Structural and canopy variables

Luciano L.M. De Benedictis

2026-01-07

```
# setup ------------------------------------------------------------------

library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.6
## v forcats   1.0.1     v stringr   1.6.0
## v ggplot2   4.0.1     v tibble    3.3.0
## v lubridate 1.9.4     v tidyr     1.3.2
## v purrr     1.2.0
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(GGally)
library(klaR)
```

```
## Loading required package: MASS
##
## Attaching package: 'MASS'
##
## The following object is masked from 'package:dplyr':
##
##     select
```

```
library(fclust)
library(party)
```

```
## Loading required package: grid
## Loading required package: mvtnorm
## Loading required package: modeltools
## Loading required package: stats4
## Loading required package: strucchange
## Loading required package: zoo
##
## Attaching package: 'zoo'
##
## The following objects are masked from 'package:base':
##
```

```
##     as.Date, as.Date.numeric
##
## Loading required package: sandwich
##
## Attaching package: 'strucchange'
##
## The following object is masked from 'package:stringr':
##
##     boundary
##
##
## Attaching package: 'party'
##
## The following object is masked from 'package:dplyr':
##
##     where
```

```
set.seed(92538)

# function for the upper triangle of pairs plot. Adapted from Solomon Kurz.

my_upper <- function(data, mapping, ...) {

  # get the x and y data to use the other code
  x <- eval_data_col(data, mapping$x)
  y <- eval_data_col(data, mapping$y)

  r  <- unname(cor.test(x, y)$estimate)
  rt <- format(r, digits = 2)[1]
  tt <- as.character(rt)

  # plot the cor value
  ggally_text(
    label = tt,
    mapping = aes(),
    size = 4)+
    theme_void()
}

# import data ------------------------------------------------------------

# DCP canopy indices
canopy <- read_csv("data/DCP.csv") |>
  group_by(site) |>
  summarise(LAImean = mean(LAI), LAIsd = sd(LAI)) |>
  rename(LAI = LAImean)
```

```
## Rows: 384 Columns: 10
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (2): site, Date
## dbl (8): Gap Fraction, Foliage Cover, Crown Cover, Crown Porosity, LAI, effe...
##
## i Use `spec()` to retrieve the full column specification for this data.
```

```
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
# TLS structural indices
structure <- read_csv("data/TLS.csv") |>
  mutate(links_norm = n_links_4_6/TopH,
         min_pc_norm = pc_min_sect/TopH,
         max_pc_norm = pc_max_sect/TopH,
         sd_perc_norm = sd_perc_clust/TopH) |>
  dplyr::select(!c("Sampling_date", "n_links_4_6", "pc_min_sect", "pc_max_sect", "sd_perc_clust"))
```

```
## Rows: 28 Columns: 11
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (2): site, Sampling_date
## dbl (9): pc_min_sect, pc_max_sect, sd_perc_clust, n_links_4_6, TopH, VCI, RH...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
# join together
variables <- structure |>
  inner_join(canopy)
```

```
## Joining with `by = join_by(site)`
```

```
rm(structure, canopy)

# join categories
cat <- read_csv("data/classification.csv")
```

```
## Rows: 32 Columns: 2
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (2): site, category
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

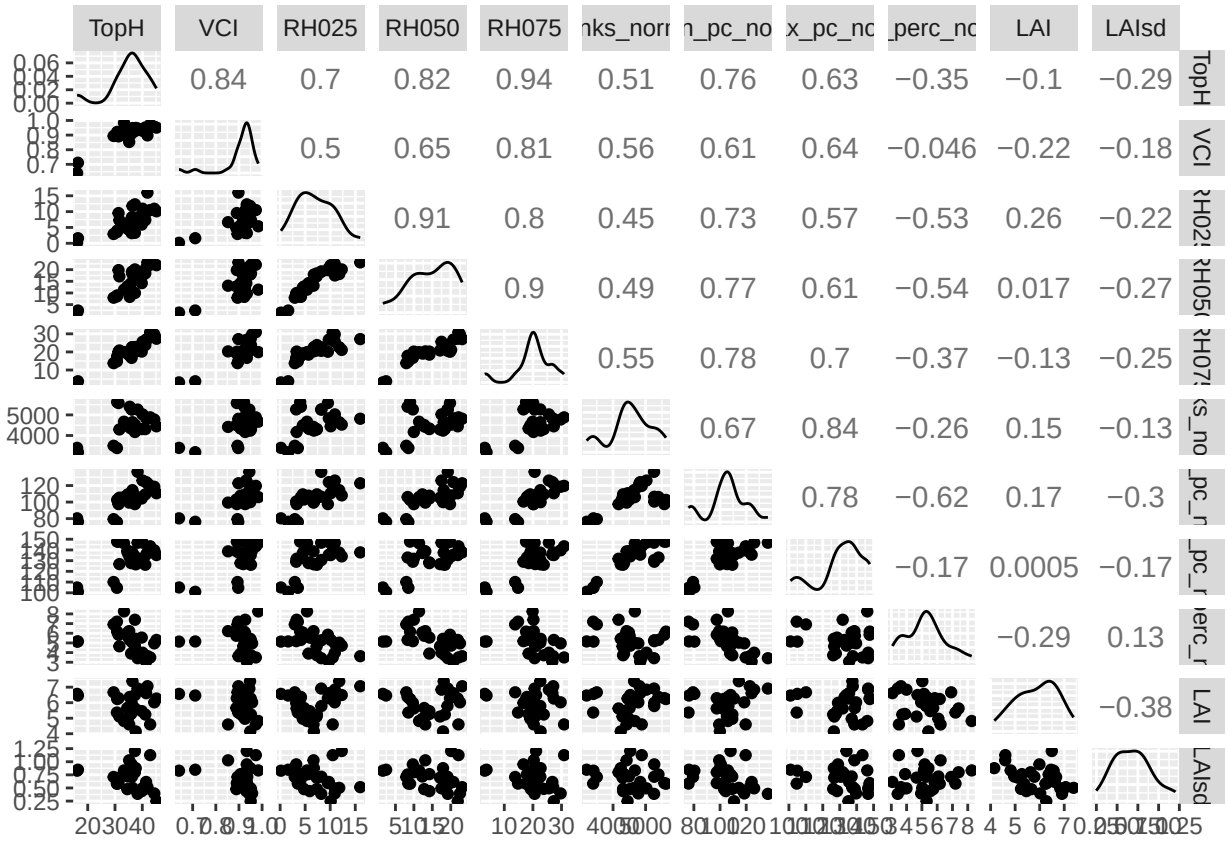
```
variables <- cat |>
  inner_join(variables) |>
  mutate(category = as.factor(category))
```

```
## Joining with `by = join_by(site)`
```

```
rm(cat)

variables |>
  dplyr::select(2:7, 12, 13) |>
  print(n = 30)
```

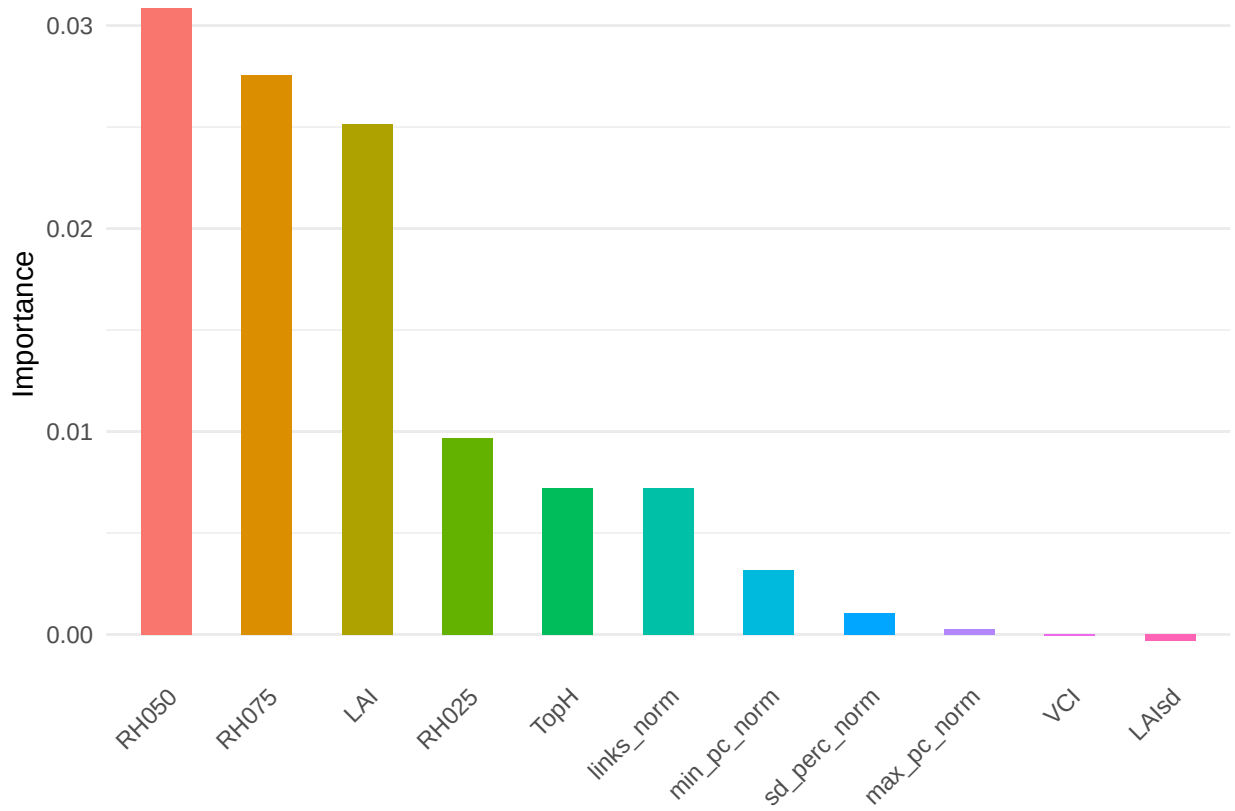
```
## # A tibble: 28 x 8
##    category     TopH   VCI  RH025 RH050 RH075   LAI LAIsd
##    <fct>       <dbl> <dbl>  <dbl> <dbl> <dbl> <dbl> <dbl>
##  1 Coppice      16.1 0.641  0.159  1.57  3.16  6.57 0.829
##  2 Coppice      16.4 0.711  1.61   2.46  3.79  6.46 0.848
##  3 High Forest  37.6 0.949  6.74  17.9  22.7   4.15 0.877
##  4 High Forest  31.6 0.919  7.29  17.1  20.8   5.11 0.696
##  5 Old growth   39.8 0.932  5.91  14.2  20.2   5.46 0.440
##  6 Old growth   41.1 0.952  7.34  18.4  23.8   5.24 0.483
##  7 Old growth   35.9 0.907  5.67  14.2  18.5   5.27 0.764
##  8 Old growth   34.7 0.926  4.53  12.3  19.1   4.96 0.719
##  9 Old growth   37.3 0.927  3.3    9.98 20.2   5.63 0.800
## 10 Coppice      29.5 0.894  2.93   7.95 13.8   6.67 0.710
## 11 Coppice      30.7 0.897  3.54   8.51 14.8   5.37 0.583
## 12 Old growth   33.3 0.984  5.37  11.4  19.9   4.82 0.834
## 13 Old growth   35.2 0.853  6.67  13.1  20.3   4.60 1.03
## 14 Old growth   34.5 0.889  4.78  12.8  19.3   5.66 0.478
## 15 High Forest  40.1 0.931 10.9   17.5  25.6   6.84 0.493
## 16 High Forest  38.3 0.945  8.21  19.2  20.6   6.91 0.388
## 17 High Forest  41.9 0.898 16     23.1  27.1   7.06 0.516
## 18 High Forest  36.2 0.936  8.24  19.1  22.6   5.78 0.714
## 19 Coppice      36.1 0.934  3.21   9.72 16.7   5.99 0.950
## 20 Coppice      35.4 0.923  4.15   8.31 17.9   6.37 0.758
## 21 Coppice      30.8 0.922  4.01   8.96 17.6   6.24 0.581
## 22 High Forest  31.3 0.891  9.54  19.8  20.1   6.29 0.593
## 23 Coppice      36.4 0.948 11.8   18.1  22.6   7.38 0.503
## 24 Coppice      37.3 0.934 12.3   20.1  21.1   6.49 1.20
## 25 High Forest  41.4 0.934  9.73  20.7  26.8   6.61 0.615
## 26 Old growth   45.2 0.951 10.1   21.9  27.3   6.27 0.248
## 27 Old growth   44.4 0.957 10.9   22.5  29.7   6.02 0.396
## 28 Old growth   42.9 0.973 10.5   22.1  30.9   4.60 1.13
```

```
#pairs plot
variables[3:13] |>
  ggpairs(upper = list(continuous =  my_upper))
```




```
# standardization
# FLM test is non-parametric, same results either way
# but standardization gives more meaningful coefficients and is required for other steps here

variables <- variables |>
  mutate(across(where(is.numeric), scale))|>
  mutate(across(where(is.numeric), as.vector))

# random forest ------------------------------------------------------------------

forest <- cforest(category ~ ., data = variables[, -1],
                  control = cforest_control(mtry = 3, ntree = 5000, mincriterion = 0)) #mtry default fr

#variable importance from RF model
importance <- varimp(forest, conditional=T)

importance <- tibble(variable = names(importance), importance = importance) |>
  mutate(variable = fct_reorder(variable, importance, .desc = T)) |>
  arrange(variable)

#plot IV
importance |>
  ggplot(aes(x = variable, y = importance, fill = variable))+
  geom_col(width = 0.5)+
  theme_minimal()+
  theme(legend.position = "none",
```

```
        axis.text.x = element_text(angle = 45, hjust = 1),
        panel.grid.major.x = element_blank())+
  labs(x = NULL, y = "Importance")
```




```
ggsave("plots/importance.png", width = 190, height = 100, units = "mm", bg = 'white',
       scale = 0.8, dpi = 1000)
ggsave('plots/FIG1.eps', width = 190, height = 100, units = "mm", bg = 'white',
       scale = 0.8, dpi = 1000)

sel_var <- variables |>
  dplyr::select(site, category, RH050, LAI)

saveRDS(sel_var, "selected_variables.rds")

# clustering ------------------------------------------------------------------

# fuzzy k-means
fuzzy <- FKM(sel_var[-c(1, 2)], k = 3)

#add to df
sel_var <- sel_var |>
  mutate(cluster_fuzzy = fuzzy$clus[,1],
         membership = fuzzy$clus[,2])

#assign the most plausible labels
```

```r
contingency_fuzzy <- as.matrix(table(sel_var$category, sel_var$cluster_fuzzy))
assignment_fuzzy <- clue::solve_LSAP(t(contingency_fuzzy), maximum = T)

sel_var <- sel_var |>
  mutate(cluster_fuzzy = assignment_fuzzy[cluster_fuzzy]) |>
  mutate(cluster_fuzzy = factor(cluster_fuzzy, labels = levels(category)))

#check agreement

table(sel_var$category, sel_var$cluster_fuzzy)
```

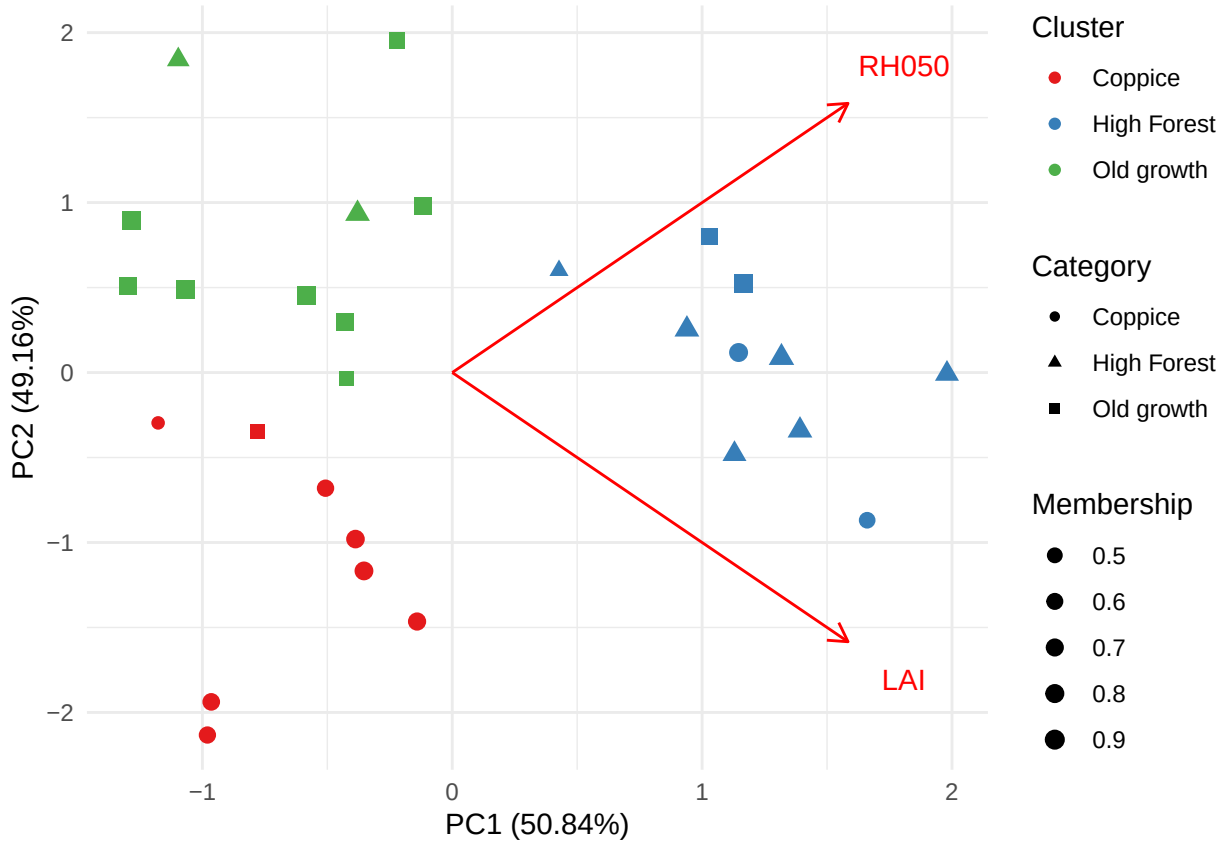
```
##
##               Coppice High Forest Old growth
##   Coppice           7           2          0
##   High Forest       0           6          2
##   Old growth        1           2          8
```

```r
#fuzzy Rand Index
RI.F(VC = sel_var$category, U = fuzzy$U)
```

```
## [1] 0.6613095
```

```r
sink("other results/external_validation.txt")
"Fuzzy Rand Index"
RI.F(VC = sel_var$category, U = fuzzy$U)
sink()

#perform PCA after clustering, only for 2D plotting

pca_results <- sel_var |>
  dplyr::select(RH050, LAI) |>
  prcomp(scale. = T, rank. = 2)

sel_var <- cbind(sel_var, pca_results$x)

#plot PCA

#adapted from ggbiplot source code

#variance explained
expl_var <- round(pca_results$sdev^2/sum(pca_results$sdev^2)*100, 2)

labels <- str_c(c("PC1 ", "PC2 "),
                str_c("(", expl_var, "%)"))

#how much to shift variable labels
lbl_shift <- 0.1

#loadings and labels
loadings_tbl <- as_tibble(pca_results$rotation, rownames = "var") |>
  mutate(xlab = PC1 + lbl_shift,
         ylab = PC2 + c(lbl_shift, -lbl_shift))
```

```r
#default scaler from ggbiplot
scaler <- min(max(abs(sel_var$PC1))/max(abs(loadings_tbl$PC1)),
              max(abs(sel_var$PC2))/max(abs(loadings_tbl$PC2)))*0.8

loadings_tbl <- loadings_tbl |>
  mutate(across(PC1:ylab, ~ .x * scaler))

#clustering + PCA plot
sel_var |>
  ggplot()+
  geom_point(aes(x = PC1, y = PC2, color = cluster_fuzzy,
                                   shape = category, size = membership)) +
  geom_segment(data = loadings_tbl, mapping = aes(x = 0, y = 0, xend = PC1,
                                                  yend = PC2),
               arrow = grid::arrow(length = grid::unit(8, "points")), colour = "red",
               linewidth = 0.5)+
  geom_text(data = loadings_tbl, aes(x = xlab, y = ylab, label = var), colour = "red")+
  scale_size_area(max_size = 3) +
  scale_color_brewer(palette = "Set1")+
  theme_minimal() +
  labs(color = "Cluster", shape = "Category", size = "Membership",
       x = labels[1], y = labels[2])
```



```r
ggsave("plots/cluster_fuzzy_select.png", width = 190, height = 117,
       units = "mm", bg = 'white', scale = 1, dpi = 1000)
ggsave('plots/FIG2.eps', width = 190, height = 117,
       units = "mm", bg = 'white', scale = 1)

#summary statistics by cluster
sel_var |>
  group_by(cluster_fuzzy) |>
  summarise(across(3:5, mean))
```

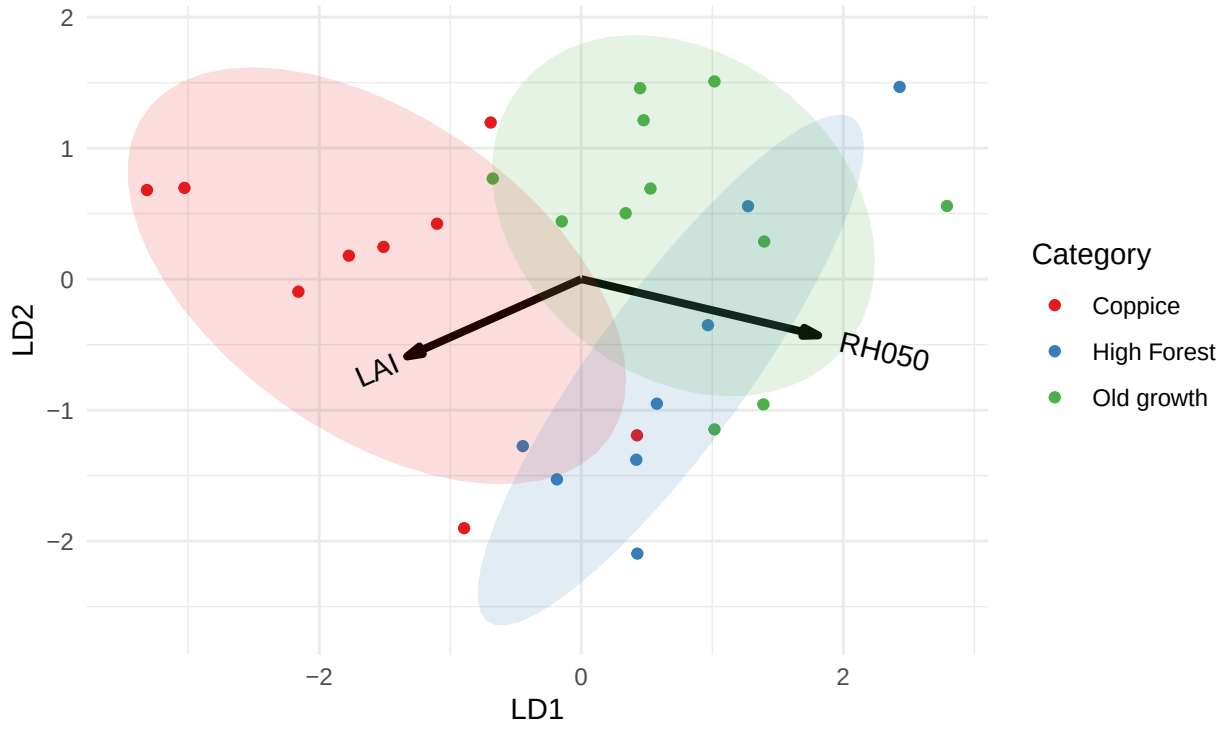
```
## # A tibble: 3 x 4
##   cluster_fuzzy  RH050    LAI membership
##   <fct>          <dbl>  <dbl>      <dbl>
## 1 Coppice       -1.26   0.328      0.791
## 2 High Forest    0.910  0.813      0.855
## 3 Old growth     0.100 -1.08       0.817
```

```r
# LDA ----------------------------------------------------------------------

# vector of classes
class <- variables[[2]]

# standardize variables
vars <- variables[3:13]

# variable selection, criterion "ability to separate"
step <- stepclass(x = vars, grouping = class, method = "lda", direction = "both",
                  criterion = "AS", improvement = 0.1)
```

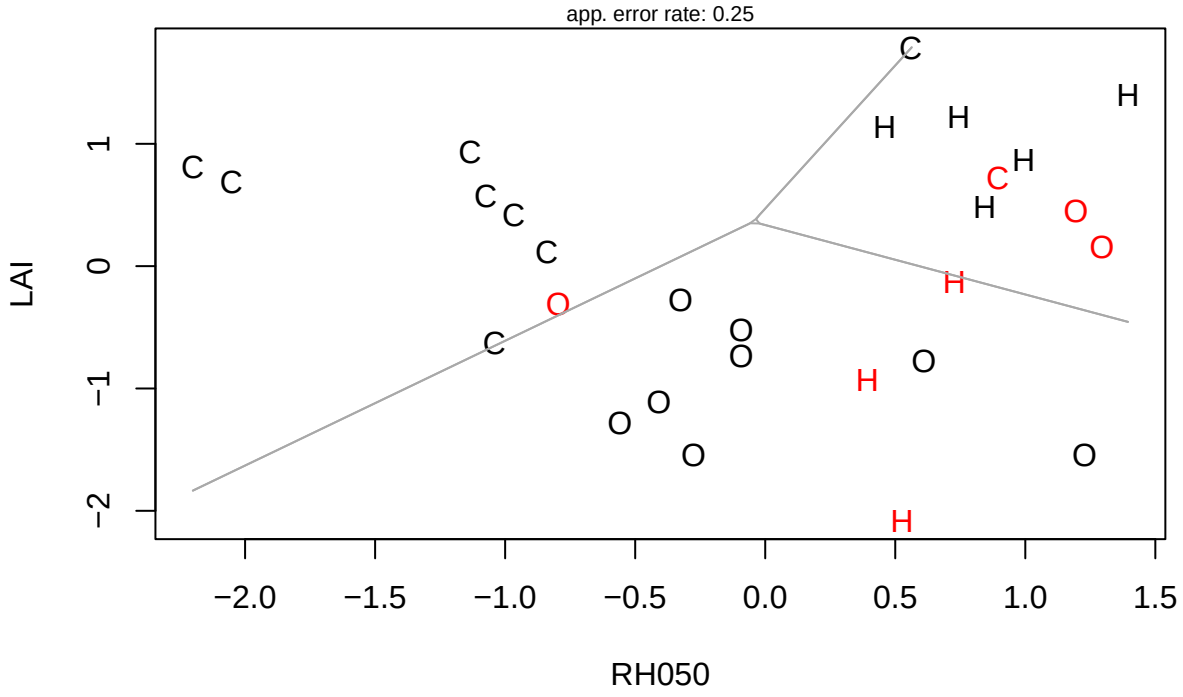
```
##  `stepwise classification', using 10-fold cross-validated abiltity to seperate of method lda'.
## 28 observations of 11 variables in 3 classes; direction: both
## stop criterion: improvement less than 10%.

## abiltity to seperate: 0.3394;  in: "LAI";  variables (1): LAI
## abiltity to seperate: 0.58293;  in: "RH050";  variables (2): LAI, RH050
##
##  hr.elapsed min.elapsed sec.elapsed
##       0.000       0.000       0.161
```

```r
wilks <- greedy.wilks(X = vars, grouping = class, niveau = 1) # keep going until all variables selected

write_csv(format(wilks$results, digits = 2), file ="other results/wilks_selection.csv")

# LDA with selected variables

selected <- vars[step[["model"]][["nr"]]]

lda <- lda(x = selected, grouping = class)

ggbiplot::ggbiplot(lda, groups = class, ellipse = T, varname.size = 4,
                   varname.adjust = 1.5,
                   ellipse.linewidth = NA, ellipse.alpha = 0.15, scale = 1)+
```

```
  labs(x = "LD1", y = "LD2", color = "Category", fill = "Category")+
  theme_minimal()+
  scale_color_brewer(palette = "Set1")+
  scale_fill_brewer(palette = "Set1")
```




```
ggsave("plots/LDA.png", width = 190, height = 117,
       units = "mm", bg = 'white', scale = 1.1, dpi = 1000)

drawparti(x = selected[[1]], y = selected[[2]], grouping = class, method = "lda",
          xlab = names(selected)[1], ylab = names(selected)[2], imageplot = F, col.mean = NULL)
```




```
png("plots/LDA_parti.png", bg = "white", width = 190, height = 117, units = "mm",
    pointsize = 12, res = 1000)
drawparti(x = selected[[1]], y = selected[[2]], grouping = class, method = "lda",
          xlab = names(selected)[1], ylab = names(selected)[2], imageplot = F, col.mean = NULL)
title("Partition plot")
dev.off()
```

```
## pdf
##   2
```

```
# session info ------------------------------------------------------------

sessionInfo()
```

```
## R version 4.5.2 (2025-10-31)
## Platform: x86_64-redhat-linux-gnu
## Running under: Nobara Linux 43 (KDE Plasma Desktop Edition)
##
## Matrix products: default
## BLAS/LAPACK: FlexiBLAS OPENBLAS-OPENMP;  LAPACK version 3.12.1
##
## locale:
##  [1] LC_CTYPE=it_IT.UTF-8      LC_NUMERIC=C
##  [3] LC_TIME=it_IT.utf8        LC_COLLATE=it_IT.UTF-8
##  [5] LC_MONETARY=it_IT.utf8    LC_MESSAGES=it_IT.UTF-8
```

```
##  [7] LC_PAPER=it_IT.utf8       LC_NAME=C
##  [9] LC_ADDRESS=C              LC_TELEPHONE=C
## [11] LC_MEASUREMENT=it_IT.utf8 LC_IDENTIFICATION=C
##
## time zone: Europe/Rome
## tzcode source: system (glibc)
##
## attached base packages:
## [1] stats4    grid      stats     graphics  grDevices datasets  utils
## [8] methods   base
##
## other attached packages:
##  [1] party_1.3-18      strucchange_1.5-4 sandwich_3.1-1    zoo_1.8-15
##  [5] modeltools_0.2-24 mvtnorm_1.3-3     fclust_2.1.3      klaR_1.7-3
##  [9] MASS_7.3-65       GGally_2.4.0      lubridate_1.9.4   forcats_1.0.1
## [13] stringr_1.6.0     dplyr_1.1.4       purrr_1.2.0       readr_2.1.6
## [17] tidyr_1.3.2       tibble_3.3.0      ggplot2_4.0.1     tidyverse_2.0.0
##
## loaded via a namespace (and not attached):
##  [1] tidyselect_1.2.1   libcoin_1.0-10     farver_2.1.2       S7_0.2.1
##  [5] fastmap_1.2.0      combinat_0.0-8     TH.data_1.1-5      promises_1.5.0
##  [9] labelled_2.16.0    digest_0.6.39      timechange_0.3.0   mime_0.13
## [13] lifecycle_1.0.4    cluster_2.1.8.1    survival_3.8-3     magrittr_2.0.4
## [17] compiler_4.5.2     rlang_1.1.6        tools_4.5.2        ggbiplot_0.6.2
## [21] utf8_1.2.6         yaml_2.3.12        knitr_1.50         labeling_0.4.3
## [25] bit_4.6.0          RColorBrewer_1.1-3 multcomp_1.4-29    miniUI_0.1.2
## [29] withr_3.0.2        xtable_1.8-4       scales_1.4.0       cli_3.6.5
## [33] rmarkdown_2.30     crayon_1.5.3       ragg_1.5.0         generics_0.1.4
## [37] otel_0.2.0         rstudioapi_0.17.1  tzdb_0.5.0         splines_4.5.2
## [41] parallel_4.5.2     CoprManager_0.5.7  matrixStats_1.5.0  vctrs_0.6.5
## [45] Matrix_1.7-4       hms_1.1.4          bit64_4.6.0-1      clue_0.3-66
## [49] systemfonts_1.3.1  glue_1.8.0         ggstats_0.12.0     codetools_0.2-20
## [53] stringi_1.8.7      gtable_0.3.6       questionr_0.8.1    later_1.4.4
## [57] pillar_1.11.1      htmltools_0.5.9    R6_2.6.1           textshaping_1.0.4
## [61] vroom_1.6.7        evaluate_1.0.5     shiny_1.12.1       lattice_0.22-7
## [65] haven_2.5.5        highr_0.11         httpuv_1.6.16      Rcpp_1.1.0
## [69] xfun_0.55          coin_1.4-3         pkgconfig_2.0.3
```