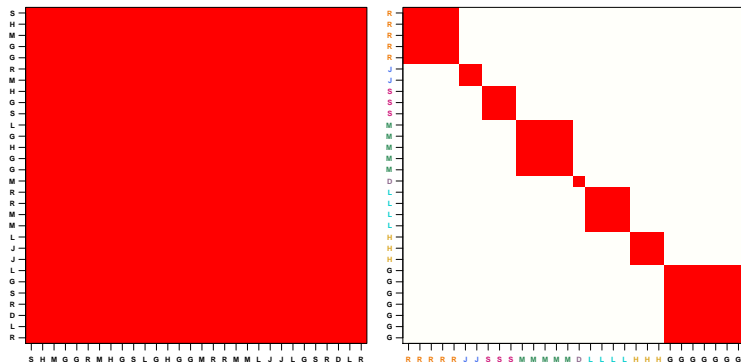


Figure 3: All-to-all record comparisons (left) versus partitioning records into blocks by lastname initial and comparing records only within each partition (right).

## Example: RLdata500

```
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)

```
# 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
```

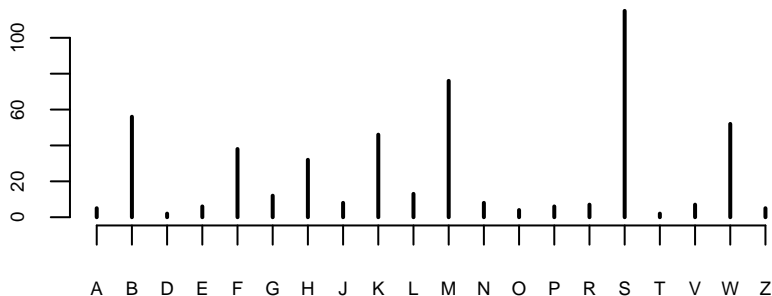
## RLdata500 (Continued)

```
# 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)

```
# 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 from the original space to the reduced space induced by blocking?

Recall the original space of comparisons was

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

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

We have reduced the number of comparisons to

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

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

# How do we calculate the reduction ratio?

The reduction ratio is

RR = % comparisons eliminated by blocking.

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

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

## How do we calculate the reduction ratio (via 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 Precision

```
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 Precision

```
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 Recall

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

# Italian Survey on Household and Wealth (SHIW)

We will now explore a case study to the SHIW

# SHIW

- ▶ The Italian Survey on Household and Wealth (SHIW) is a sample survey 383 households conducted by the Bank of Italy every two years (2008 and 2010).
- ▶ The data set is anonymized to remove first and last name (and other sensitive information).



# SHIW

The following attribute information is available:

- ▶ PARENT (parental status)
- ▶ GENDER
- ▶ ANASC (year of birth)
- ▶ NASCREG (working status)
- ▶ CIT (employment status)
- ▶ ACOM4C (branch of activity)
- ▶ STUDIO (town size)
- ▶ Q (quality of life status)
- ▶ QUAL (whether or not Italian national)
- ▶ SETT (highest educational level obtained)
- ▶ IREG (region of Italy)

# Explore Data

```
head(italy08) # first year of SHIW
```

##	id	PARENT	SEX	ANASC	NASCREG	CIT	ACOM4C	STUDIO	Q	QUAL	SETT	IREG
## 1	1040021	1	2	1948	16	1	0	5	1	2	3	16
## 2	1040022	10	2	1952	16	1	0	7	1	2	3	16
## 3	1110521	1	1	1972	20	1	2	5	1	1	4	20
## 4	1110522	3	1	1935	20	1	2	2	3	6	5	20
## 5	1110523	3	2	1941	20	1	2	3	3	6	5	20
## 6	119401	1	1	1941	7	1	0	4	3	6	5	7

# Explore Data

```
head(italy08) # second year of SHIW
```

##	id	PARENT	SEX	ANASC	NASCREG	CIT	ACOM4C	STUDIO	Q	QUAL	SETT	IREG
## 1	1040021	1	2	1948	16	1	0	5	1	2	3	16
## 2	1040022	10	2	1952	16	1	0	7	1	2	3	16
## 3	1110521	1	1	1972	20	1	2	5	1	1	4	20
## 4	1110522	3	1	1935	20	1	2	2	3	6	5	20
## 5	1110523	3	2	1941	20	1	2	3	3	6	5	20
## 6	119401	1	1	1941	7	1	0	4	3	6	5	7

# Reformat Data

```
id08 <- italy08$id
id10 <- italy10$id
id <- c(italy08$id, italy10$id) # combine the id
italy08 <- italy08[-c(1)] # remove the id
italy10 <- italy10[-c(1)] # remove the id
italy <- rbind(italy08, italy10)
head(italy)
```

##	PARENT	SEX	ANASC	NASCREG	CIT	ACOM4C	STUDIO	Q	QUAL	SETT	IREG
## 1	1	2	1948	16	1	0	5	1	2	3	16
## 2	10	2	1952	16	1	0	7	1	2	3	16
## 3	1	1	1972	20	1	2	5	1	1	4	20
## 4	3	1	1935	20	1	2	2	3	6	5	20
## 5	3	2	1941	20	1	2	3	3	6	5	20
## 6	1	1	1941	7	1	0	4	3	6	5	7

## Your turn

- ▶ Construct a blocking criterion for the SHIW data set.
- ▶ Provide code to construct the blocks
- ▶ Are your blocks well balanced?
- ▶ What is the reduction ratio?
- ▶ What is the pairwise recall and precision?
- ▶ Would you recommend your blocking criterion for an ER task?  
Why or why not.

Hint: You might consider blocking on gender, regions (in Italy), or combinations of these. What do you find?

## Your turn solution

Let's block on gender.

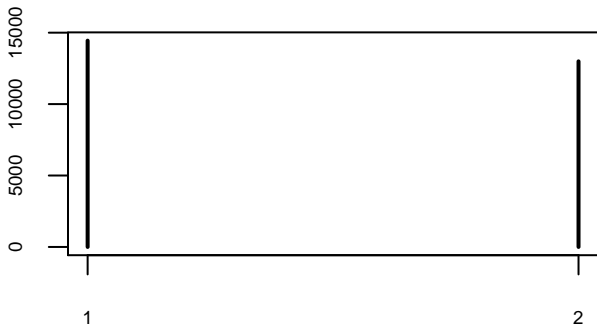
```
# block by gender  
blockByGender <- italy$SEX  
recordsPerBlock <- table(blockByGender)  
head(recordsPerBlock)
```

```
## blockByGender  
##      1      2  
## 14442 12993
```

## Your turn solution

The block sizes are similar. But note, they are still quite large.

```
# Checking block sizes  
plot(recordsPerBlock,  
      cex.axis=0.6, xlab="", ylab="")
```



## Your turn solution

```
print(rr <- reduction.ratio(blockByGender))
```

```
## [1] 0.4986234
```

We have reduced the overall space by roughly 50 percent.



## Your turn solution

```
precision(blockByGender, id)
```

```
## [1] 3.599727e-05
```

```
recall(blockByGender, id)
```

```
## [1] 0.9113109
```

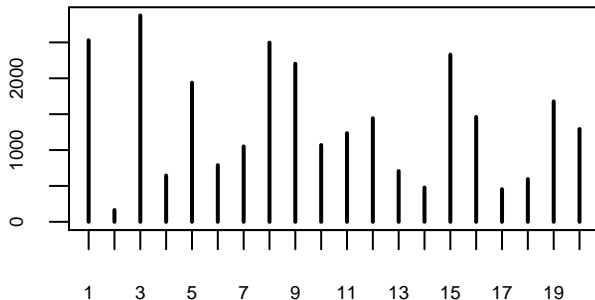
This is not an optimal blocking criterion as ideally, we would want both the precision and recall to be close to 1.

# Your turn solution

Let's block on the twenty regions in Italy.

```
blockRule <- (italy$IREG)
(recordsPerBlock <- table(blockRule))
```

```
## blockRule
##    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16
## 2530  164 2875  645 1938  789 1050 2497 2203 1070 1235 1443  707  478 2329 1461
##   17   18   19   20
##  455  595 1678 1293
plot(recordsPerBlock,
     cex.axis=0.6, xlab="", ylab="")
```



```
print(rr <- reduction.ratio(blockRule))
```

```
## [1] 0.9339148
precision(blockRule, id)
```

## Your turn solution

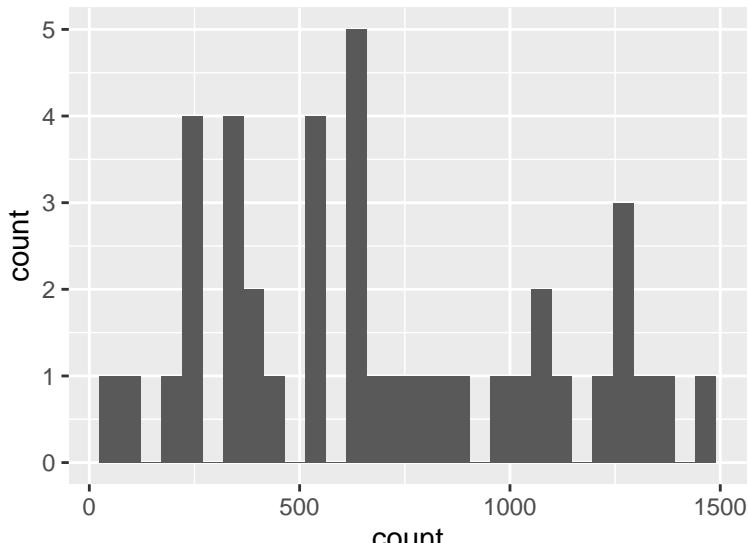
Let's block on a combination of gender and region.

Trying to make this work using tidyverse as it seems more elegant.

The histogram is not printing now for some reason.

## Your turn solution

```
italy %>%  
  group_by(italy$IREG, italy$SEX) %>%  
  summarise(count = n()) %>%  
  group_by(count) %>%  
  ggplot() +  
  geom_histogram(aes(count))
```



## Your turn solution

```
recordsPerBlock <- italy %>%  
group_by(IREG, SEX) %>%  
summarise(count = n(), .groups="drop") %>%  
group_by(count) %>%  
ggplot() +  
geom_histogram(aes(count))  
head(recordsPerBlock)
```

```
## $data  
## # A tibble: 40 x 4  
## # Groups:   count [40]  
##       IREG  SEX count .group  
##   <int> <int> <int> <int>  
## 1     1     1  1282     37  
## 2     1     2  1248     35  
## 3     2     1    91      2  
## 4     2     2    73      1  
## 5     3     1  1489     40  
## 6     3     2  1386     39  
## 7     4     1   324      9  
## 8     4     2   321      8  
## 9     5     1  1056     31  
## 10    5     2   882     28  
## # ... with 30 more rows  
##  
## $layers  
## $layers[[1]]  
## mapping: x = ~count  
## geom_bar: na.rm = FALSE, orientation = NA  
## stat_bin: binwidth = NULL, bins = NULL, na.rm = FALSE, orientation = NA, pad = FALSE  
## position_stack  
##  
##  
## $scales  
## <ggproto object: Class ScalesList, gg>  
##   add: function
```

# Your turn solution

```
grouping <- group_by(italy, italy$IREG, italy$SEX)
recordsPerBlock <- summarise(grouping, count=n())
```

```
## `summarise()` regrouping output by 'italy$IREG' (override with ``.groups` argument)
blockIDs = paste0(italy$IREG, italy$SEX, sep="_")
table(recordsPerBlock$count)
```

```
##
##  73  91  209  229  246  249  262  321  324  333  339  368  371  418  517  533
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## 534 536 611 615 624 634 659 671 709 772 846 882 969 1019 1056 1079
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## 1124 1243 1248 1254 1282 1310 1386 1489
##   1   1   1   1   1   1   1   1
```

```
print(rr <- reduction.ratio(recordsPerBlock$count))
```

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

```
precision(recordsPerBlock$count, blockIDs)
```

```
## [1] NaN
```

```
recall(recordsPerBlock$count, blockIDs)
```

```
## [1] 0.002189145
```