Create your own recipes steps for omics data: the {scimo} package

useR! 2024
Antoine Bichat
July 9th
Salzburg

SERVIER
moved by you

- Intro
- Preprocessing
- Omics data
- Create your first step
- Dependencies without dependencies
- Outro

Presentation

Data scientist @ Servier

- Exploratory analysis
- Oncology, pediatric cancers, targeted therapies
- R, packages, shiny apps

SERVIER

Workshop at **Agrostat**, Bragança, Portugal, in September 24

Creation of an end-to-end machine learning pipeline with {tidymodels}

Notes

Open source package

`scimo` is a package developed on my own free time, and not for my employer.

Code chunk

There will be a lot of code in this presentation, but it is not necessary to look at it in detail on the first read-through.

The Palmer Archipelago penguins

```r
library(tidyverse)
library(palmerpenguins)

penguins
```

```
# A tibble: 344 × 8
   species island    bill_length_mm bill_depth_mm flipper_length_mm body_mass_g sex     year
   <fct>   <fct>              <dbl>         <dbl>             <int>       <int> <fct>  <int>
 1 Adelie  Torgersen           39.1          18.7               181        3750 male    2007
 2 Adelie  Torgersen           39.5          17.4               186        3800 female  2007
 3 Adelie  Torgersen           40.3          18                 195        3250 female  2007
 4 Adelie  Torgersen           NA            NA                  NA          NA <NA>    2007
 5 Adelie  Torgersen           36.7          19.3               193        3450 female  2007
 6 Adelie  Torgersen           39.3          20.6               190        3650 male    2007
 7 Adelie  Torgersen           38.9          17.8               181        3625 female  2007
 8 Adelie  Torgersen           39.2          19.6               195        4675 male    2007
 9 Adelie  Torgersen           34.1          18.1               193        3475 <NA>    2007
10 Adelie  Torgersen           42            20.2               190        4250 <NA>    2007
# ℹ 334 more rows
```

- Intro
- **Preprocessing**
- Omics data
- Create your first step
- Dependencies without dependencies
- Outro

tidymodels

> The tidymodels framework is a collection of packages for modeling and machine learning using tidyverse principles.

Your first recipe

> A recipe is an object that defines a series of roles and steps for data processing.

```
1 library(recipes)
2
3 penguins %>%
4   recipe(flipper_length_mm ~ .)
```

```
── Recipe ──────────────────────────────────────────────────────────────────────

── Inputs

Number of variables by role

outcome:   1
predictor: 7
```

Steps

```
penguins %>%
  recipe(flipper_length_mm  ~ .) %>%
  step_impute_mean(all_numeric_predictors(), -year) %>%
  step_normalize(all_numeric_predictors(), -year) %>%
  step_pca(all_numeric_predictors(), -year, num_comp = 2)
```

```
── Recipe ──────────────────────────────────────────────────────────────────────

── Inputs

Number of variables by role

outcome:   1
predictor: 7

── Operations

• Mean imputation for: all_numeric_predictors() and -year
• Centering and scaling for: all_numeric_predictors() and -year
• PCA extraction with: all_numeric_predictors() and -year
```

Estimation

```
penguins %>%
  recipe(flipper_length_mm  ~ .) %>%
  step_impute_mean(all_numeric_predictors(), -year) %>%
  step_normalize(all_numeric_predictors(), -year) %>%
  step_pca(all_numeric_predictors(), -year, num_comp = 2) %>%
  prep()
```

```
── Recipe ──────────────────────────────────────────────────────────────────────

── Inputs

Number of variables by role

outcome:   1
predictor: 7

── Training information

Training data contained 344 data points and 11 incomplete rows.

── Operations

• Mean imputation for: bill_length_mm, bill_depth_mm, body_mass_g | Trained
• Centering and scaling for: bill_length_mm, bill_depth_mm, body_mass_g | Trained
• PCA extraction with: bill_length_mm, bill_depth_mm, body_mass_g | Trained
```

Application

```
penguins %>%
  recipe(flipper_length_mm  ~ .) %>%
  step_impute_mean(all_numeric_predictors(), -year) %>%
  step_normalize(all_numeric_predictors(), -year) %>%
  step_pca(all_numeric_predictors(), -year, num_comp = 2) %>%
  prep() %>%
  bake(new_data = NULL)
```

```
# A tibble: 344 × 7
   species island    sex     year flipper_length_mm       PC1        PC2
   <fct>   <fct>     <fct>  <int>             <int>     <dbl>      <dbl>
 1 Adelie  Torgersen male    2007               181 -1.27     -0.0476
 2 Adelie  Torgersen female  2007               186 -0.854     0.429
 3 Adelie  Torgersen female  2007               195 -1.37      0.157
 4 Adelie  Torgersen <NA>    2007                NA  0.000199 -0.0000262
 5 Adelie  Torgersen female  2007               193 -1.92      0.00581
 6 Adelie  Torgersen male    2007               190 -1.81     -0.827
 7 Adelie  Torgersen female  2007               181 -1.16      0.352
 8 Adelie  Torgersen male    2007               195 -0.731    -0.522
 9 Adelie  Torgersen <NA>    2007               193 -1.86      0.774
10 Adelie  Torgersen <NA>    2007               190 -0.936    -1.03
# ℹ 334 more rows
```

Step info

```
penguins %>%
  recipe(flipper_length_mm  ~ .) %>%
  step_impute_mean(all_numeric_predictors(), -year) %>%
  step_normalize(all_numeric_predictors(), -year) %>%
  step_pca(all_numeric_predictors(), -year, num_comp = 2) %>%
  prep() %>%
  tidy(2)
```

```
# A tibble: 6 × 4
  terms          statistic   value id
  <chr>          <chr>       <dbl> <chr>
1 bill_length_mm mean         43.9  normalize_hFPQb
2 bill_depth_mm  mean         17.2  normalize_hFPQb
3 body_mass_g    mean       4202.   normalize_hFPQb
4 bill_length_mm sd            5.44 normalize_hFPQb
5 bill_depth_mm  sd            1.97 normalize_hFPQb
6 body_mass_g    sd          800.   normalize_hFPQb
```

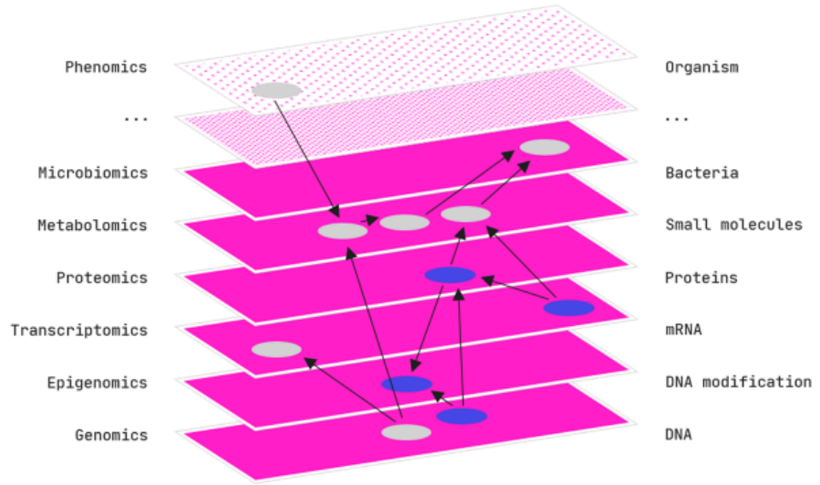
- Intro
- Preprocessing
- **Omics data**
- Create your first step
- Dependencies without dependencies
- Outro

Omics

> Omics data refers to **large** datasets generated through high-throughput sequencing, used to study and understand **complex biological systems**.

- **Genomics**: gene mutations, fusions...
- **Transcriptomics**: gene expression.
- **Proteomics**: protein abundance.
- **Metagenomics**: microorganism abundance.
- ...

Human multi-omics dataset



Need for specific preprocessing

- Feature selection steps:
  - keep genes with the most variability,
  - keep features significantly associated with the outcome.
- Feature aggregation steps:
  - compute pathway activity score,
  - sum all abundances belonging to the same clade.
- Feature normalization steps:
  - convert absolute counts to proportion.
- Feature generation steps:
  - reduce dimension for special data distribution,
  - extract clades from lineages.

{scimo}

```
1 library(scimo)
```

{scimo} provides extra recipes steps for dealing with omics data, while also being adaptable to other data types.


```
step_select_cv()
step_select_wilcoxon()
step_aggregate_list()
step_rownormalize_tss()
step_taxonomy()
```

...

Pediatric cancer dataset

Gene expression of 108 CCLE cell lines from 5 different pediatric cancers.

```
data("pedcan_expression")
pedcan_expression
```

```
# A tibble: 108 × 19,197
   cell_line sex    event      disease          A1BG   A1CF   A2M  A2ML1 A3GALT2 A4GALT  A4GNT  AAAS  AACS
   <chr>     <chr>  <chr>      <chr>           <dbl>  <dbl> <dbl>  <dbl>   <dbl>  <dbl>  <dbl> <dbl> <dbl>
 1 143B      Female Primary    Osteosarcoma     3.02 0.0566 2.78  0       0       2.13   0       5.55  2.90
 2 A-673     Female Primary    Ewing Sarcoma    4.87 0      2.00  3.19    0.0841  4.62   0.189   6.64  4.53
 3 BT-12     Female Primary    Embryonal Tumor  3.52 0.0286 0.111 0       0       2.32   0.0704  5.78  2.64
 4 BT-16     Male   Unknown    Embryonal Tumor  3.51 0      0.433 0.0144  0       1.54   0.0144  5.48  2.72
 5 C396      Male   Metastatic Osteosarcoma     4.59 0      0.956 0       0       5.10   0       4.81  2.70
 6 CADO-ES1  Female Metastatic Ewing Sarcoma    5.89 0      0.614 0.379   0.0704  6.60   0.151   6.08  3.92
 7 CAL-72    Male   Primary    Osteosarcoma     4.35 0.0426 0.333 0       0       0.614  0       4.53  3.42
 8 CBAGPN    Female Primary    Ewing Sarcoma    4.87 0.0976 1.33  0.111   0       0.722  0.0704  5.94  3.40
 9 CHLA-06   Female Unknown    Embryonal Tumor  5.05 0      0.124 0       0       0.848  0.138   4.11  2.22
10 CHLA-10   Female Unknown    Ewing Sarcoma    5.05 0.0144 0.949 1.73    0.0704  0.506  0.0704  5.86  4.65
# ℹ 98 more rows
# ℹ 19,184 more variables: AADAC <dbl>, AADACL2 <dbl>, AADACL3 <dbl>, AADACL4 <dbl>, AADAT <dbl>,
#   AAGAB <dbl>, AAK1 <dbl>, AAMDC <dbl>, AAMP <dbl>, AANAT <dbl>, AAR2 <dbl>, AARD <dbl>, AARS1 <dbl>,
#   AARS2 <dbl>, AARSD1 <dbl>, AASDH <dbl>, AASDHPPT <dbl>, AASS <dbl>, AATF <dbl>, AATK <dbl>, ABAT <dbl>, …
```

- Intro
- Preprocessing
- Omics data
- **Create your first step**
- Dependencies without dependencies
- Outro

Coefficient of variation

Omics data are usually **large data**. For `pedcan_expression`, $p \approx 20000 \gg n \approx 100$.

We want a step that will keep the top 10% of all variables that have the greater coefficient of variation.

$$CV = \frac{\sigma}{|\mu|}$$

```
cv <- function(x, na.rm = TRUE) {
  sd(x, na.rm = na.rm) / abs(mean(x, na.rm = na.rm))
}
```

`step_select_cv()` in action

```
rec_cv <-
  # recipe(disease ~ ., data = pedcan_expression) %>%  # too many variables for a formula
  recipe(pedcan_expression) %>%                        # <- workaround to avoid stack overflow error
  update_role(disease, new_role = "output") %>%        #    should be resolved in a future version of recipes
  update_role(-disease, new_role = "predictor") %>%    #
  step_select_cv(all_numeric_predictors(), prop_kept = 0.1) %>%
  prep()
```

```
bake(rec_cv, new_data = NULL)
```

```
# A tibble: 108 × 1,923
   cell_line sex    event      disease   A1CF  AADAC AADACL3
   <fct>     <fct>  <fct>      <fct>    <dbl>  <dbl>   <dbl>
 1 143B      Female Primary    Osteos… 0.0566 0.0704       0
 2 A-673     Female Primary    Ewing … 0      0.151        0
 3 BT-12     Female Primary    Embryo… 0.0286 3.49         0
 4 BT-16     Male   Unknown    Embryo… 0      0.0286       0
 5 C396      Male   Metastatic Osteos… 0      0            0
 6 CADO-ES1  Female Metastatic Ewing … 0      0            0
 7 CAL-72    Male   Primary    Osteos… 0.0426 0            0
 8 CBAGPN    Female Primary    Ewing … 0.0976 0            0
 9 CHLA-06   Female Unknown    Embryo… 0      0            0
10 CHLA-10   Female Unknown    Ewing … 0.0144 0            0
# ℹ 98 more rows
# ℹ 1,916 more variables: ABCB11 <dbl>, ABCC12 <dbl>,
#   ABRA <dbl>, AC002456.2 <dbl>, AC008397.1 <dbl>,
#   ACOD1 <dbl>, ACSM1 <dbl>, ACSM2B <dbl>, ACSM5 <dbl>, …
```

```
tidy(rec_cv, 1)
```

```
# A tibble: 19,193 × 4
   terms       cv kept  id
   <chr>    <dbl> <lgl> <chr>
 1 A1BG     0.371 FALSE select_cv_3YfWC
 2 A1CF     4.60  TRUE  select_cv_3YfWC
 3 A2M      1.69  FALSE select_cv_3YfWC
 4 A2ML1    2.45  FALSE select_cv_3YfWC
 5 A3GALT2  2.37  FALSE select_cv_3YfWC
 6 A4GALT   0.979 FALSE select_cv_3YfWC
 7 A4GNT    1.53  FALSE select_cv_3YfWC
 8 AAAS     0.0934 FALSE select_cv_3YfWC
 9 AACS     0.194 FALSE select_cv_3YfWC
10 AADAC    3.40  TRUE  select_cv_3YfWC
# ℹ 19,183 more rows
```

User interface: `step_select_cv()`

```
step_select_cv <- function(recipe, ..., role = NA, trained = FALSE,
                           n_kept = NULL, prop_kept = NULL,
                           cutoff = NULL, res = NULL,
                           skip = FALSE,
                           id = rand_id("select_cv")) {

  add_step(                          # Add a new step to the existing recipe
    recipe,
    step_select_cv_new(              # Arguments are passed as is
      terms = enquos(...),
      role = role,
      trained = trained,             # trained = FALSE
      n_kept = n_kept,
      prop_kept = prop_kept,
      cutoff = cutoff,
      res = res,                     # res = NULL to update later
      skip = skip,
      id = id                        # Random id
    )
  )
}
```

Step creation: `step_select_cv_new()`

```r
step_select_cv_new <- function(terms, role, trained,
                               n_kept, prop_kept, cutoff,
                               res, skip, id) {

  step(
    subclass = "select_cv",  # Specify the class to dispatch future methods
    terms = terms,           # Arguments are passed as is
    role = role,
    trained = trained,
    n_kept = n_kept,
    prop_kept = prop_kept,
    cutoff = cutoff,
    res = res,
    skip = skip,
    id = id
  )
}
```

Computing: `prep.step_select_cv()`

```
prep.step_select_cv <- function(x, training, info = NULL, ...) {

  col_names <- recipes_eval_select(x$terms, training, info)    # x is a list containing step info
  check_type(training[, col_names], quant = TRUE)              # Check variables

  #####

  res_cv <-                                  # Compute quantities
    training[, col_names] %>%                #  A tibble with 3 columns:
    apply(2, cv) %>%                         #  features, CVs and whether to keep them
    enframe(name = "terms", value = "cv") %>%
    mutate(kept = var_to_keep(.data$cv, x$n_kept, x$prop_kept, x$cutoff, maximize = TRUE))

  #####

  step_select_cv_new(    # Update step in recipe
    terms = x$terms,     # Most arguments are passed as is
    role = x$role,
    trained = TRUE,      # The step is now trained
    n_kept = x$n_kept,
    prop_kept = x$prop_kept,
    cutoff = x$cutoff,
    res = res_cv,        # Result to store to use later
    skip = x$skip,
    id = x$id
  )
}
```

Applying: `bake.step_select_cv()`

```
bake.step_select_cv <- function(object, new_data, ...) {

  col_names <- object$res$terms                # object is a list containing step info
  check_new_data(col_names, object, new_data)  # Check variables

  #####

  col_to_remove <-             # Do things
    object$res %>%             #  remove unwanted columns
    filter(!.data$kept) %>%
    pull(.data$terms)

  new_data[col_to_remove] <- NULL

  #####

  new_data # Return updated dataset
}
```

Getting informations: `tidy.step_select_cv()`

```
tidy.step_select_cv <- function(x, ...) {

  if (is_trained(x)) {
    res <- x$res                        # res contains all necessary information
  } else {
    term_names <- sel2char(x$terms)
    res <-
      tibble(
        terms = term_names,             # Returns NA when not trained
        cv = rlang::na_dbl,
        rank = rlang::na_dbl,
        kept = rlang::na_lgl
      )
  }

  res$id <- x$id                       # Add the unique random id
  res
}
```

Printing: `print.step_select_cv()`

```
print.step_select_cv <- function(x, width = max(20, options()$width - 35), ...) {

  title <- "Top CV filtering on "

  print_step(
    tr_obj = x$res$terms,
    untr_obj = x$terms,
    trained = x$trained,
    title = title,
    width = width
  )

  invisible(x)
}
```

Methods to import

To correctly manage your NAMESPACE

```
#' @keywords internal
"_PACKAGE"

## usethis namespace: start
#' @importFrom generics required_pkgs tidy
#' @importFrom recipes prep bake
#' @importFrom tibble tibble
## usethis namespace: end
NULL
```

- Intro
- Preprocessing
- Omics data
- Create your first step
- **Dependencies without dependencies**
- Outro

Deal with taxonomic lineages

```
data("cheese_taxonomy")

cheese_taxonomy %>%
  select(asv, lineage)
```

```
# A tibble: 74 × 2
   asv    lineage
   <chr>  <chr>
 1 asv_01 k__Fungi|p__Ascomycota|c__Dothideomycetes|o__Dothideales|f__Dothioraceae|g__Aureobasidium|s__Aureob…
 2 asv_02 k__Fungi|p__Ascomycota|c__Eurotiomycetes|o__Eurotiales|f__Aspergillaceae|g__Aspergillus|s__Aspergil…
 3 asv_03 k__Fungi|p__Ascomycota|c__Eurotiomycetes|o__Eurotiales|f__Aspergillaceae|g__Penicillium|s__Penicill…
 4 asv_04 k__Fungi|p__Ascomycota|c__Eurotiomycetes|o__Eurotiales|f__Aspergillaceae|g__Penicillium|s__Penicill…
 5 asv_05 k__Fungi|p__Ascomycota|c__Eurotiomycetes|o__Eurotiales|f__Aspergillaceae|g__Penicillium|s__Penicill…
 6 asv_06 k__Fungi|p__Ascomycota|c__Eurotiomycetes|o__Eurotiales|f__Aspergillaceae|g__Penicillium|s__Penicill…
 7 asv_07 k__Fungi|p__Ascomycota|c__Eurotiomycetes|o__Eurotiales|f__Aspergillaceae|g__Penicillium|s__Penicill…
 8 asv_08 k__Fungi|p__Ascomycota|c__Eurotiomycetes|o__Eurotiales|f__Aspergillaceae|g__Penicillium|s__Penicill…
 9 asv_09 k__Fungi|p__Ascomycota|c__Eurotiomycetes|o__Eurotiales|f__Aspergillaceae|g__Penicillium|s__Penicill…
10 asv_10 k__Fungi|p__Ascomycota|c__Saccharomycetes|o__Saccharomycetales|f__Debaryomycetaceae|g__Debaryomyces…
# ℹ 64 more rows
```

Deal with taxonomic lineages

```
data("cheese_taxonomy")

cheese_taxonomy %>%
  select(asv, lineage) %>%
  mutate(order = yatah::get_clade(lineage, "order"),
         genus = yatah::get_clade(lineage, "genus"))
```

```
# A tibble: 74 × 4
   asv    lineage                                                          order           genus
   <chr>  <chr>                                                            <chr>           <chr>
 1 asv_01 k__Fungi|p__Ascomycota|c__Dothideomycetes|o__Dothideales|f__Dothiorac… Dothideales     Aureobasidium
 2 asv_02 k__Fungi|p__Ascomycota|c__Eurotiomycetes|o__Eurotiales|f__Aspergillac… Eurotiales      Aspergillus
 3 asv_03 k__Fungi|p__Ascomycota|c__Eurotiomycetes|o__Eurotiales|f__Aspergillac… Eurotiales      Penicillium
 4 asv_04 k__Fungi|p__Ascomycota|c__Eurotiomycetes|o__Eurotiales|f__Aspergillac… Eurotiales      Penicillium
 5 asv_05 k__Fungi|p__Ascomycota|c__Eurotiomycetes|o__Eurotiales|f__Aspergillac… Eurotiales      Penicillium
 6 asv_06 k__Fungi|p__Ascomycota|c__Eurotiomycetes|o__Eurotiales|f__Aspergillac… Eurotiales      Penicillium
 7 asv_07 k__Fungi|p__Ascomycota|c__Eurotiomycetes|o__Eurotiales|f__Aspergillac… Eurotiales      Penicillium
 8 asv_08 k__Fungi|p__Ascomycota|c__Eurotiomycetes|o__Eurotiales|f__Aspergillac… Eurotiales      Penicillium
 9 asv_09 k__Fungi|p__Ascomycota|c__Eurotiomycetes|o__Eurotiales|f__Aspergillac… Eurotiales      Penicillium
10 asv_10 k__Fungi|p__Ascomycota|c__Saccharomycetes|o__Saccharomycetales|f__Deb… Saccharomyceta… Debaryomyces
# ℹ 64 more rows
```

step_taxonomy()

```
cheese_taxonomy %>%
  recipe(~ asv + lineage) %>%
  step_taxonomy(lineage, rank = c("order", "genus")) %>%
  prep() %>%
  bake(new_data = NULL)
```

```
# A tibble: 74 × 3
   asv    lineage_order     lineage_genus
   <fct>  <chr>             <chr>
 1 asv_01 Dothideales       Aureobasidium
 2 asv_02 Eurotiales        Aspergillus
 3 asv_03 Eurotiales        Penicillium
 4 asv_04 Eurotiales        Penicillium
 5 asv_05 Eurotiales        Penicillium
 6 asv_06 Eurotiales        Penicillium
 7 asv_07 Eurotiales        Penicillium
 8 asv_08 Eurotiales        Penicillium
 9 asv_09 Eurotiales        Penicillium
10 asv_10 Saccharomycetales Debaryomyces
# ℹ 64 more rows
```

The classical way

```
#' @importFrom yatah get_clade
bake.step_taxonomy <- function(object, new_data, ...) {
  ...
  new_col <- paste0(term, "_", rank)

  new_data[[new_col]] <- get_clade(new_data[[term]], rank = rank, same = FALSE)
  ...
  return(new_data)
}
```

- Need to add `{yatah}` as a dependency.

A word about `call2()` and `eval_tidy()`

```
head(fruit)
```

```
[1] "apple"       "apricot"     "avocado"     "banana"      "bell pepper" "bilberry"
```

```
knitr::combine_words(head(fruit), and = " or ")
```

```
apple, apricot, avocado, banana, bell pepper, or bilberry
```

```
library(rlang)
cl <- call2("combine_words", .ns = "knitr",
            words = head(fruit), and = " or ")
cl
```

```
knitr::combine_words(words = c("apple", "apricot", "avocado",
"banana", "bell pepper", "bilberry"), and = " or ")
```

```
eval_tidy(cl)
```

```
apple, apricot, avocado, banana, bell pepper, or bilberry
```

The dependency-free way

```
#' @importFrom rlang eval_tidy call2
bake.step_taxonomy <- function(object, new_data, ...) {
  ...
  new_col <- paste0(term, "_", rank)

  yatah_call <- call2("get_clade", .ns = "yatah",
                      lineage = new_data[[term]], rank = rank, same = TRUE)
  new_data[[new_col]] <- eval_tidy(yatah_call)
  ...
  return(new_data)
}
```

- `{yatah}` is no longer needed.
- `{rlang}` is already a `{recipes}` dependency.

`required_pkgs()`

```
required_pkgs.step_taxonomy <- function(x, ...) {
  c("yatah", "scimo")
}
```

- Check if the used package is installed.
- Correctly load the package for parallel processing.
- Used also for other steps in `{scimo}`, and returns only `"scimo"`.

- Intro
- Preprocessing
- Omics data
- Create your first step
- Dependencies without dependencies
- **Outro**

Next steps

- Tunable arguments.
- New steps
  - Other tests (`limma`, `DESeq2`...) for feature selection,
  - `PLNmodels` dimension reduction,
  - Batch effect removal,
  - Multi-omics steps (how to define groups with tidy selection?).

To go further

- Packages
  - tidymodels
  - recipes
  - scimo
- Books
  - Tidy Modeling with R
- Vignette
  - Create your own recipe step function

Big thanks

Julie Aubert

for ideas, discussions and contributions

Emil Hvitfeldt

for review and issues

Sylvain Jonchery

for the logo