project

2025-11-25

The dataset used in this project corresponds to the supplementary file **MOESM2 (ESM)** from the publication by Fan et al. (2018). It contains the raw GC-TOF-MS metabolomics data for 120 urine samples from healthy adults (60 males and 60 females), enabling independent preprocessing, exploratory analysis, feature selection, and classification.

Import Library

```
# ------------------------------
# Libraries
# ------------------------------

# Data import and manipulation
library(readxl)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ─
─
## ✔ forcats   1.0.1     ✔ readr     2.1.6
## ✔ ggplot2   4.0.1     ✔ stringr   1.6.0
## ✔ lubridate 1.9.4     ✔ tibble    3.3.0
## ✔ purrr     1.2.0     ✔ tidyr     1.3.1
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ─
─
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conf
licts to become errors
```

```
# Visualization
library(ggplot2)
library(ellipse)
```

```
##
## Attaching package: 'ellipse'
##
## The following object is masked from 'package:graphics':
##
##     pairs
```

```
library(knitr)

# Multivariate analysis
#library(pls)
library(mixOmics)
```

```
## Loading required package: MASS
##
## Attaching package: 'MASS'
##
## The following object is masked from 'package:dplyr':
##
##     select
##
## Loading required package: lattice
##
## Loaded mixOmics 6.30.0
## Thank you for using mixOmics!
## Tutorials: http://mixomics.org
## Bookdown vignette: https://mixomicsteam.github.io/Bookdown
## Questions, issues: Follow the prompts at http://mixomics.org/contact-us
## Cite us:  citation('mixOmics')
##
##
## Attaching package: 'mixOmics'
##
## The following object is masked from 'package:purrr':
##
##     map
```

```
# Machine learning and validation
library(caret)
```

```
##
## Attaching package: 'caret'
##
## The following objects are masked from 'package:mixOmics':
##
##     nearZeroVar, plsda, splsda
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.
##
## Attaching package: 'pROC'
##
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
# Statistics and utilities
library(DescTools)
```

```
##
## Attaching package: 'DescTools'
##
## The following objects are masked from 'package:caret':
##
##     MAE, RMSE
```

Import dataset

```
raw_df <- read_excel("41598_2018_29592_MOESM2_ESM.xlsx")
```

```
## New names:
## • `` -> `...1`
## • `` -> `...2`
## • `` -> `...3`
## • `` -> `...4`
## • `` -> `...5`
## • `` -> `...6`
## • `` -> `...7`
## • `` -> `...8`
## • `` -> `...9`
## • `` -> `...10`
## • `` -> `...11`
## • `` -> `...12`
## • `` -> `...13`
## • `` -> `...14`
```

```
sex_labels <- unlist(raw_df[3, 17:136])
sex <- as.factor(sex_labels)

df <- read_excel("41598_2018_29592_MOESM2_ESM.xlsx", skip = 6)
```

```
## New names:
```

```
## • `` -> `...1`
## • `` -> `...2`
## • `` -> `...3`
## • `` -> `...4`
## • `` -> `...5`
## • `` -> `...6`
## • `` -> `...7`
## • `` -> `...8`
## • `` -> `...9`
## • `` -> `...10`
## • `` -> `...12`
## • `` -> `...14`
## • `1 - Male` -> `1 - Male...16`
## • `1 - Male` -> `1 - Male...17`
## • `1 - Male` -> `1 - Male...18`
## • `1 - Male` -> `1 - Male...19`
## • `1 - Male` -> `1 - Male...20`
## • `1 - Male` -> `1 - Male...21`
## • `1 - Male` -> `1 - Male...22`
## • `1 - Male` -> `1 - Male...23`
## • `1 - Male` -> `1 - Male...24`
## • `1 - Male` -> `1 - Male...25`
## • `1 - Male` -> `1 - Male...26`
## • `1 - Male` -> `1 - Male...27`
## • `1 - Male` -> `1 - Male...28`
## • `1 - Male` -> `1 - Male...29`
## • `1 - Male` -> `1 - Male...30`
## • `1 - Male` -> `1 - Male...31`
## • `1 - Male` -> `1 - Male...32`
## • `1 - Male` -> `1 - Male...33`
## • `1 - Male` -> `1 - Male...34`
## • `1 - Male` -> `1 - Male...35`
## • `1 - Male` -> `1 - Male...36`
## • `1 - Male` -> `1 - Male...37`
## • `1 - Male` -> `1 - Male...38`
## • `1 - Male` -> `1 - Male...39`
## • `1 - Male` -> `1 - Male...40`
## • `1 - Male` -> `1 - Male...41`
## • `1 - Male` -> `1 - Male...42`
## • `1 - Male` -> `1 - Male...43`
## • `1 - Male` -> `1 - Male...44`
## • `1 - Male` -> `1 - Male...45`
## • `1 - Male` -> `1 - Male...46`
## • `1 - Male` -> `1 - Male...47`
## • `1 - Male` -> `1 - Male...48`
## • `1 - Male` -> `1 - Male...49`
## • `1 - Male` -> `1 - Male...50`
## • `1 - Male` -> `1 - Male...51`
## • `1 - Male` -> `1 - Male...52`
## • `1 - Male` -> `1 - Male...53`
## • `1 - Male` -> `1 - Male...54`
## • `1 - Male` -> `1 - Male...55`
## • `1 - Male` -> `1 - Male...56`
## • `1 - Male` -> `1 - Male...57`
## • `1 - Male` -> `1 - Male...58`
## • `1 - Male` -> `1 - Male...59`
## • `1 - Male` -> `1 - Male...60`
## • `1 - Male` -> `1 - Male...61`
## • `1 - Male` -> `1 - Male...62`
## • `1 - Male` -> `1 - Male...63`
## • `1 - Male` -> `1 - Male...64`
## • `1 - Male` -> `1 - Male...65`
## • `1 - Male` -> `1 - Male...66`
## • `1 - Male` -> `1 - Male...67`
## • `1 - Male` -> `1 - Male...68`
## • `1 - Male` -> `1 - Male...69`
## • `1 - Male` -> `1 - Male...70`
## • `1 - Male` -> `1 - Male...71`
## • `1 - Male` -> `1 - Male...72`
## • `1 - Male` -> `1 - Male...73`
## • `1 - Male` -> `1 - Male...74`
## • `1 - Male` -> `1 - Male...75`
## • `2 - Female` -> `2 - Female...76`
## • `2 - Female` -> `2 - Female...77`
## • `2 - Female` -> `2 - Female...78`
## • `2 - Female` -> `2 - Female...79`
## • `2 - Female` -> `2 - Female...80`
## • `2 - Female` -> `2 - Female...81`
## • `2 - Female` -> `2 - Female...82`
## • `2 - Female` -> `2 - Female...83`
## • `2 - Female` -> `2 - Female...84`
## • `2 - Female` -> `2 - Female...85`
## • `2 - Female` -> `2 - Female...86`
## • `2 - Female` -> `2 - Female...87`
## • `2 - Female` -> `2 - Female...88`
## • `2 - Female` -> `2 - Female...89`
## • `2 - Female` -> `2 - Female...90`
## • `2 - Female` -> `2 - Female...91`
## • `2 - Female` -> `2 - Female...92`
## • `2 - Female` -> `2 - Female...93`
## • `2 - Female` -> `2 - Female...94`
## • `2 - Female` -> `2 - Female...95`
## • `2 - Female` -> `2 - Female...96`
## • `2 - Female` -> `2 - Female...97`
## • `2 - Female` -> `2 - Female...98`
## • `2 - Female` -> `2 - Female...99`
## • `2 - Female` -> `2 - Female...100`
## • `2 - Female` -> `2 - Female...101`
## • `2 - Female` -> `2 - Female...102`
## • `2 - Female` -> `2 - Female...103`
## • `2 - Female` -> `2 - Female...104`
## • `2 - Female` -> `2 - Female...105`
## • `2 - Female` -> `2 - Female...106`
## • `2 - Female` -> `2 - Female...107`
## • `2 - Female` -> `2 - Female...108`
## • `2 - Female` -> `2 - Female...109`
## • `2 - Female` -> `2 - Female...110`
## • `2 - Female` -> `2 - Female...111`
## • `2 - Female` -> `2 - Female...112`
## • `2 - Female` -> `2 - Female...113`
## • `2 - Female` -> `2 - Female...114`
## • `2 - Female` -> `2 - Female...115`
## • `2 - Female` -> `2 - Female...116`
## • `2 - Female` -> `2 - Female...117`
## • `2 - Female` -> `2 - Female...118`
## • `2 - Female` -> `2 - Female...119`
## • `2 - Female` -> `2 - Female...120`
## • `2 - Female` -> `2 - Female...121`
## • `2 - Female` -> `2 - Female...122`
## • `2 - Female` -> `2 - Female...123`
## • `2 - Female` -> `2 - Female...124`
## • `2 - Female` -> `2 - Female...125`
## • `2 - Female` -> `2 - Female...126`
## • `2 - Female` -> `2 - Female...127`
## • `2 - Female` -> `2 - Female...128`
## • `2 - Female` -> `2 - Female...129`
## • `2 - Female` -> `2 - Female...130`
## • `2 - Female` -> `2 - Female...131`
## • `2 - Female` -> `2 - Female...132`
## • `2 - Female` -> `2 - Female...133`
## • `2 - Female` -> `2 - Female...134`
## • `2 - Female` -> `2 - Female...135`
## • `2 - Female` -> `2 - Female...136`
```

```
glimpse(df)
```

```
## Rows: 415
## Columns: 136
## $ ...1                        <chr> "index", "1", "2", "3", "4", "5", "6", "7",…
## $ ...2                        <chr> "Inchikey", "SRBFZHDQGSBBOR-IOVATXLUSA-N", …
## $ ...3                        <chr> "ret.index", "543267", "590775", "589278", …
## $ ...4                        <chr> "quant mz", "103", "189", "333", "217", "35…
## $ ...5                        <chr> "BB id", "169", "17400", "3470", "5857", "1…
## $ ...6                        <chr> "PubChem", "135191", "10264", "6602431", "6…
## $ ...7                        <chr> "KEGG", "C00181", "C02341", "C00502", "C003…
## $ ...8                        <chr> "knownORunknown", "known", "known", "known"…
## $ ...9                        <chr> "PLS-DA VIP Score", "1.2817599999999998", "…
## $ ...10                       <chr> "RSD Pooled QC", "0.21743276163921682", "0.…
## $ `Male vs Female`            <chr> "p_value", "1.87244428976912E-2", "0.919478…
## $ ...12                       <chr> "p_value_adj", "0.107665546661724", "0.9564…
## $ `median Male/median Femal` <chr> "FoldChange", "0.69725822693591988", "1.047…
## $ ...14                      <chr> "log2FoldChange", "-0.36059945342486499", "…
## $ treatment                  <chr> "label", "xylose", "xylonic acid isomer", "…
## $ `1 - Male...16`            <chr> "M1_001", "19537", "685", "534", "4766", "3…
## $ `1 - Male...17`            <chr> "M2_002", "6619", "607", "58", "4525", "28"…
## $ `1 - Male...18`            <chr> "M3_003", "20015", "1371", "176", "8679", "…
## $ `1 - Male...19`            <chr> "M4_004", "13599", "2057", "1722", "23461",…
## $ `1 - Male...20`            <chr> "M5_005", "21525", "1509", "597", "8231", "…
## $ `1 - Male...21`            <chr> "M6_006", "11649", "1187", "691", "5288", "…
## $ `1 - Male...22`            <chr> "M7_007", "41682", "1681", "630", "10392", …
## $ `1 - Male...23`            <chr> "M8_008", "31105", "912", "168", "4884", "7…
## $ `1 - Male...24`            <chr> "M9_009", "18169", "940", "800", "5900", "4…
## $ `1 - Male...25`            <chr> "M10_010", "64254", "4816", "1571", "6805",…
## $ `1 - Male...26`            <chr> "M11_011", "49394", "1426", "556", "7627", …
## $ `1 - Male...27`            <chr> "M12_012", "21989", "1347", "648", "24960",…
## $ `1 - Male...28`            <chr> "M13_013", "21908", "1454", "584", "24409",…
## $ `1 - Male...29`            <chr> "M14_014", "31015", "1399", "437", "12902",…
## $ `1 - Male...30`            <chr> "M15_015", "6137", "437", "172", "4718", "2…
## $ `1 - Male...31`            <chr> "M16_016", "64278", "1251", "460", "13487",…
## $ `1 - Male...32`            <chr> "M19_017", "28667", "2713", "361", "24524",…
## $ `1 - Male...33`            <chr> "M25_018", "13701", "1426", "103", "5171", …
## $ `1 - Male...34`            <chr> "M26_019", "85301", "1324", "304", "13610",…
## $ `1 - Male...35`            <chr> "M27_020", "17763", "1290", "195", "5099", …
## $ `1 - Male...36`            <chr> "M28_021", "12335", "1187", "311", "7043", …
## $ `1 - Male...37`            <chr> "M29_022", "11820", "1655", "280", "7592", …
## $ `1 - Male...38`            <chr> "M30_023", "11597", "5584", "137", "4677", …
## $ `1 - Male...39`            <chr> "M31_024", "43870", "1904", "159", "292573"…
## $ `1 - Male...40`            <chr> "M32_025", "99634", "2148", "723", "9605", …
## $ `1 - Male...41`            <chr> "M33_026", "23361", "1853", "791", "2358", …
## $ `1 - Male...42`            <chr> "M34_027", "27391", "1311", "378", "9844", …
## $ `1 - Male...43`            <chr> "M35_028", "19707", "958", "145", "6664", "…
## $ `1 - Male...44`            <chr> "M36_029", "19578", "1426", "222", "8813", …
## $ `1 - Male...45`            <chr> "M37_030", "69364", "1661", "423", "11889",…
## $ `1 - Male...46`            <chr> "M38_031", "26924", "2012", "1112", "7238",…
## $ `1 - Male...47`            <chr> "M39C_032", "19304", "1516", "329", "4464",…
## $ `1 - Male...48`            <chr> "M38C_033", "16851", "971", "230", "6012", …
## $ `1 - Male...49`            <chr> "M41_034", "26777", "2083", "330", "6137", …
## $ `1 - Male...50`            <chr> "M42_035", "31988", "2286", "921", "5853", …
## $ `1 - Male...51`            <chr> "M43_036", "22082", "1404", "402", "9731", …
## $ `1 - Male...52`            <chr> "M44_037", "25969", "770", "733", "9075", "…
## $ `1 - Male...53`            <chr> "M45_038", "22631", "1501", "146", "6569", …
## $ `1 - Male...54`            <chr> "M46_039", "6895", "1229", "350", "8359", "…
## $ `1 - Male...55`            <chr> "M47_040", "7945", "2317", "527", "2445", "…
## $ `1 - Male...56`            <chr> "M48_041", "46258", "2201", "432", "12879",…
## $ `1 - Male...57`            <chr> "M49_042", "32313", "1969", "923", "7531", …
## $ `1 - Male...58`            <chr> "M50_043", "33191", "474", "1156", "11364",…
## $ `1 - Male...59`            <chr> "M51_044", "27520", "2273", "578", "7694", …
## $ `1 - Male...60`            <chr> "M52_045", "20968", "455", "367", "9312", "…
## $ `1 - Male...61`            <chr> "M53_046", "23805", "1754", "352", "8930", …
## $ `1 - Male...62`            <chr> "M54_047", "29822", "1805", "694", "11447",…
## $ `1 - Male...63`            <chr> "M55_048", "11049", "1927", "546", "5333", …
## $ `1 - Male...64`            <chr> "M56_049", "15787", "1413", "182", "3473", …
## $ `1 - Male...65`            <chr> "M57_050", "26463", "1791", "342", "11472",…
## $ `1 - Male...66`            <chr> "M59_051", "27187", "1697", "1349", "4936",…
## $ `1 - Male...67`            <chr> "M60_052", "22953", "327", "319", "1968", "…
## $ `1 - Male...68`            <chr> "M61_053", "8921", "1968", "352", "11250", …
## $ `1 - Male...69`            <chr> "M62_054", "23322", "1846", "410", "13670",…
## $ `1 - Male...70`            <chr> "M63_055", "7730", "1585", "647", "2934", "…
## $ `1 - Male...71`            <chr> "M64_056", "34764", "1795", "245", "10920",…
## $ `1 - Male...72`            <chr> "M65_057", "19825", "1605", "785", "2955", …
## $ `1 - Male...73`            <chr> "M66_058", "26585", "1884", "308", "6479", …
## $ `1 - Male...74`            <chr> "M68_059", "33453", "1406", "283", "8139", …
## $ `1 - Male...75`            <chr> "M69_060", "21094", "1927", "260", "7171", …
## $ `2 - Female...76`          <chr> "F1_061", "207654", "2272", "802", "8907", …
## $ `2 - Female...77`          <chr> "F2_062", "138959", "2283", "304", "6531", …
## $ `2 - Female...78`          <chr> "F3_063", "44637", "1158", "639", "4529", "…
## $ `2 - Female...79`          <chr> "F4_064", "47100", "1383", "369", "12162", …
## $ `2 - Female...80`          <chr> "F5_065", "43956", "1041", "255", "7904", "…
## $ `2 - Female...81`          <chr> "F6_066", "24287", "1983", "855", "8075", "…
## $ `2 - Female...82`          <chr> "F7_067", "44540", "1758", "517", "11145", …
## $ `2 - Female...83`          <chr> "F8_068", "20695", "1267", "272", "12828", …
## $ `2 - Female...84`          <chr> "F9_069", "40152", "1765", "274", "10461", …
## $ `2 - Female...85`          <chr> "F13_070", "23179", "1173", "335", "6387", …
## $ `2 - Female...86`          <chr> "F14_071", "9091", "1238", "283", "8957", "…
## $ `2 - Female...87`          <chr> "F15_072", "34901", "1684", "714", "2412", …
## $ `2 - Female...88`          <chr> "F17_073", "16010", "1390", "332", "5117", …
## $ `2 - Female...89`          <chr> "F18_074", "33081", "1279", "465", "18314",…
## $ `2 - Female...90`          <chr> "F19_075", "25998", "1822", "282", "19522",…
## $ `2 - Female...91`          <chr> "F20_076", "41107", "4023", "530", "8338", …
## $ `2 - Female...92`          <chr> "F21_077", "20267", "1631", "728", "1495", …
## $ `2 - Female...93`          <chr> "F22_078", "36248", "1427", "291", "9259", …
## $ `2 - Female...94`          <chr> "F23_079", "28616", "1013", "288", "10373",…
## $ `2 - Female...95`          <chr> "F24_080", "35872", "1025", "399", "4675", …
## $ `2 - Female...96`          <chr> "F25C_081", "10802", "1465", "306", "12318"…
## $ `2 - Female...97`          <chr> "F27C_082", "44126", "1365", "593", "4528",…
## $ `2 - Female...98`          <chr> "F28C_083", "54269", "1962", "566", "5922",…
## $ `2 - Female...99`          <chr> "F29C_084", "357540", "1708", "834", "7701"…
## $ `2 - Female...100`         <chr> "F30_085", "38477", "1575", "328", "7569", …
## $ `2 - Female...101`         <chr> "F31_086", "40909", "2075", "946", "8371", …
## $ `2 - Female...102`         <chr> "F32_087", "74415", "917", "597", "7763", "…
## $ `2 - Female...103`         <chr> "F33_088", "43431", "3299", "864", "12328",…
## $ `2 - Female...104`         <chr> "F34_089", "80439", "1264", "359", "7527", …
## $ `2 - Female...105`         <chr> "F35C_090", "34328", "1134", "352", "8894",…
## $ `2 - Female...106`         <chr> "F36C_091", "19135", "2424", "358", "5976",…
## $ `2 - Female...107`         <chr> "F37_092", "24238", "1451", "513", "9855", …
## $ `2 - Female...108`         <chr> "F38_093", "22072", "1775", "566", "9779", …
## $ `2 - Female...109`         <chr> "F39_094", "29264", "2206", "499", "4779", …
## $ `2 - Female...110`         <chr> "F43_095", "47070", "1764", "356", "6239", …
## $ `2 - Female...111`         <chr> "F44_096", "18263", "1723", "284", "7309", …
## $ `2 - Female...112`         <chr> "F45_097", "16612", "1448", "157", "11034",…
## $ `2 - Female...113`         <chr> "F47_098", "155085", "1282", "402", "148486…
## $ `2 - Female...114`         <chr> "F48_099", "13783", "1178", "238", "6571", …
## $ `2 - Female...115`         <chr> "F49_100", "44815", "1391", "416", "8434", …
## $ `2 - Female...116`         <chr> "F50_101", "83091", "888", "964", "10702", …
## $ `2 - Female...117`         <chr> "F51_102", "18811", "1431", "381", "9231", …
## $ `2 - Female...118`         <chr> "F52_103", "18699", "1374", "223", "8527", …
## $ `2 - Female...119`         <chr> "103_104", "17372", "1644", "599", "7258", …
## $ `2 - Female...120`         <chr> "105_105", "18347", "1422", "406", "6321", …
## $ `2 - Female...121`         <chr> "108_106", "15163", "1299", "37", "7601", "…
## $ `2 - Female...122`         <chr> "111_107", "42048", "2153", "576", "16220",…
## $ `2 - Female...123`         <chr> "116_108", "33530", "1209", "496", "2713", …
## $ `2 - Female...124`         <chr> "120_109", "13645", "2249", "892", "729", "…
## $ `2 - Female...125`         <chr> "121_110", "66094", "4184", "874", "14473",…
## $ `2 - Female...126`         <chr> "122_111", "13053", "1648", "1135", "6141",…
## $ `2 - Female...127`         <chr> "123_112", "27924", "1472", "303", "10796",…
## $ `2 - Female...128`         <chr> "124_113", "28929", "1825", "323", "10386",…
## $ `2 - Female...129`         <chr> "125_114", "69691", "78", "498", "13457", "…
## $ `2 - Female...130`         <chr> "126_115", "16387", "1434", "285", "3818", …
## $ `2 - Female...131`         <chr> "127_116", "15305", "2168", "351", "10821",…
## $ `2 - Female...132`         <chr> "128_117", "13898", "1726", "136", "12544",…
## $ `2 - Female...133`         <chr> "129_118", "240411", "1941", "627", "7634",…
## $ `2 - Female...134`         <chr> "130_119", "16822", "1495", "368", "8759", …
## $ `2 - Female...135`         <chr> "131_120", "17116", "1226", "131", "12496",…
## $ `2 - Female...136`         <chr> "132_121", "19975", "386", "381", "3202", "…
```

Cleaning and Data Transformation

```
# organize names
# first df line has de colnames
colnames(df) <- as.character(unlist(df[1, ]))
df <- df[-1, , drop = FALSE]

# Convert measurement columns (17:end) to numeric explicitly
measure_cols <- 17:ncol(df)

for (i in measure_cols) {
  df[[i]] <- as.numeric(df[[i]])
}



X <- as.data.frame(df[, measure_cols])

colnames(X) <- paste0("S", seq_len(ncol(X)))
# number of metabolites and samples
if ("Metabolite" %in% colnames(df)) {
  rownames(X) <- df$Metabolite
}

dim(X)
```

```
## [1] 414 120
```

It matches with the article 120 samples (60 males + 60 females)

Summary statistics

```
summary(X)
```

```
##        S1                  S2                  S3                 S4         
##  Min.   :     11.0   Min.   :     49.0   Min.   :    35.0   Min.   :    55.0  
##  1st Qu.:    361.8   1st Qu.:    974.8   1st Qu.:   928.5   1st Qu.:   886.2  
##  Median :    931.5   Median :   2567.0   Median :  2413.5   Median :  2200.0  
##  Mean   :  10396.4   Mean   :  17884.3   Mean   : 15434.6   Mean   : 16240.0  
##  3rd Qu.:   2820.5   3rd Qu.:   7546.2   3rd Qu.:  7103.8   3rd Qu.:  6561.8  
##  Max.   :1159279.0   Max.   :1312984.0   Max.   :887773.0   Max.   :846476.0  
##        S5                  S6                 S7                  S8        
##  Min.   :     57   Min.   :    49   Min.   :     63   Min.   :    36  
##  1st Qu.:   1034   1st Qu.:  1218   1st Qu.:    871   1st Qu.:  1084  
##  Median :   2678   Median :  3170   Median :   2626   Median :  2892  
##  Mean   :  20265   Mean   : 17583   Mean   :  19948   Mean   : 15306  
##  3rd Qu.:   7338   3rd Qu.:  9827   3rd Qu.:   8472   3rd Qu.:  7588  
##  Max.   :1331535   Max.   :756697   Max.   :1085291   Max.   :654485  
##        S9                 S10                 S11                S12        
##  Min.   :    45.0   Min.   :    51.0   Min.   :    42   Min.   :    32  
##  1st Qu.:   883.2   1st Qu.:   999.8   1st Qu.:   854   1st Qu.:   836  
##  Median :  2420.0   Median :  2603.5   Median :  2386   Median :  2347  
##  Mean   : 16003.7   Mean   : 16155.8   Mean   : 15655   Mean   : 14944  
##  3rd Qu.:  8532.5   3rd Qu.:  7937.0   3rd Qu.:  7422   3rd Qu.:  7623  
##  Max.   :613979.0   Max.   :979339.0   Max.   :662597   Max.   :600872  
##       S13                 S14                 S15                S16        
##  Min.   :     53.0   Min.   :     23.0   Min.   :    31   Min.   :    32  
##  1st Qu.:    898.2   1st Qu.:    425.2   1st Qu.:   652   1st Qu.:   814  
##  Median :   2714.5   Median :   1077.5   Median :  1932   Median :  2408  
##  Mean   :  20509.0   Mean   :  12255.6   Mean   : 14991   Mean   : 15781  
##  3rd Qu.:   8495.5   3rd Qu.:   2973.8   3rd Qu.:  5742   3rd Qu.:  7016  
##  Max.   :1223172.0   Max.   :1357588.0   Max.   :664051   Max.   :923306  
##       S17                S18                 S19                 S20        
##  Min.   :     45   Min.   :    43.0   Min.   :     37.0   Min.   :    57  
##  1st Qu.:    802   1st Qu.:   872.5   1st Qu.:    760.8   1st Qu.:  1001  
##  Median :   2398   Median :  2497.0   Median :   2002.5   Median :  2714  
##  Mean   :  22298   Mean   : 14868.9   Mean   :  16331.1   Mean   : 16196  
##  3rd Qu.:   8043   3rd Qu.:  7397.0   3rd Qu.:   5930.2   3rd Qu.:  6979  
##  Max.   :1412185   Max.   :615658.0   Max.   :1047043.0   Max.   :718072  
##       S21                 S22                S23                 S24         
##  Min.   :     63.0   Min.   :    42   Min.   :    19.0   Min.   :     35.0  
##  1st Qu.:    829.8   1st Qu.:  1020   1st Qu.:   517.5   1st Qu.:    997.5  
##  Median :   2398.0   Median :  2610   Median :  1475.0   Median :   2618.5  
##  Mean   :  17775.5   Mean   : 16756   Mean   : 12719.6   Mean   :  17888.5  
##  3rd Qu.:   7253.5   3rd Qu.:  7651   3rd Qu.:  4454.2   3rd Qu.:   7489.2  
##  Max.   :1026464.0   Max.   :849364   Max.   :704274.0   Max.   :1002880.0  
##       S25               S26                 S27                S28          
##  Min.   :     57   Min.   :     57.0   Min.   :    38.0   Min.   :    55.0  
##  1st Qu.:   1102   1st Qu.:    843.2   1st Qu.:   769.5   1st Qu.:   995.8  
##  Median :   2633   Median :   2615.0   Median :  2272.0   Median :  2581.0  
##  Mean   :  17566   Mean   :  28104.5   Mean   : 15048.0   Mean   : 15940.6  
##  3rd Qu.:   7642   3rd Qu.:   9100.0   3rd Qu.:  7267.2   3rd Qu.:  8733.2  
##  Max.   :1243970   Max.   :2448882.0   Max.   :590052.0   Max.   :795015.0  
##       S29               S30               S31                S32         
##  Min.   :     30   Min.   :     61   Min.   :    28.0   Min.   :    43  
##  1st Qu.:   1006   1st Qu.:   1034   1st Qu.:   835.8   1st Qu.:   634  
##  Median :   2598   Median :   3132   Median :  2850.0   Median :  1934  
##  Mean   :  21812   Mean   :  20275   Mean   : 19758.3   Mean   : 14803  
##  3rd Qu.:   8412   3rd Qu.:   8966   3rd Qu.:  8962.2   3rd Qu.:  6673  
##  Max.   :1474685   Max.   :1033848   Max.   :652214.0   Max.   :611685  
##       S33              S34                 S35              S36        
##  Min.   :    57   Min.   :     49.0   Min.   :    69   Min.   :    40  
##  1st Qu.:  1094   1st Qu.:    875.5   1st Qu.:  1199   1st Qu.:  1023  
##  Median :  3131   Median :   2439.5   Median :  2678   Median :  3118  
##  Mean   : 18916   Mean   :  20133.2   Mean   : 17923   Mean   : 20058  
##  3rd Qu.:  8026   3rd Qu.:   8751.2   3rd Qu.:  8075   3rd Qu.:  9702  
##  Max.   :988258   Max.   :1156623.0   Max.   :750958   Max.   :903330  
##       S37                 S38                 S39              S40        
##  Min.   :     56.0   Min.   :     30.0   Min.   :    27   Min.   :    49  
##  1st Qu.:    792.8   1st Qu.:    913.2   1st Qu.:  1198   1st Qu.:  1165  
##  Median :   2520.0   Median :   2379.5   Median :  2990   Median :  3186  
##  Mean   :  21491.9   Mean   :  20571.5   Mean   : 18004   Mean   : 16763  
##  3rd Qu.:   9072.8   3rd Qu.:   7049.2   3rd Qu.:  8534   3rd Qu.:  9282  
##  Max.   :1382326.0   Max.   :1343658.0   Max.   :930312   Max.   :622009  
##       S41                S42               S43              S44           
##  Min.   :    67.0   Min.   :     60   Min.   :    58   Min.   :     47.0  
##  1st Qu.:   933.2   1st Qu.:   1284   1st Qu.:  1288   1st Qu.:    930.2  
##  Median :  2570.5   Median :   3593   Median :  3736   Median :   2621.5  
##  Mean   : 18537.8   Mean   :  21514   Mean   : 19702   Mean   :  20043.4  
##  3rd Qu.:  7634.5   3rd Qu.:  10507   3rd Qu.: 10996   3rd Qu.:   9137.2  
##  Max.   :958901.0   Max.   :1050715   Max.   :978076   Max.   :1062304.0  
##       S45                S46                S47               S48         
##  Min.   :    45.0   Min.   :    27.0   Min.   :     71   Min.   :     14  
##  1st Qu.:   928.5   1st Qu.:   983.2   1st Qu.:   1028   1st Qu.:    367  
##  Median :  2550.0   Median :  2764.0   Median :   2775   Median :   1138  
##  Mean   : 19243.2   Mean   : 18173.2   Mean   :  22158   Mean   :  10432  
##  3rd Qu.:  8023.2   3rd Qu.:  8656.0   3rd Qu.:   8392   3rd Qu.:   3546  
##  Max.   :894751.0   Max.   :814803.0   Max.   :1384220   Max.   :1378594  
##       S49              S50              S51                S52         
##  Min.   :    33   Min.   :    37   Min.   :    11.0   Min.   :     44  
##  1st Qu.:   948   1st Qu.:  1020   1st Qu.:   327.8   1st Qu.:   1118  
##  Median :  2785   Median :  2486   Median :   996.0   Median :   2982  
##  Mean   : 19689   Mean   : 16457   Mean   :  9556.1   Mean   :  20580  
##  3rd Qu.:  7976   3rd Qu.:  6876   3rd Qu.:  2873.2   3rd Qu.:   8585  
##  Max.   :928126   Max.   :895948   Max.   :920842.0   Max.   :1012064  
##       S53               S54               S55              S56
##  Min.   :     59   Min.   :     81   Min.   :    53   Min.   :     37.0
##  1st Qu.:   1013   1st Qu.:   1026   1st Qu.:  1404   1st Qu.:    920.8
##  Median :   3070   Median :   2434   Median :  3426   Median :   2498.0
##  Mean   :  21624   Mean   :  19055   Mean   : 17594   Mean   :  19834.2
##  3rd Qu.:   9218   3rd Qu.:   7902   3rd Qu.:  9361   3rd Qu.:   7251.0
##  Max.   :1159151   Max.   :1158978   Max.   :897174   Max.   :1274193.0
##       S57               S58               S59                S60
##  Min.   :     41   Min.   :     48   Min.   :    55.0   Min.   :    32
##  1st Qu.:   1067   1st Qu.:   1032   1st Qu.:   747.8   1st Qu.:   882
##  Median :   2767   Median :   2590   Median :  2419.0   Median :  2614
##  Mean   :  19441   Mean   :  19249   Mean   : 17639.2   Mean   : 15477
##  3rd Qu.:   9708   3rd Qu.:   8131   3rd Qu.:  8217.2   3rd Qu.:  7118
##  Max.   :1521963   Max.   :1204027   Max.   :857508.0   Max.   :481398
##       S61                S62               S63                S64
##  Min.   :    30.0   Min.   :    14   Min.   :      44.0   Min.   :     45
##  1st Qu.:   756.5   1st Qu.:   550   1st Qu.:     810.5   1st Qu.:    697
##  Median :  2116.5   Median :  1470   Median :    2586.0   Median :   1960
##  Mean   : 14449.6   Mean   : 10995   Mean   :   19064.3   Mean   :  16921
##  3rd Qu.:  6930.0   3rd Qu.:  4598   3rd Qu.:    7443.8   3rd Qu.:   5548
##  Max.   :654335.0   Max.   :473352   Max.   :1252073.0   Max.   :1166064
##       S65              S66                S67                S68
##  Min.   :    37   Min.   :     36   Min.   :    43.0   Min.   :    47
##  1st Qu.:  1117   1st Qu.:   1033   1st Qu.:   809.8   1st Qu.:  1122
##  Median :  2720   Median :   2960   Median :  2232.0   Median :  3137
##  Mean   : 16367   Mean   :  20217   Mean   : 16391.2   Mean   : 16657
##  3rd Qu.:  8012   3rd Qu.:   9554   3rd Qu.:  7177.2   3rd Qu.:  9066
##  Max.   :645905   Max.   :1239046   Max.   :809789.0   Max.   :717275
##       S69                S70                S71                S72
##  Min.   :    46.0   Min.   :     45   Min.   :     73   Min.   :      65.0
##  1st Qu.:   893.2   1st Qu.:    947   1st Qu.:   1064   1st Qu.:     709.5
##  Median :  2427.5   Median :   2646   Median :   2829   Median :    1909.0
##  Mean   : 15771.6   Mean   :  18762   Mean   :  24544   Mean   :   18327.2
##  3rd Qu.:  6434.0   3rd Qu.:   7962   3rd Qu.:   8758   3rd Qu.:    5963.2
##  Max.   :786634.0   Max.   :1026648   Max.   :1763202   Max.   :1164459.0
##       S73               S74                 S75                 S76
##  Min.   :     89   Min.   :      41.0   Min.   :    51.0   Min.   :    94.0
##  1st Qu.:   1027   1st Qu.:     833.2   1st Qu.:   824.5   1st Qu.:   973.5
##  Median :   2436   Median :    2737.5   Median :  2282.5   Median :  2513.5
##  Mean   :  19399   Mean   :   18837.0   Mean   : 17355.3   Mean   : 15380.8
##  3rd Qu.:   7973   3rd Qu.:    8114.0   3rd Qu.:  8586.5   3rd Qu.:  7788.5
##  Max.   :1431153   Max.   :1011305.0   Max.   :817150.0   Max.   :756530.0
##       S77               S78                S79                S80
##  Min.   :     55   Min.   :    50.0   Min.   :    57.0   Min.   :    49
##  1st Qu.:    729   1st Qu.:   896.8   1st Qu.:   718.8   1st Qu.:  1025
##  Median :   2194   Median :  2277.5   Median :  1976.0   Median :  3026
##  Mean   :  18062   Mean   : 14365.5   Mean   : 16101.9   Mean   : 16095
##  3rd Qu.:   7172   3rd Qu.:  6604.8   3rd Qu.:  6489.0   3rd Qu.:  9898
##  Max.   :1236177   Max.   :727831.0   Max.   :876808.0   Max.   :585919
##       S81                S82               S83                S84
##  Min.   :    21.0   Min.   :    46   Min.   :      37.0   Min.   :    54.0
##  1st Qu.:   826.5   1st Qu.:   827   1st Qu.:    904.8   1st Qu.:   967.8
##  Median :  2425.5   Median :  1850   Median :   2553.0   Median :  2523.5
##  Mean   : 16124.8   Mean   : 15839   Mean   :  22022.1   Mean   : 16216.5
##  3rd Qu.:  6523.0   3rd Qu.:  6794   3rd Qu.:   7966.8   3rd Qu.:  7958.8
##  Max.   :766114.0   Max.   :803939   Max.   :1093912.0   Max.   :739373.0
##       S85               S86                 S87              S88
##  Min.   :     55   Min.   :     46.0   Min.   :    76   Min.   :    39
##  1st Qu.:    838   1st Qu.:    828.2   1st Qu.:  1249   1st Qu.:   777
##  Median :   2952   Median :   2569.0   Median :  3672   Median :  2065
##  Mean   :  23280   Mean   :  18416.6   Mean   : 18228   Mean   : 15579
##  3rd Qu.:   9084   3rd Qu.:   8779.8   3rd Qu.: 11428   3rd Qu.:  5805
##  Max.   :1358932   Max.   :1076740.0   Max.   :793472   Max.   :838662
##       S89              S90                S91                 S92
##  Min.   :    42   Min.   :    38.0   Min.   :      0.0   Min.   :    66.0
##  1st Qu.:  1044   1st Qu.:   678.8   1st Qu.:    701.8   1st Qu.:   989.2
##  Median :  2636   Median :  2041.5   Median :   2077.0   Median :  2734.0
##  Mean   : 15765   Mean   : 14162.8   Mean   :  16451.8   Mean   : 14734.1
##  3rd Qu.:  7137   3rd Qu.:  5630.2   3rd Qu.:   7192.8   3rd Qu.:  7427.8
##  Max.   :709939   Max.   :651143.0   Max.   :1002377.0   Max.   :485330.0
##       S93              S94              S95                 S96
##  Min.   :    53   Min.   :     0   Min.   :     42.0   Min.   :    45
##  1st Qu.:   860   1st Qu.:   941   1st Qu.:    676.5   1st Qu.:   914
##  Median :  2600   Median :  2556   Median :   1890.5   Median :  2553
##  Mean   : 17691   Mean   : 16531   Mean   :  17419.6   Mean   : 17090
##  3rd Qu.:  8764   3rd Qu.:  7706   3rd Qu.:   6036.5   3rd Qu.:  8547
##  Max.   :858142   Max.   :713027   Max.   :1147218.0   Max.   :838240
##       S97                S98                 S99              S100
##  Min.   :    23.0   Min.   :     33.0   Min.   :    46   Min.   :     33.0
##  1st Qu.:   475.8   1st Qu.:    842.8   1st Qu.:   796   1st Qu.:    890.2
##  Median :  1519.0   Median :   2379.5   Median :  2492   Median :   2497.5
##  Mean   : 11912.7   Mean   :  19292.5   Mean   : 17882   Mean   :  17162.3
##  3rd Qu.:  4719.5   3rd Qu.:   6850.2   3rd Qu.:  7372   3rd Qu.:   7196.5
##  Max.   :394734.0   Max.   :1195071.0   Max.   :937666   Max.   :1047894.0
##       S101             S102                S103              S104
##  Min.   :    41   Min.   :     86.0   Min.   :     51   Min.   :    37.0
##  1st Qu.:   961   1st Qu.:    900.5   1st Qu.:   1150   1st Qu.:   752.5
##  Median :  2355   Median :   2503.5   Median :   2639   Median :  2033.5
##  Mean   : 14641   Mean   :  21895.1   Mean   :  17280   Mean   : 14209.1
##  3rd Qu.:  6269   3rd Qu.:   7907.5   3rd Qu.:   7093   3rd Qu.:  6888.2
##  Max.   :555121   Max.   :1875999.0   Max.   :1476436   Max.   :835449.0
##       S105               S106               S107             S108
##  Min.   :    18.0   Min.   :    47.0   Min.   :    34   Min.   :    53
##  1st Qu.:   735.2   1st Qu.:   890.5   1st Qu.:   910   1st Qu.:   981
##  Median :  2122.5   Median :  3107.5   Median :  2410   Median :  2956
##  Mean   : 14972.5   Mean   : 17048.6   Mean   : 14732   Mean   : 17213
##  3rd Qu.:  7262.2   3rd Qu.:  9262.5   3rd Qu.:  7289   3rd Qu.:  7791
##  Max.   :658556.0   Max.   :681033.0   Max.   :608973   Max.   :939649
##       S109             S110                S111              S112
##  Min.   :    50   Min.   :     46.0   Min.   :     52   Min.   :    52
##  1st Qu.:   942   1st Qu.:    834.2   1st Qu.:   1023   1st Qu.:  1009
##  Median :  2816   Median :   2394.0   Median :   2802   Median :  2761
##  Mean   : 16926   Mean   :  17572.9   Mean   :  21115   Mean   : 18704  
##  3rd Qu.:  7823   3rd Qu.:   7052.0   3rd Qu.:   8988   3rd Qu.:  8308  
##  Max.   :533155   Max.   :1037573.0   Max.   :1427015   Max.   :777934  
##       S113             S114             S115             S116      
##  Min.   :    48   Min.   :    33   Min.   :    82   Min.   :    41  
##  1st Qu.:  1098   1st Qu.:  1011   1st Qu.:  1132   1st Qu.:  1107  
##  Median :  3168   Median :  2760   Median :  2595   Median :  2692  
##  Mean   : 17953   Mean   : 14737   Mean   : 16030   Mean   : 18147  
##  3rd Qu.:  8649   3rd Qu.:  7086   3rd Qu.:  8018   3rd Qu.:  7900  
##  Max.   :818457   Max.   :655562   Max.   :710631   Max.   :994444  
##       S117               S118                S119               S120          
##  Min.   :    42.0   Min.   :     45.0   Min.   :    51.0   Min.   :     21.0  
##  1st Qu.:   859.8   1st Qu.:    989.5   1st Qu.:   926.2   1st Qu.:    421.5  
##  Median :  2474.0   Median :   2849.0   Median :  2439.0   Median :   1033.0  
##  Mean   : 14971.4   Mean   :  19385.2   Mean   : 17778.0   Mean   :  10041.5  
##  3rd Qu.:  8341.2   3rd Qu.:   7703.2   3rd Qu.:  9066.8   3rd Qu.:   3205.0  
##  Max.   :655882.0   Max.   :1265896.0   Max.   :944806.0   Max.   :1565960.0
```

```
# Count total zero values
sum(X == 0, na.rm = TRUE)
```

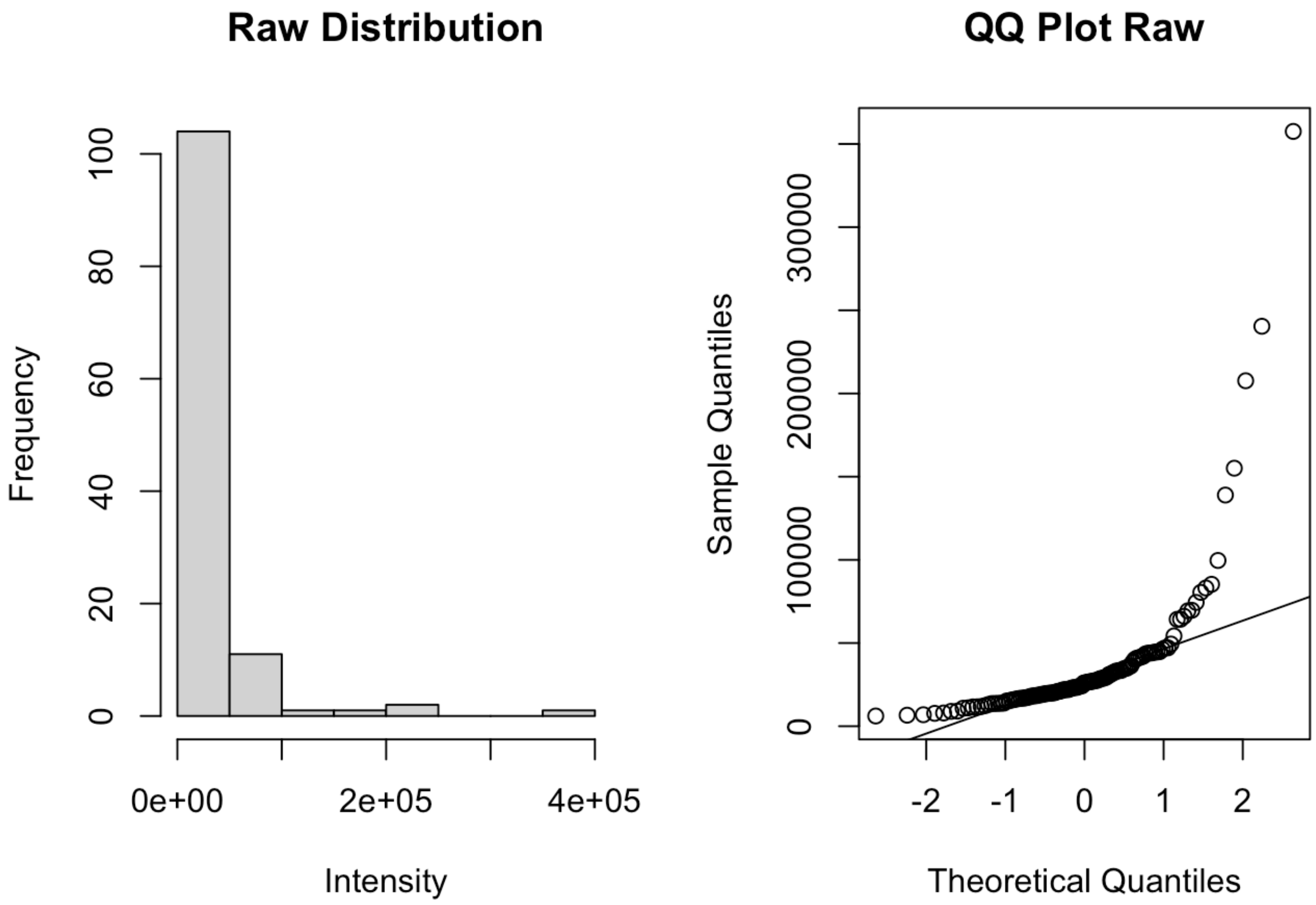
```
## [1] 2
```

```
# Missing values check
sum(is.na(X))
```

```
## [1] 0
```

Is it the data Gaussian Distributed?

```
par(mfrow=c(1,2))
hist(as.numeric(X[1,]), main="Raw Distribution", xlab="Intensity")
qqnorm(as.numeric(X[1,]), main="QQ Plot Raw")
qqline(as.numeric(X[1,]))
```

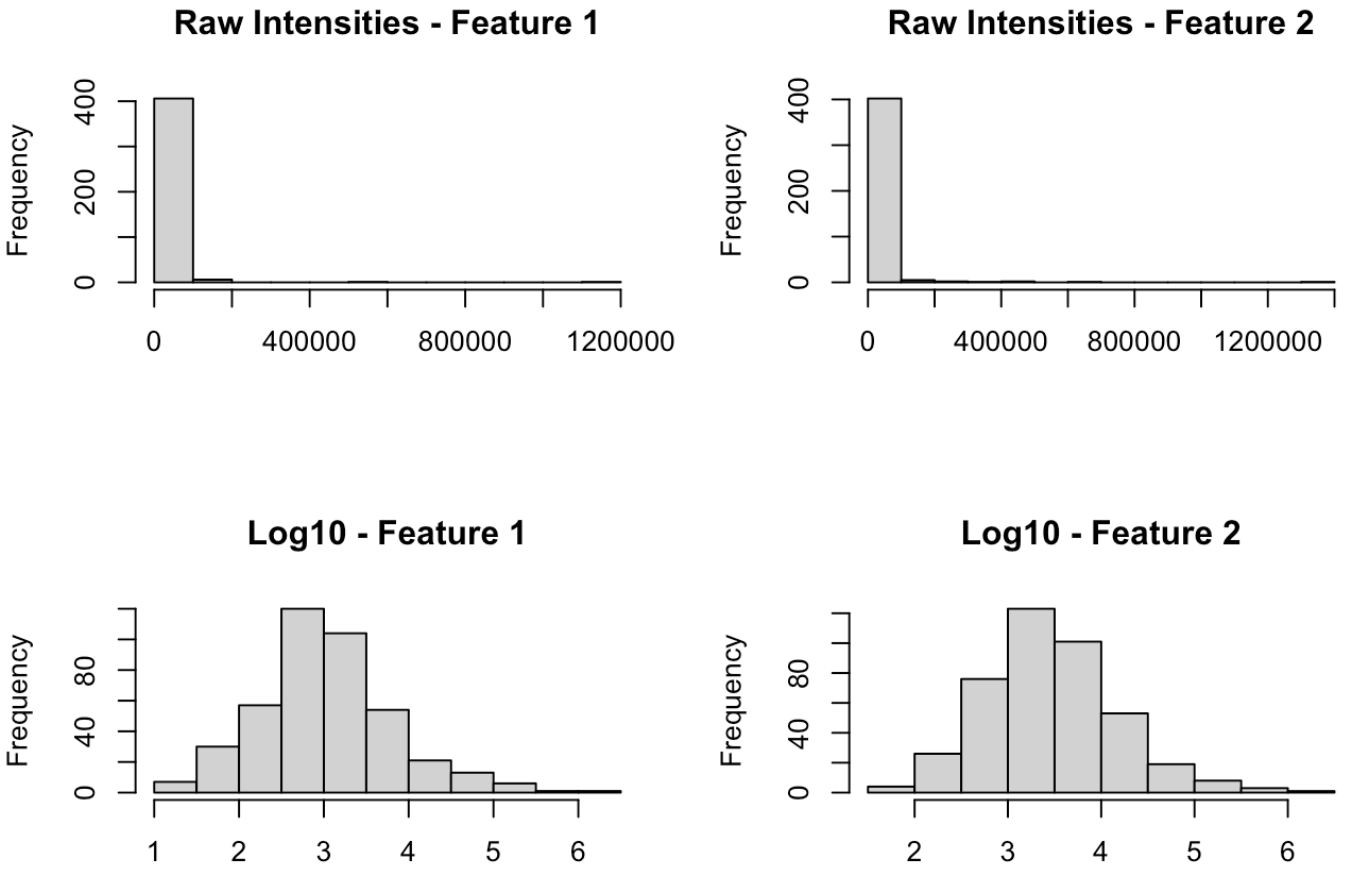


```
par(mfrow=c(1,1))
```

Histogram inspection

To assess distributional properties, histogram inspection was performed. Raw metabolite intensities showed pronounced right-skewness, supporting the need for a log-transformation to improve normality and stabilize variance prior to multivariate analysis.

```
feat1 <- as.numeric(X[[1]]) #S1
feat2 <- as.numeric(X[[2]]) #S2

par(mfrow = c(2,2))

# Raw intensity distributions
hist(feat1, main = "Raw Intensities - Feature 1", xlab = "", col = "lightgray")
hist(feat2, main = "Raw Intensities - Feature 2", xlab = "", col = "lightgray")

# After log10 transformation
hist(log10(feat1 + 1), main = "Log10 - Feature 1", xlab = "", col = "lightgray")
hist(log10(feat2 + 1), main = "Log10 - Feature 2", xlab = "", col = "lightgray")
```



```
par(mfrow = c(1,1))
```

The raw intensity histograms show strong right-skewness with extreme values. After log10 transformation, the distributions become more symmetric and less heteroscedastic, supporting the use of log-transformed and autoscaled data for PCA and PLS-DA.

Non linear

```
# Convert to standard data frame to avoid tibble recycling issues
X_nozero <- as.data.frame(X)

for(i in 1:nrow(X_nozero)){
  row_vals <- as.numeric(X_nozero[i, ])  # convert row to numeric vector

  # Safeguard: if the row is all zeros (rare but possible)
  if(all(row_vals == 0)){
    row_vals[row_vals == 0] <- 1  # placeholder before log10
  } else {
    min_val <- min(row_vals[row_vals > 0], na.rm = TRUE)
    row_vals[row_vals == 0] <- min_val / 2
  }

  X_nozero[i, ] <- row_vals  # assign back properly
}
```

PCA

A preliminary variance-based feature filtering step (top 20% most variable metabolites) was evaluated as an unsupervised method for noise reduction and dimensionality control. However, this procedure resulted in the removal of several metabolites previously identified as statistically significant in the univariate analysis, consequently reducing the number of biologically relevant features and weakening the discrimination between sexes in downstream multivariate models. Therefore, variance filtering was not retained in the final workflow to preserve subtle yet meaningful metabolic differences.

The matrix was transposed so that samples correspond to rows and metabolites to columns, which is the correct structure required for multivariate analyses such as PCA and PLS-DA, where each row must represent an independent observation:

- With <- scale(t(X_log)).

```
# Log10 transformation
X_log <- log10(X_nozero)

# Scaling for PCA (samples as rows)
X_scaled <- scale(t(X_log))

# PCA
pca <- prcomp(X_scaled, center = TRUE, scale. = FALSE)
summary(pca)
```

```
## Importance of components:
##                           PC1     PC2     PC3     PC4     PC5     PC6     PC7
## Standard deviation     8.9816 4.96339 4.88671 4.34067 3.71722 3.65416 3.42666
## Proportion of Variance 0.1948 0.05951 0.05768 0.04551 0.03338 0.03225 0.02836
## Cumulative Proportion  0.1948 0.25436 0.31204 0.35755 0.39093 0.42318 0.45154
##                            PC8     PC9    PC10    PC11    PC12    PC13    PC14
## Standard deviation     3.05556 2.93806 2.73820 2.66792 2.60252 2.49513 2.43427
## Proportion of Variance 0.02255 0.02085 0.01811 0.01719 0.01636 0.01504 0.01431
## Cumulative Proportion  0.47410 0.49495 0.51306 0.53025 0.54661 0.56165 0.57596
##                           PC15    PC16    PC17    PC18    PC19    PC20    PC21
## Standard deviation     2.38762 2.33392 2.23402 2.21841 2.16754 2.08917 2.07092
## Proportion of Variance 0.01377 0.01316 0.01206 0.01189 0.01135 0.01054 0.01036
## Cumulative Proportion  0.58973 0.60289 0.61494 0.62683 0.63818 0.64872 0.65908
##                           PC22    PC23    PC24   PC25    PC26    PC27    PC28
## Standard deviation     2.00821 1.96491 1.92119 1.9196 1.89262 1.86169 1.82722
## Proportion of Variance 0.00974 0.00933 0.00892 0.0089 0.00865 0.00837 0.00806
## Cumulative Proportion  0.66882 0.67815 0.68706 0.6960 0.70462 0.71299 0.72105
##                          PC29    PC30    PC31    PC32    PC33    PC34    PC35
## Standard deviation     1.8090 1.78058 1.74800 1.70625 1.68245 1.67253 1.64633
## Proportion of Variance 0.0079 0.00766 0.00738 0.00703 0.00684 0.00676 0.00655
## Cumulative Proportion  0.7290 0.73661 0.74399 0.75103 0.75786 0.76462 0.77117
##                           PC36    PC37    PC38    PC39    PC40    PC41    PC42
## Standard deviation     1.62664 1.60528 1.57799 1.54693 1.52758 1.51314 1.50251
## Proportion of Variance 0.00639 0.00622 0.00601 0.00578 0.00564 0.00553 0.00545
## Cumulative Proportion  0.77756 0.78378 0.78980 0.79558 0.80121 0.80675 0.81220
##                           PC43    PC44    PC45    PC46    PC47    PC48    PC49
## Standard deviation     1.47915 1.46136 1.44511 1.42020 1.40728 1.40112 1.38371
## Proportion of Variance 0.00528 0.00516 0.00504 0.00487 0.00478 0.00474 0.00462
## Cumulative Proportion  0.81748 0.82264 0.82769 0.83256 0.83734 0.84208 0.84671
##                          PC50    PC51    PC52    PC53    PC54    PC55    PC56
## Standard deviation     1.3504 1.34801 1.33024 1.32638 1.30937 1.27730 1.26135
## Proportion of Variance 0.0044 0.00439 0.00427 0.00425 0.00414 0.00394 0.00384
## Cumulative Proportion  0.8511 0.85550 0.85978 0.86403 0.86817 0.87211 0.87595
##                           PC57    PC58    PC59    PC60    PC61    PC62    PC63
## Standard deviation     1.24856 1.23645 1.22854 1.22654 1.20842 1.18012 1.16099
## Proportion of Variance 0.00377 0.00369 0.00365 0.00363 0.00353 0.00336 0.00326
## Cumulative Proportion  0.87972 0.88341 0.88705 0.89069 0.89422 0.89758 0.90084
##                           PC64    PC65    PC66    PC67    PC68    PC69    PC70
## Standard deviation     1.15572 1.14283 1.13768 1.11035 1.10403 1.08755 1.07984
## Proportion of Variance 0.00323 0.00315 0.00313 0.00298 0.00294 0.00286 0.00282
## Cumulative Proportion  0.90406 0.90722 0.91034 0.91332 0.91626 0.91912 0.92194
##                           PC71   PC72    PC73   PC74    PC75    PC76    PC77
## Standard deviation     1.07325 1.0564 1.04196 1.0383 1.01892 1.01293 1.01156
## Proportion of Variance 0.00278 0.0027 0.00262 0.0026 0.00251 0.00248 0.00247
## Cumulative Proportion  0.92472 0.9274 0.93004 0.9326 0.93515 0.93763 0.94010
##                           PC78    PC79    PC80    PC81    PC82    PC83    PC84
## Standard deviation     1.00178 0.98745 0.97119 0.96110 0.94572 0.92841 0.92163
## Proportion of Variance 0.00242 0.00236 0.00228 0.00223 0.00216 0.00208 0.00205
## Cumulative Proportion  0.94252 0.94488 0.94716 0.94939 0.95155 0.95363 0.95568
##                           PC85    PC86    PC87    PC88   PC89    PC90    PC91
## Standard deviation     0.90321 0.89679 0.88415 0.87200 0.8639 0.85119 0.83708
## Proportion of Variance 0.00197 0.00194 0.00189 0.00184 0.0018 0.00175 0.00169
## Cumulative Proportion  0.95765 0.95960 0.96148 0.96332 0.9651 0.96687 0.96857
##                           PC92    PC93    PC94    PC95    PC96    PC97    PC98
## Standard deviation     0.83220 0.82397 0.80990 0.79329 0.77982 0.77263 0.76297
## Proportion of Variance 0.00167 0.00164 0.00158 0.00152 0.00147 0.00144 0.00141
## Cumulative Proportion  0.97024 0.97188 0.97346 0.97498 0.97645 0.97789 0.97930
##                           PC99  PC100   PC101   PC102  PC103   PC104   PC105
## Standard deviation     0.75965 0.7335 0.72827 0.70693 0.7059 0.69253 0.68757
## Proportion of Variance 0.00139 0.0013 0.00128 0.00121 0.0012 0.00116 0.00114
## Cumulative Proportion  0.98069 0.9820 0.98328 0.98448 0.9857 0.98684 0.98799
##                          PC106   PC107   PC108   PC109   PC110   PC111   PC112
## Standard deviation     0.67829 0.66059 0.65036 0.63401 0.61801 0.60428 0.60118
## Proportion of Variance 0.00111 0.00105 0.00102 0.00097 0.00092 0.00088 0.00087
## Cumulative Proportion  0.98910 0.99015 0.99117 0.99214 0.99307 0.99395 0.99482
##                          PC113   PC114   PC115   PC116   PC117   PC118  PC119
## Standard deviation     0.58644 0.58182 0.56989 0.55374 0.54339 0.53442 0.4991
## Proportion of Variance 0.00083 0.00082 0.00078 0.00074 0.00071 0.00069 0.0006
## Cumulative Proportion  0.99565 0.99647 0.99725 0.99800 0.99871 0.99940 1.0000
##                            PC120
## Standard deviation     3.989e-15
## Proportion of Variance 0.000e+00
## Cumulative Proportion  1.000e+00
```

- Summary statistics show very different scales and non-normal distributions
- A few zero values detected → replaced with half of minimum value
  → required for log10 transformation
- Applied log10(x) to reduce skewness and stabilize variance
- Autoscaling (mean-center + unit variance) to make metabolites comparable

```
# Sanity check: do sex labels align with PCA samples?
sex_row <- as.character(raw_df[3, ])
sample_names_raw <- sex_row[sex_row %in% c("Male", "Female")]
sample_cols <- colnames(X)

sex <- factor(sample_names_raw[1:length(sample_cols)])

length(sex)
```

```
## [1] 120
```

```
table(sex)
```

```
## sex
## Female   Male
##     60     60
```

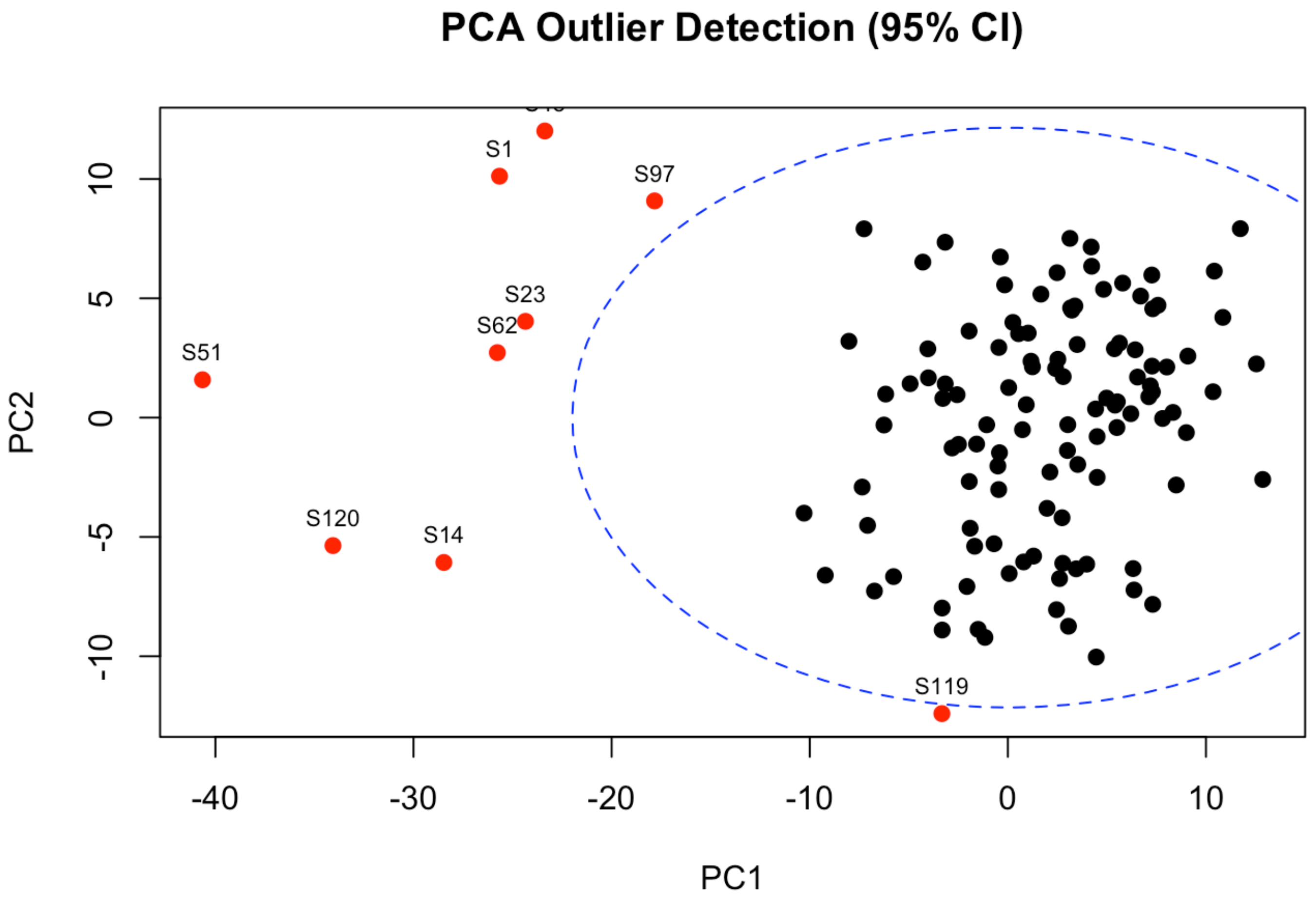
```
all(!is.na(sex))
```

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

```
identical(length(sex), nrow(pca$x))
```

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

```
pca_scores <- as.data.frame(pca$x[, 1:2])



center <- colMeans(pca_scores)
cov_mat <- cov(pca_scores)

distances <- mahalanobis(pca_scores, center, cov_mat)

# Cutoff (chi with p=0.95 and 2 df)
cutoff <- qchisq(0.95, df = 2)

# plot with condidese matrix
plot(pca$x[,1], pca$x[,2],
     pch = 19,
     col = ifelse(distances > cutoff, "red", "black"), # Outliers a vermelho
     xlab = "PC1", ylab = "PC2",
     main = "PCA Outlier Detection (95% CI)")

lines(ellipse(cov_mat, centre = center, level = 0.95), col="blue", lty=2)

text(pca$x[,1], pca$x[,2],
     labels = ifelse(distances > cutoff, rownames(pca$x), ""),
     pos = 3, cex = 0.7)
```



```
# Identify outliers
outliers <- which(distances > cutoff)
print(paste("Outliers detected:", length(outliers)))
```

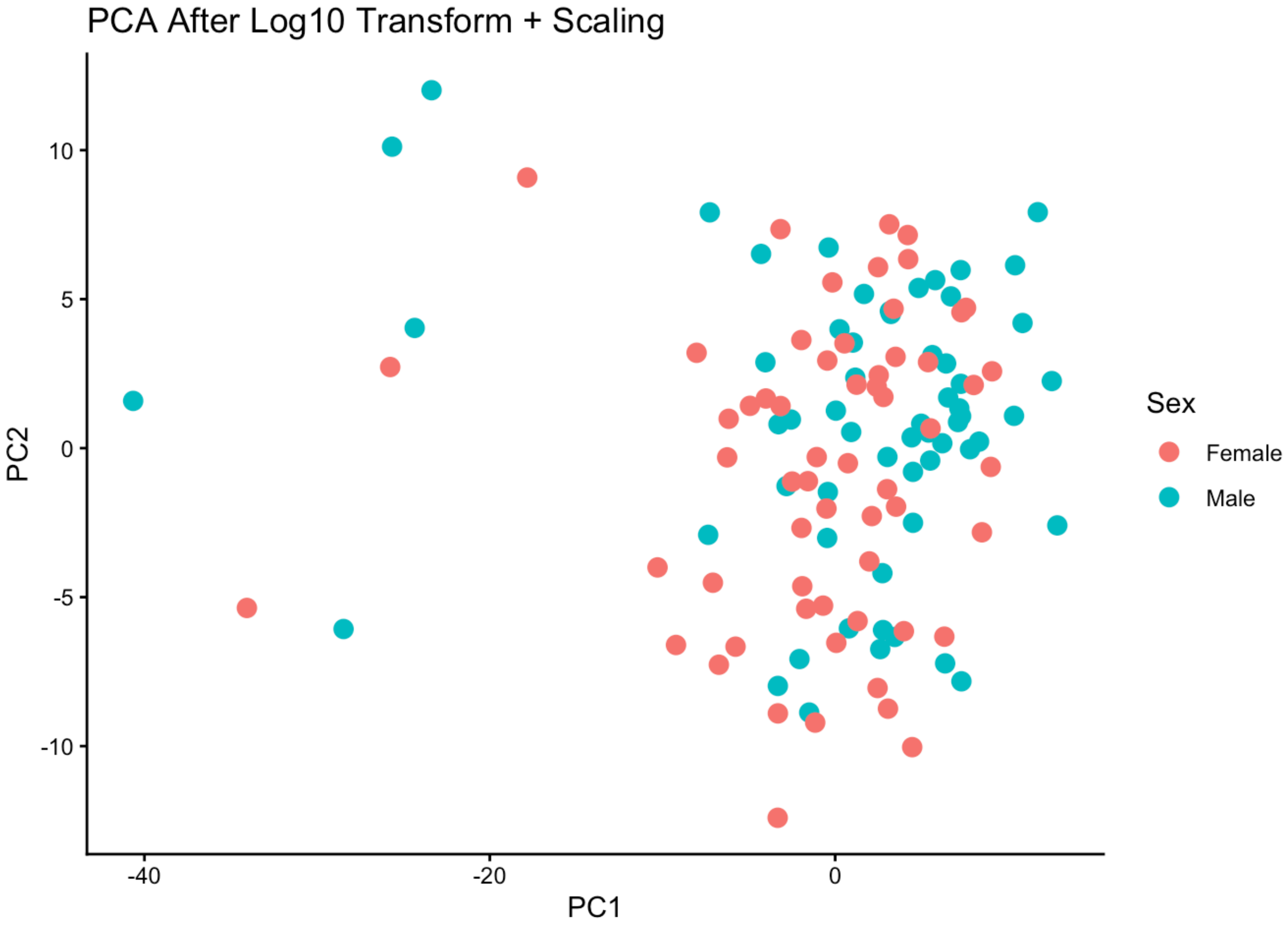
```
## [1] "Outliers detected: 9"
```

We detected 9 outliers, but they weren’t removed because:

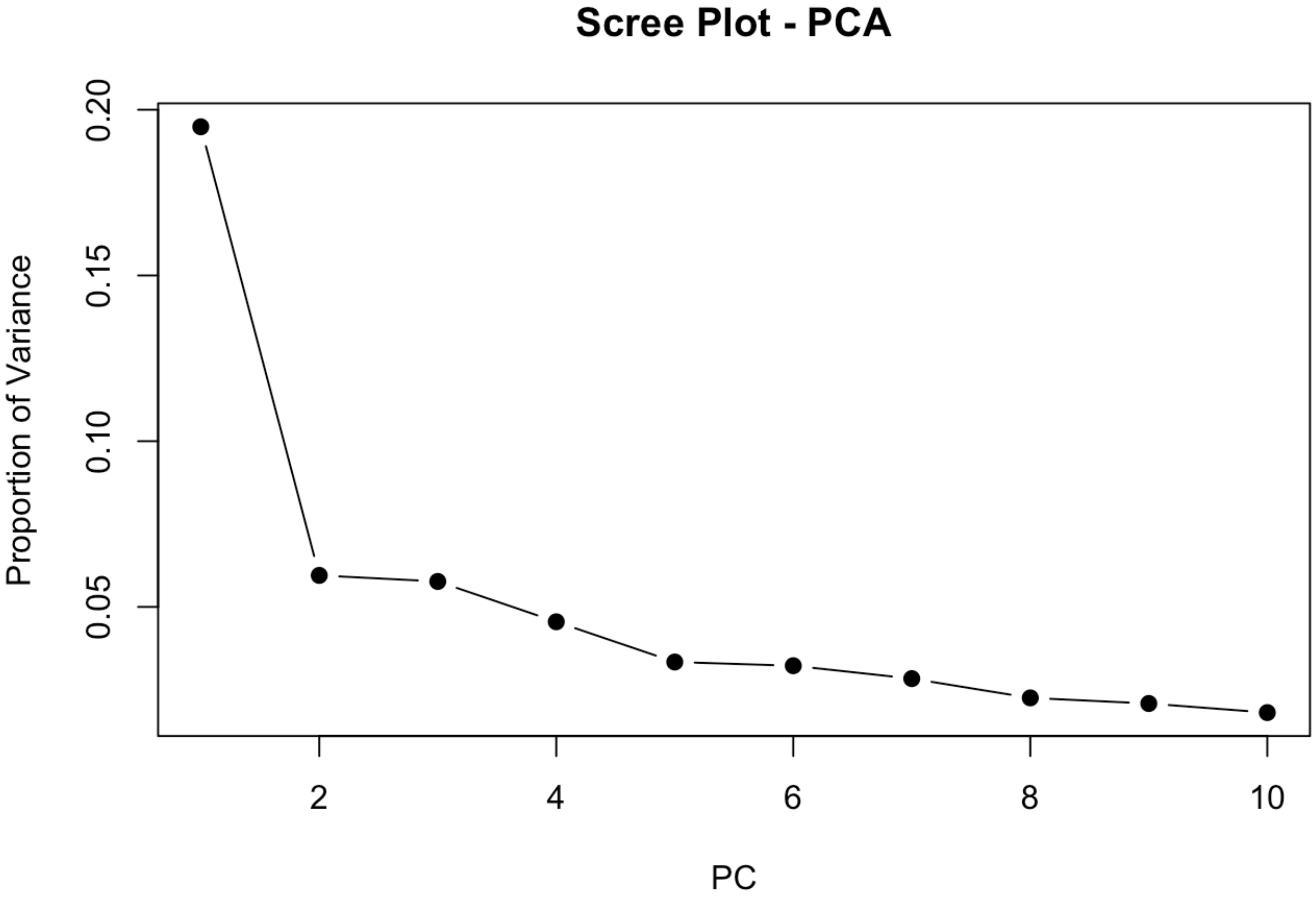
Outliers reflect **real biological variability** within the healthy population.
Removing them could introduce **bias**, especially if outliers are unevenly distributed between sexes.
Furthermore, the original study **did not exclude any samples** all **120 subjects** were analyzed in full.

PCA after log transformation

```
score_df <- data.frame(
  PC1 = pca$x[,1],
  PC2 = pca$x[,2],
  Sex = sex
)

ggplot(score_df, aes(PC1, PC2, color = Sex)) +
  geom_point(size = 3) +
  theme_classic() +
  ggtitle("PCA After Log10 Transform + Scaling")
```



```
# Scree plot (PCA Variance Explained)
variance <- pca$sdev^2 / sum(pca$sdev^2)

plot(variance[1:10], type="b", pch=19,
     xlab="PC", ylab="Proportion of Variance",
     main="Scree Plot - PCA")
```



- Score plot shows a trend of separation between Male and Female
- Some possible outliers appear (next step: detection)
- Scree Plot indicates PC1+PC2 explain ~20–30% variance → normal for metabolomics

Split Test/Training

We split the data into training and test sets (70/30) and perform univariate t-tests on the training set only. FDR correction is applied to identify metabolites significantly different between sexes.

```
set.seed(123)

train_index <- createDataPartition(sex, p = 0.7, list = FALSE)

X_train <- X_scaled[train_index, ]
X_test  <- X_scaled[-train_index, ]

X_train <- as.data.frame(X_train)
X_test  <- as.data.frame(X_test)

y_train <- sex[train_index]
y_test  <- sex[-train_index]

pvals <- apply(X_train, 2, function(x) t.test(x ~ y_train)$p.value)
pvals_fdr <- p.adjust(pvals, method = "fdr")

head(sort(pvals_fdr))
```

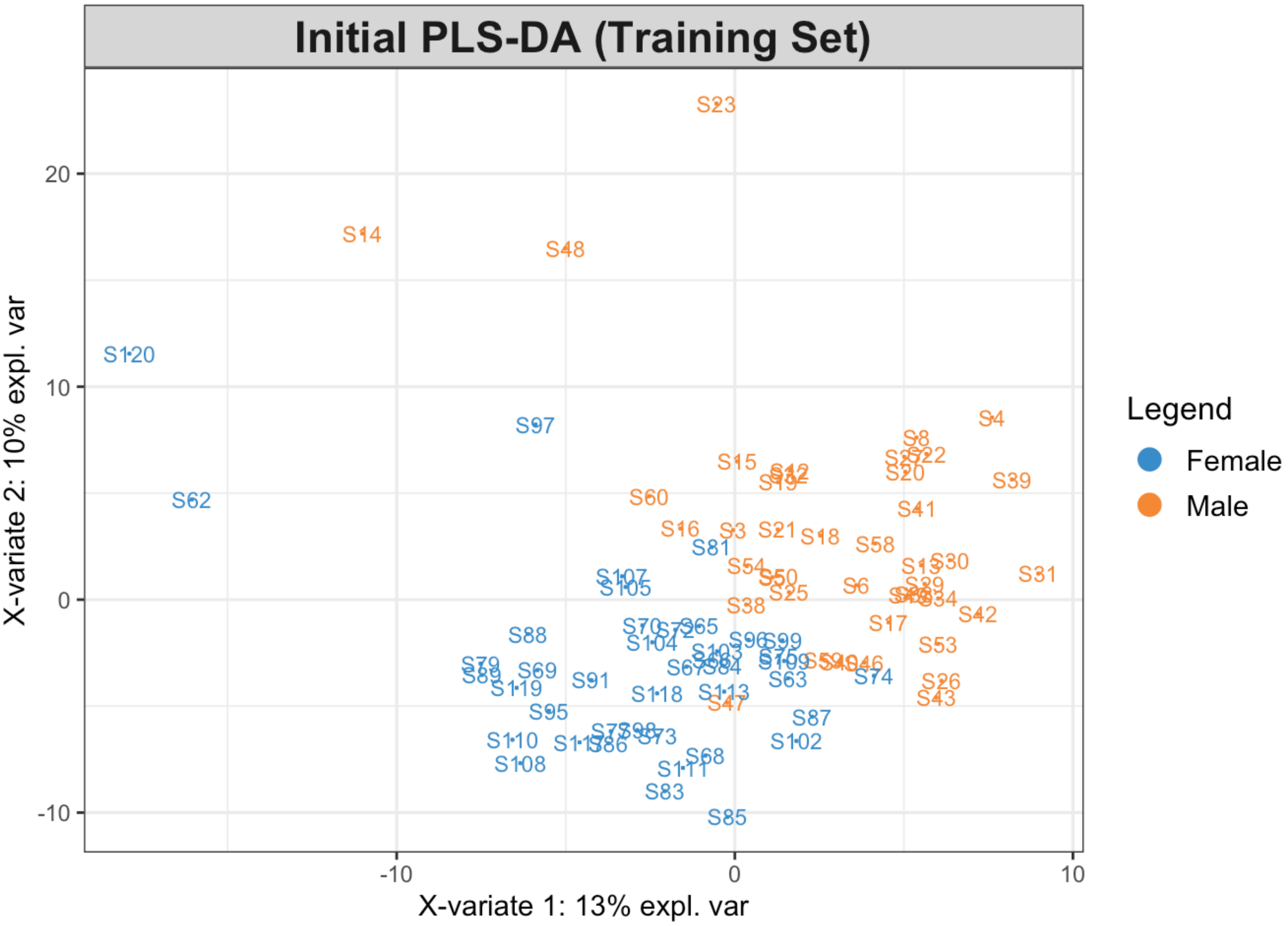
```
##         V22        V399         V10         V84         V51        V113 
## 0.001753165 0.002134547 0.044318477 0.044318477 0.044551591 0.044551591
```

Several metabolites show significant sex differences (FDR < 0.05). These will be considered as candidates for biomarker selection in the supervised modeling step.

Initials PLS-DA

An initial PLS-DA model with 10 components was trained on the training set to evaluate whether the metabolomic profiles discriminate between sexes.

```
#Initial PLS-DA model

pls_initial <- mixOmics::plsda(X_train, y_train, ncomp = 10)

mixOmics::plotIndiv(pls_initial, comp = c(1,2),
          group = y_train, legend = TRUE,
          title = "Initial PLS-DA (Training Set)")
```



The PLS-DA score plot shows a clear separation between males and females along the first two latent variables, indicating strong sex-related differences in urinary metabolomic profiles.

Internal Validation: 5-Fold CV + AUCROC

A 5-fold cross-validation was used on the training set to determine the optimal number of latent variables (LVs) for the PLS-DA model, using AUC as the performance metric.

```
set.seed(123)

folds <- createFolds(y_train, k = 5, returnTrain = TRUE)
auc_results <- data.frame(LV = integer(), AUC = numeric())

for (lv in 1:5) {
  auc_fold <- c()

  for (f in folds) {
    model <- mixOmics::plsda(X_train[f,], y_train[f], ncomp = lv)
    pred <- predict(model, X_train[-f,])$predict[,1,lv]
    roc_obj <- roc(y_train[-f], pred)
    auc_fold <- c(auc_fold, auc(roc_obj))
  }

  auc_results <- rbind(auc_results,
                       data.frame(LV = lv,
                                  AUC = mean(auc_fold)))
}
```

```
## Setting levels: control = Female, case = Male
```

```
## Setting direction: controls > cases
```

```
## Setting levels: control = Female, case = Male
```

```
## Setting direction: controls > cases
```

```
## Setting levels: control = Female, case = Male
```

```
## Setting direction: controls > cases
```

```
## Setting levels: control = Female, case = Male
```

```
## Setting direction: controls > cases
```

```
## Setting levels: control = Female, case = Male
```

```
## Setting direction: controls > cases
```

```
## Setting levels: control = Female, case = Male
## Setting direction: controls > cases
```

```
## Setting levels: control = Female, case = Male
```

```
## Setting direction: controls > cases
```

```
## Setting levels: control = Female, case = Male
```

```
## Setting direction: controls > cases
```

```
## Setting levels: control = Female, case = Male
```

```
## Setting direction: controls > cases
```

```
## Setting levels: control = Female, case = Male
```

```
## Setting direction: controls > cases
```

```
## Setting levels: control = Female, case = Male
```

```
## Setting direction: controls > cases
```

```
## Setting levels: control = Female, case = Male
```

```
## Setting direction: controls > cases
```

```
## Setting levels: control = Female, case = Male
```

```
## Setting direction: controls > cases
```

```
## Setting levels: control = Female, case = Male
```

```
## Setting direction: controls > cases
```

```
## Setting levels: control = Female, case = Male
## Setting direction: controls > cases
```

```
## Setting levels: control = Female, case = Male
```

```
## Setting direction: controls > cases
```

```
## Setting levels: control = Female, case = Male
```

```
## Setting direction: controls > cases
```

```
## Setting levels: control = Female, case = Male
```

```
## Setting direction: controls > cases
```

```
## Setting levels: control = Female, case = Male
```

```
## Setting direction: controls > cases
```

```
## Setting levels: control = Female, case = Male
```

```
## Setting direction: controls > cases
```

```
## Setting levels: control = Female, case = Male
```

```
## Setting direction: controls > cases
```

```
## Setting levels: control = Female, case = Male
```

```
## Setting direction: controls > cases
```

```
## Setting levels: control = Female, case = Male
```

```
## Setting direction: controls > cases
```

```
## Setting levels: control = Female, case = Male
## Setting direction: controls > cases
```

```
## Setting levels: control = Female, case = Male
```

```
## Setting direction: controls > cases
```

```
auc_results
```

```
##   LV       AUC
## 1  1 0.7246914
## 2  2 0.7500772
## 3  3 0.8280478
## 4  4 0.8102623
## 5  5 0.8032793
```

LV = 3 showed the highest mean AUC, and was therefore selected as the optimal number of components for the final PLS-DA model.

Univariate Analysis on Training set (Male vs Female)

Univariate Welch t-tests were applied to each metabolite in the training set, followed by FDR correction. Mean differences (Male – Female) were calculated to determine the direction of change.

```
# Compute raw p-values using Welch t-test for each metabolite
pvals <- apply(X_train, 2, function(x) t.test(x ~ y_train)$p.value)

# FDR correction (Benjamini-Hochberg)
pvals_fdr <- p.adjust(pvals, method = "fdr")


fc <- apply(X_train, 2, function(x) mean(x[y_train == "Male"]) -
                               mean(x[y_train == "Female"]))


uni_results <- data.frame(
  Metabolite = colnames(X_train),
  p_value = pvals,
  p_FDR = pvals_fdr,
  mean_diff = fc
)


uni_results <- uni_results[order(uni_results$p_FDR), ]
significant <- uni_results[uni_results$p_FDR < 0.05, ]
significant
```

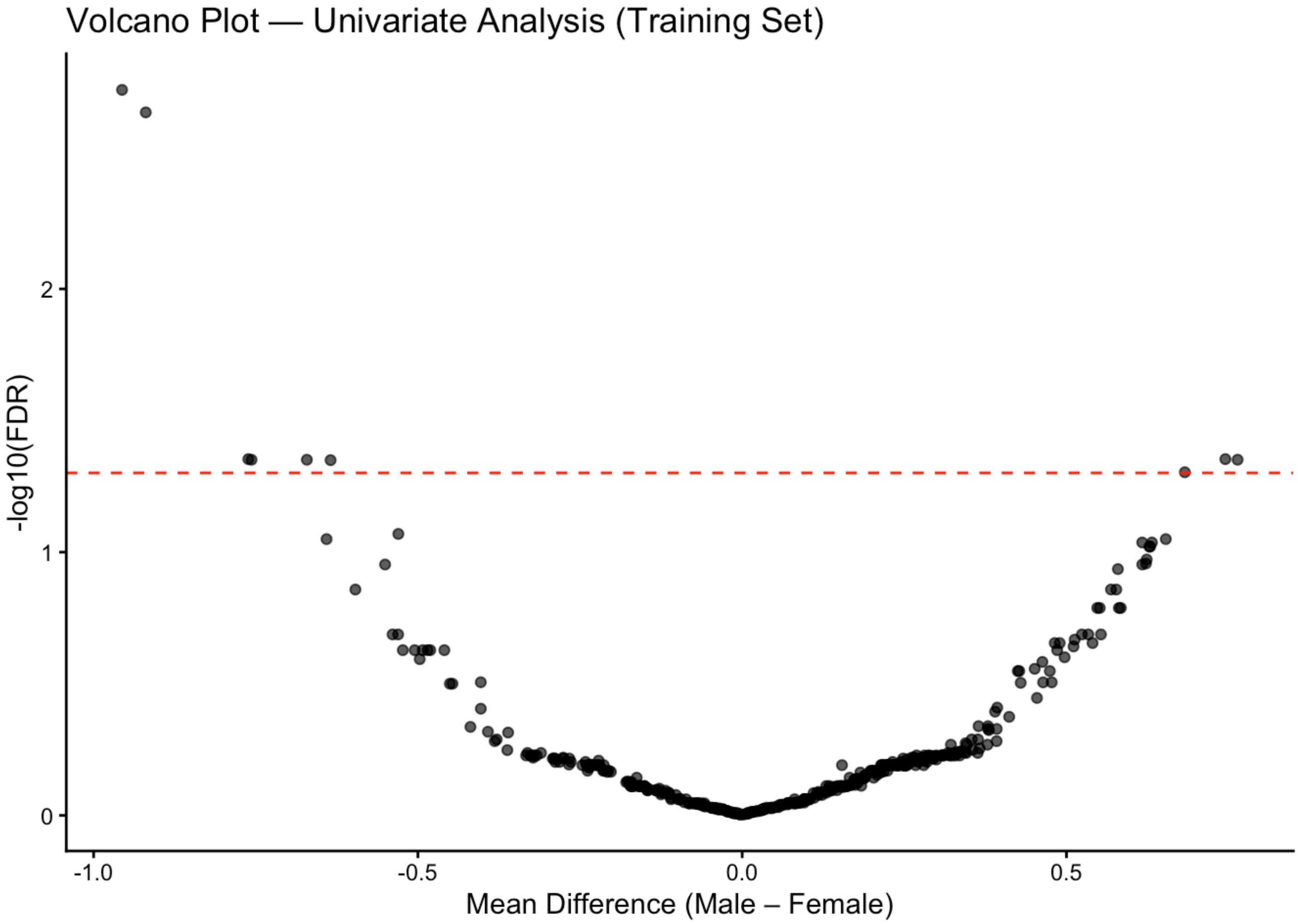
```
##      Metabolite      p_value       p_FDR  mean_diff
## V22         V22 4.234697e-06 0.001753165 -0.9561637
## V399       V399 1.031182e-05 0.002134547 -0.9196217
## V10         V10 4.281978e-04 0.044318477  0.7449845
## V84         V84 3.253387e-04 0.044318477 -0.7615165
## V51         V51 6.958870e-04 0.044551591 -0.7565657
## V113       V113 6.825835e-04 0.044551591  0.7638415
## V291       V291 7.532878e-04 0.044551591 -0.6712124
## V57         V57 8.641712e-04 0.044720859 -0.6346747
## V250       V250 1.080601e-03 0.049707666  0.6825126
```

Several metabolites showed significant sex differences (FDR < 0.05). These features represent potential biomarkers and will be further evaluated in the supervised model.

Volcano Plot - Univariate Analysis

A volcano plot was generated to visualize effect size (mean difference) against statistical significance (FDR-corrected p-values).

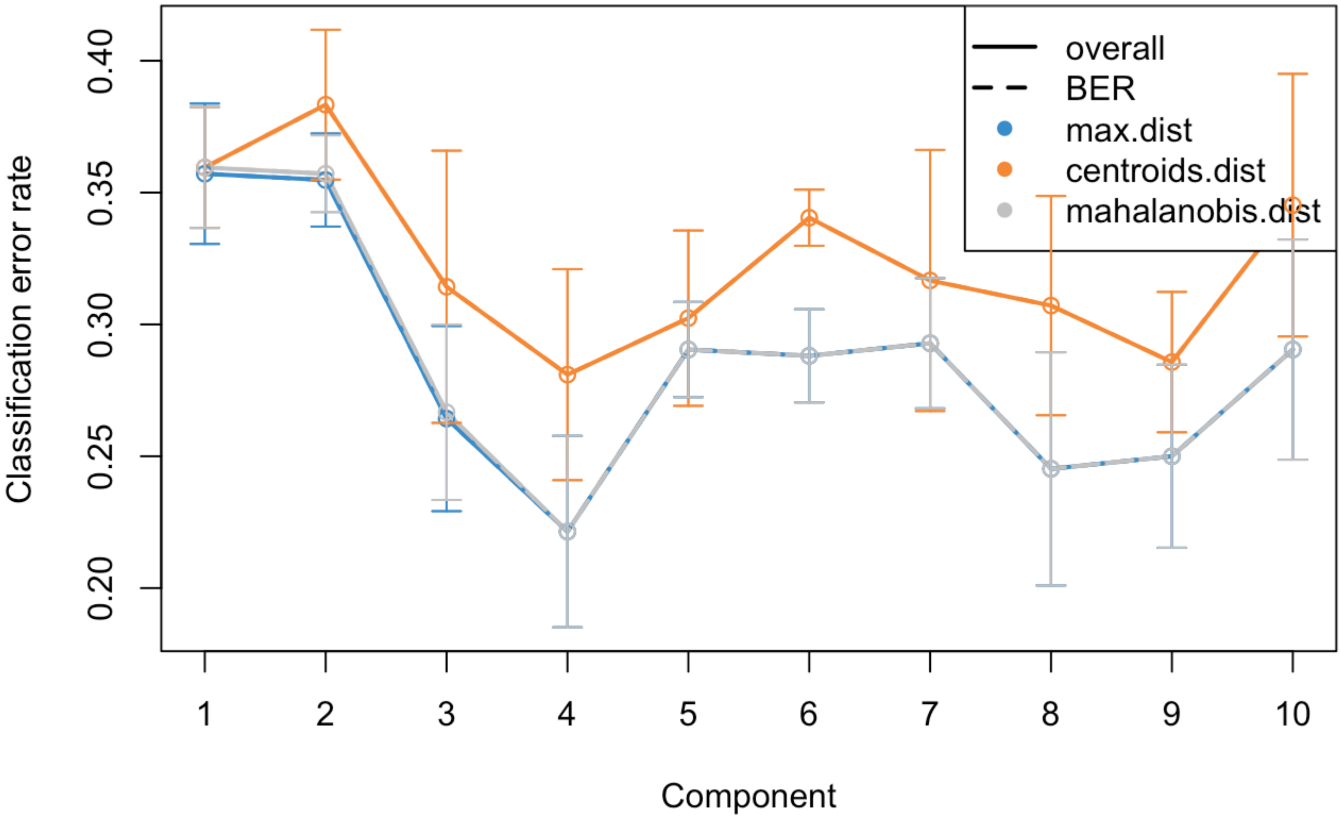
```
volcano <- uni_results
volcano$logP <- -log10(volcano$p_FDR)

ggplot(volcano, aes(x = mean_diff, y = logP)) +
  geom_point(alpha = 0.7) +
  geom_hline(yintercept = -log10(0.05), col = "red", lty = 2) +
  theme_classic() +
  xlab("Mean Difference (Male – Female)") +
  ylab("-log10(FDR)") +
  ggtitle("Volcano Plot — Univariate Analysis (Training Set)")
```



Univariate analysis performed on the training set identified 9 metabolites with significant sex-related differences after FDR correction (p_FDR < 0.05). Six metabolites (V22, V399, V84, V51, V291, V57) showed higher concentrations in females, while three metabolites (V10, V113, V250) were higher in males. The strongest discriminators were V22 and V399 (p_FDR < 0.005).

```
colnames(X_train) <- paste0("X", 1:ncol(X_train))
colnames(X_test)  <- paste0("X", 1:ncol(X_test))

y_train <- as.factor(y_train)
y_test  <- as.factor(y_test)
```

```
set.seed(30)

perf_res <- perf(pls_initial,
                 validation = "Mfold",
                 folds = 5,
                 nrepeat = 5,
                 progressBar = FALSE)

plot(perf_res, sd = TRUE)
```



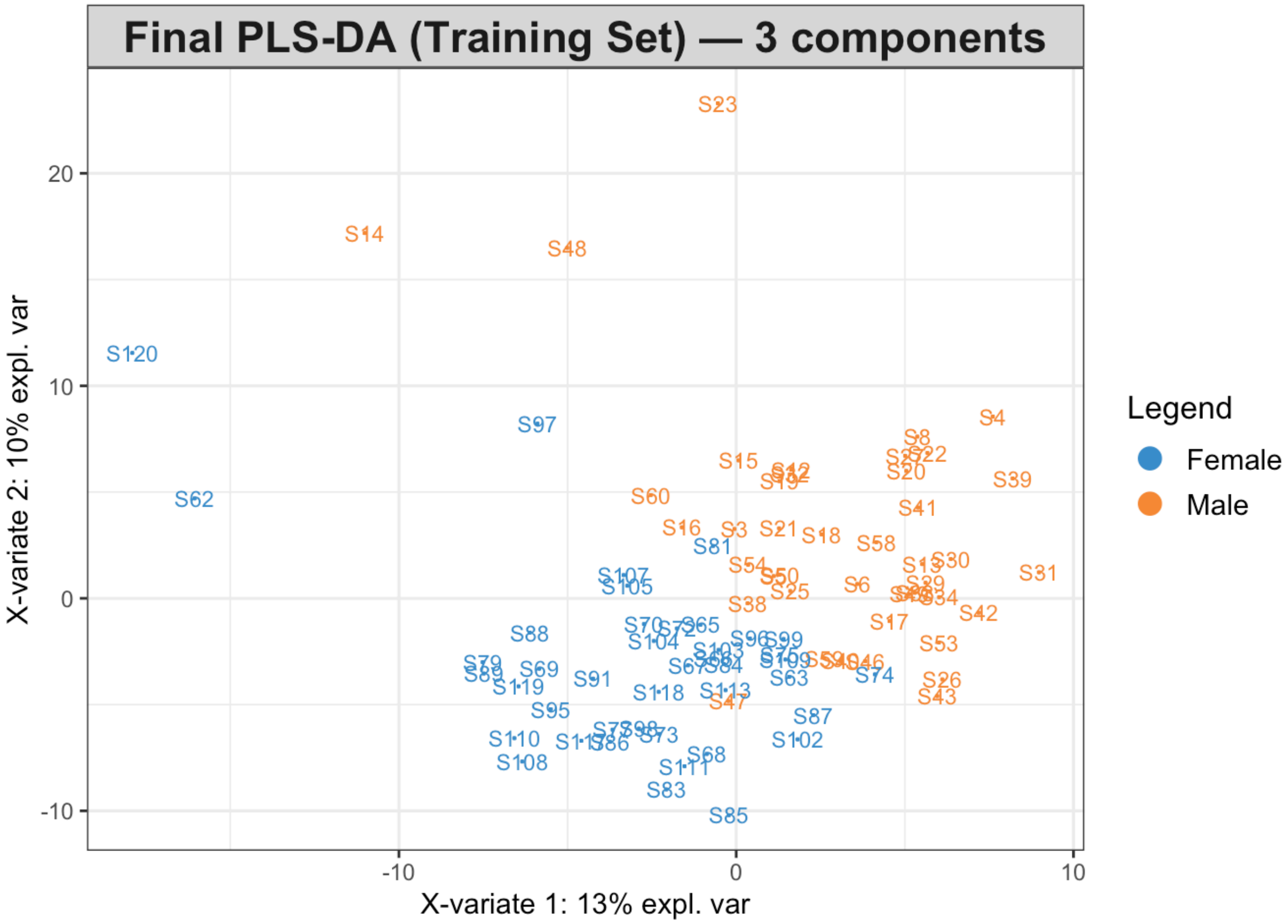
Calculate the optimal number of components, using other metrics, like BER. This shows that increasing the number of components doesn't always decrease the error.

As an additional exploratory validation, the perf() method from mixOmics was applied. Although BER suggested 4 components, this criterion is less appropriate for binary classification than AUC. Therefore, AUC-based tuning remained the primary selection method.

```
optimal_ncomp <- which.min(perf_res$error.rate$BER)
optimal_ncomp
```

```
## [1] 4
```

```
optimal_ncomp <-3
pls_final <- mixOmics::plsda(X_train, y_train, ncomp = optimal_ncomp)

mixOmics::plotIndiv(pls_final, comp = c(1,2),
          group = y_train, legend = TRUE,
          title = "Final PLS-DA (Training Set) — 3 components")
```



We selected n = 3 components because they maximised the AUC during cross-validation while avoiding unnecessary model complexity. Although BER suggested 4 components, the improvement was marginal and adding extra components risks overfitting, especially in a two-class dataset.

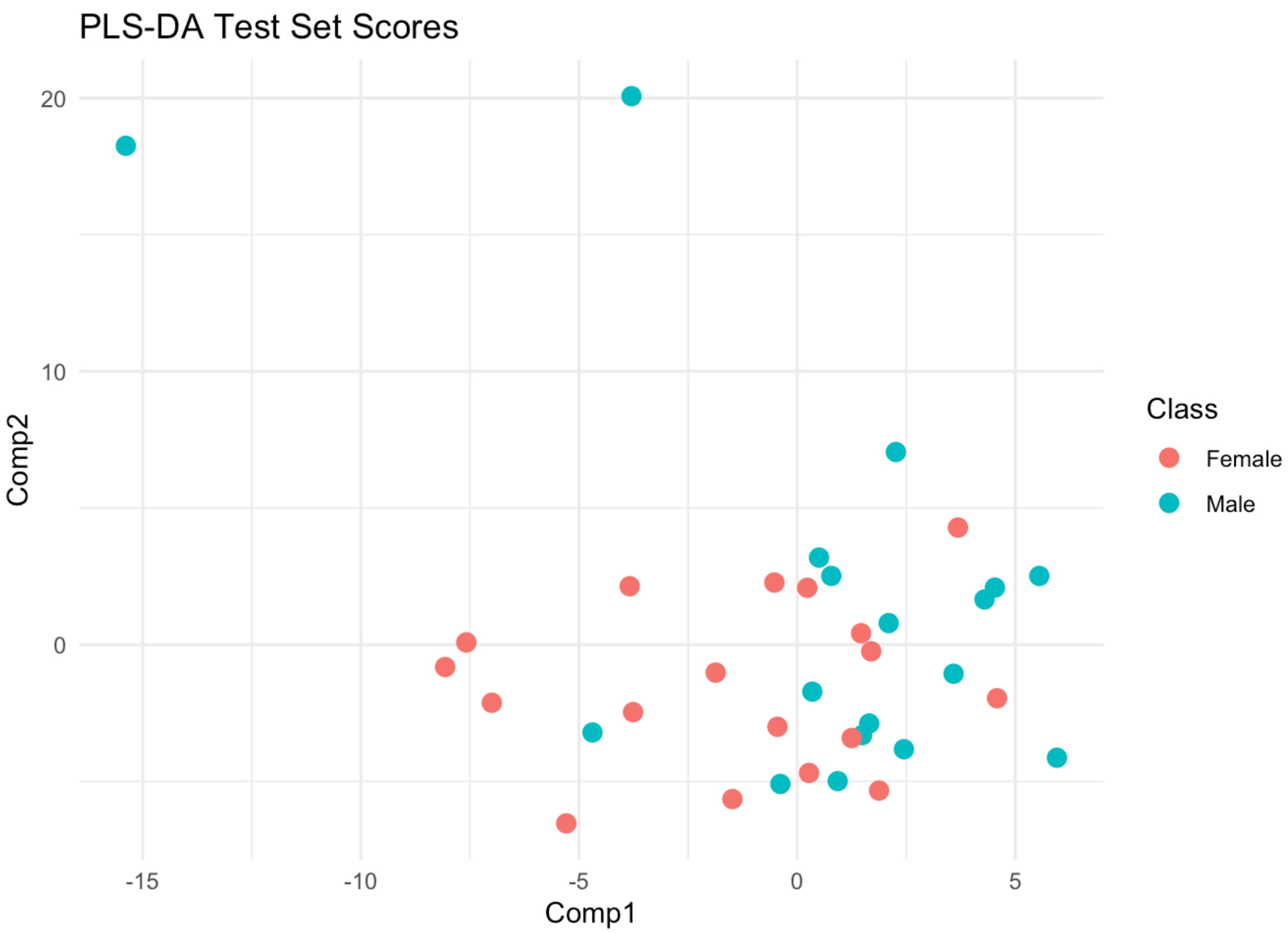
```
pred_test <- predict(pls_final, X_test)
pred_class <- pred_test$class$max.dist[, optimal_ncomp]

table(True = y_test, Predicted = pred_class)
```

```
##         Predicted
## True      Female Male
##   Female      14    4
##   Male         5   13
```

```
mean(pred_class == y_test)
```

```
## [1] 0.75
```

```
test_variates <- pred_test$variates

test_df <- data.frame(
  Comp1 = test_variates[,1],
  Comp2 = if (optimal_ncomp >= 2) test_variates[,2] else rep(0, nrow(test_variate
s)),
  Class = y_test
)

ggplot(test_df, aes(Comp1, Comp2, color = Class)) +
  geom_point(size = 3) +
  theme_minimal() +
  ggtitle("PLS-DA Test Set Scores")
```



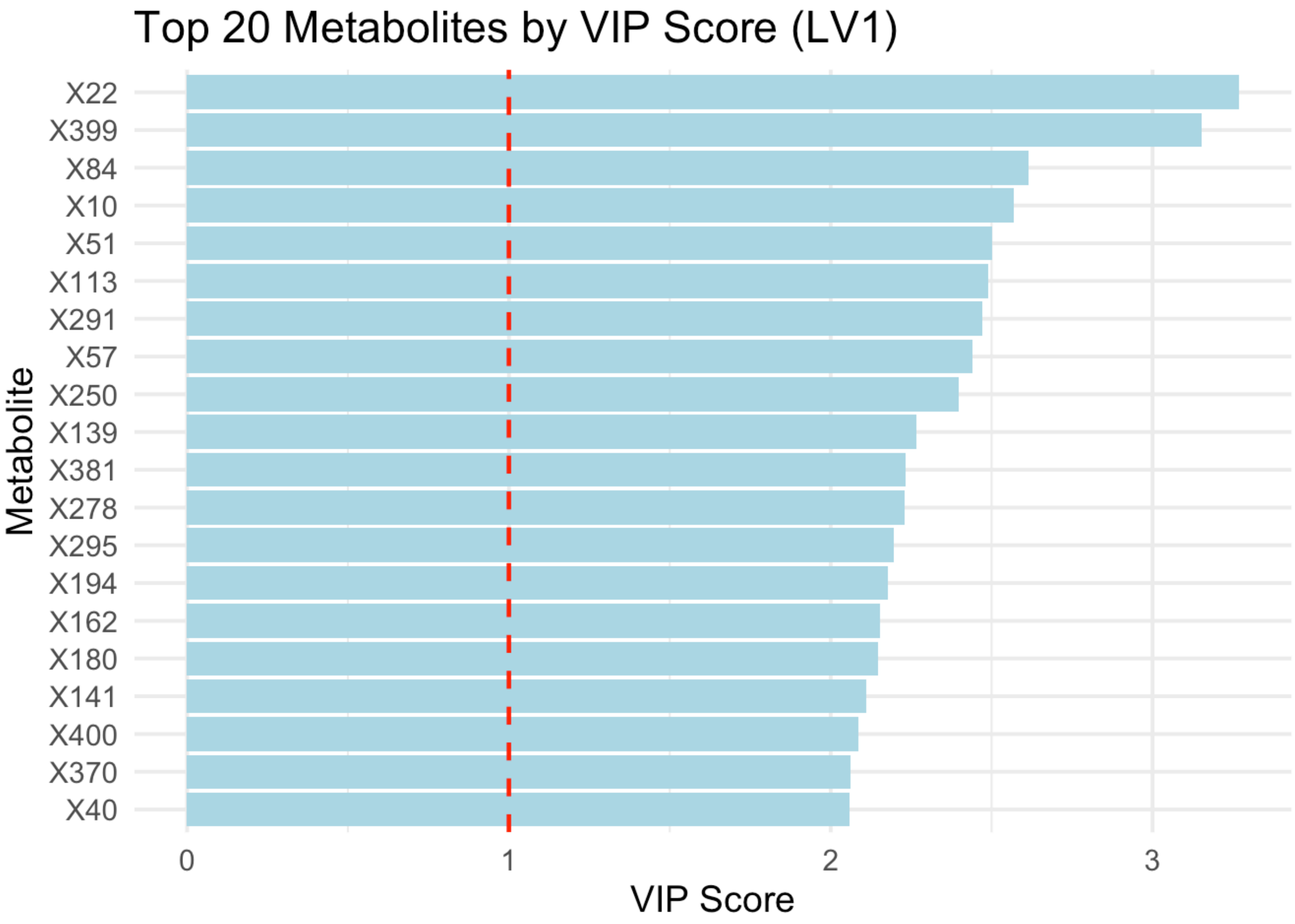
The final PLS-DA model does not show a visibly stronger separation because the main class structure was already captured by the first two components in the initial model. Tuning improves model stability and classification performance, but these gains occur mostly beyond component 2, which is why the 2D score plot appears similar.

Feature Selection: Calculate and Rank VIP Scores

Calculate and Rank VIP Scores

```
# 1. Extract the VIP scores matrix from the final PLS-DA model
# We typically focus on the scores from the first component (LV1)
vip_matrix <- vip(pls_final)
vip_scores_lv1 <- vip_matrix[, 1]

# 2. Order the metabolites from most important to least important
vip_ranked <- sort(vip_scores_lv1, decreasing = TRUE)

# 3. Convert to a data frame for plotting and analysis
vip_df <- data.frame(
  Metabolite = names(vip_ranked),
  VIP = vip_ranked
)

# 4. Visualize the top 20 most important metabolites
top_n <- 20
vip_top <- head(vip_df, top_n)

# Generate the bar plot
ggplot(vip_top, aes(x = reorder(Metabolite, VIP), y = VIP)) +
  geom_bar(stat = "identity", fill = "light blue") +
  geom_hline(yintercept = 1, linetype = "dashed", color = "red", size = 0.8) +
  coord_flip() +
  theme_minimal(base_size = 14) +
  labs(title = paste("Top", top_n, "Metabolites by VIP Score (LV1)"),
       y = "VIP Score",
       x = "Metabolite")
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## ℹ Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



Implement Recursive Feature Elimination (RFE) using the k-NN Classifier

```
set.seed(123)

# Define subset sizes for RFE
subset_sizes <- c(2, 5, 10, 15, 20, 25, 30, 40, 50)

# RFE control with default caret functions
rfe_control <- rfeControl(
  functions = caretFuncs,   # default ranking
  method = "repeatedcv",
  number = 5,
  repeats = 5,
  verbose = TRUE,          # suppress intermediate output
  returnResamp = "all"
)

# Execute RFE
rfe_results <- rfe(
  x = X_train,
  y = y_train,
  sizes = subset_sizes,
  rfeControl = rfe_control,
  method = "knn",
  tuneLength = 5,
  metric = "Accuracy",
  preProcess = c("center","scale")
)
```

```
## +(rfe) fit Fold1.Rep1 size: 414
## -(rfe) fit Fold1.Rep1 size: 414
## +(rfe) imp Fold1.Rep1
## -(rfe) imp Fold1.Rep1
## +(rfe) fit Fold1.Rep1 size:  50
## -(rfe) fit Fold1.Rep1 size:  50
## +(rfe) fit Fold1.Rep1 size:  40
## -(rfe) fit Fold1.Rep1 size:  40
## +(rfe) fit Fold1.Rep1 size:  30
## -(rfe) fit Fold1.Rep1 size:  30
## +(rfe) fit Fold1.Rep1 size:  25
## -(rfe) fit Fold1.Rep1 size:  25
## +(rfe) fit Fold1.Rep1 size:  20
## -(rfe) fit Fold1.Rep1 size:  20
## +(rfe) fit Fold1.Rep1 size:  15
## -(rfe) fit Fold1.Rep1 size:  15
## +(rfe) fit Fold1.Rep1 size:  10
## -(rfe) fit Fold1.Rep1 size:  10
## +(rfe) fit Fold1.Rep1 size:   5
## -(rfe) fit Fold1.Rep1 size:   5
## +(rfe) fit Fold1.Rep1 size:   2
## -(rfe) fit Fold1.Rep1 size:   2
## +(rfe) fit Fold2.Rep1 size: 414
## -(rfe) fit Fold2.Rep1 size: 414
## +(rfe) imp Fold2.Rep1
## -(rfe) imp Fold2.Rep1
## +(rfe) fit Fold2.Rep1 size:  50
## -(rfe) fit Fold2.Rep1 size:  50
## +(rfe) fit Fold2.Rep1 size:  40
## -(rfe) fit Fold2.Rep1 size:  40
## +(rfe) fit Fold2.Rep1 size:  30
## -(rfe) fit Fold2.Rep1 size:  30
## +(rfe) fit Fold2.Rep1 size:  25
## -(rfe) fit Fold2.Rep1 size:  25
## +(rfe) fit Fold2.Rep1 size:  20
## -(rfe) fit Fold2.Rep1 size:  20
## +(rfe) fit Fold2.Rep1 size:  15
## -(rfe) fit Fold2.Rep1 size:  15
## +(rfe) fit Fold2.Rep1 size:  10
## -(rfe) fit Fold2.Rep1 size:  10
## +(rfe) fit Fold2.Rep1 size:   5
## -(rfe) fit Fold2.Rep1 size:   5
## +(rfe) fit Fold2.Rep1 size:   2
## -(rfe) fit Fold2.Rep1 size:   2
## +(rfe) fit Fold3.Rep1 size: 414
## -(rfe) fit Fold3.Rep1 size: 414
## +(rfe) imp Fold3.Rep1
## -(rfe) imp Fold3.Rep1
## +(rfe) fit Fold3.Rep1 size:  50
## -(rfe) fit Fold3.Rep1 size:  50
## +(rfe) fit Fold3.Rep1 size:  40
## -(rfe) fit Fold3.Rep1 size:  40
## +(rfe) fit Fold3.Rep1 size:  30
## -(rfe) fit Fold3.Rep1 size:  30
## +(rfe) fit Fold3.Rep1 size:  25
## -(rfe) fit Fold3.Rep1 size:  25
## +(rfe) fit Fold3.Rep1 size:  20
## -(rfe) fit Fold3.Rep1 size:  20
## +(rfe) fit Fold3.Rep1 size:  15
## -(rfe) fit Fold3.Rep1 size:  15
## +(rfe) fit Fold3.Rep1 size:  10
## -(rfe) fit Fold3.Rep1 size:  10
## +(rfe) fit Fold3.Rep1 size:   5
## -(rfe) fit Fold3.Rep1 size:   5
## +(rfe) fit Fold3.Rep1 size:   2
## -(rfe) fit Fold3.Rep1 size:   2
## +(rfe) fit Fold4.Rep1 size: 414
## -(rfe) fit Fold4.Rep1 size: 414
## +(rfe) imp Fold4.Rep1
## -(rfe) imp Fold4.Rep1
## +(rfe) fit Fold4.Rep1 size:  50
## -(rfe) fit Fold4.Rep1 size:  50
## +(rfe) fit Fold4.Rep1 size:  40
## -(rfe) fit Fold4.Rep1 size:  40
## +(rfe) fit Fold4.Rep1 size:  30
## -(rfe) fit Fold4.Rep1 size:  30
## +(rfe) fit Fold4.Rep1 size:  25
## -(rfe) fit Fold4.Rep1 size:  25
## +(rfe) fit Fold4.Rep1 size:  20
## -(rfe) fit Fold4.Rep1 size:  20
## +(rfe) fit Fold4.Rep1 size:  15
## -(rfe) fit Fold4.Rep1 size:  15
## +(rfe) fit Fold4.Rep1 size:  10
## -(rfe) fit Fold4.Rep1 size:  10
## +(rfe) fit Fold4.Rep1 size:   5
## -(rfe) fit Fold4.Rep1 size:   5
## +(rfe) fit Fold4.Rep1 size:   2
## -(rfe) fit Fold4.Rep1 size:   2
## +(rfe) fit Fold5.Rep1 size: 414
## -(rfe) fit Fold5.Rep1 size: 414
## +(rfe) imp Fold5.Rep1
## -(rfe) imp Fold5.Rep1
## +(rfe) fit Fold5.Rep1 size:  50
## -(rfe) fit Fold5.Rep1 size:  50
## +(rfe) fit Fold5.Rep1 size:  40
## -(rfe) fit Fold5.Rep1 size:  40
## +(rfe) fit Fold5.Rep1 size:  30
## -(rfe) fit Fold5.Rep1 size:  30
## +(rfe) fit Fold5.Rep1 size:  25
## -(rfe) fit Fold5.Rep1 size:  25
## +(rfe) fit Fold5.Rep1 size:  20
## -(rfe) fit Fold5.Rep1 size:  20
## +(rfe) fit Fold5.Rep1 size:  15
## -(rfe) fit Fold5.Rep1 size:  15
## +(rfe) fit Fold5.Rep1 size:  10
## -(rfe) fit Fold5.Rep1 size:  10
## +(rfe) fit Fold5.Rep1 size:   5
## -(rfe) fit Fold5.Rep1 size:   5
## +(rfe) fit Fold5.Rep1 size:   2
## -(rfe) fit Fold5.Rep1 size:   2
## +(rfe) fit Fold1.Rep2 size: 414
## -(rfe) fit Fold1.Rep2 size: 414
## +(rfe) imp Fold1.Rep2
## -(rfe) imp Fold1.Rep2
## +(rfe) fit Fold1.Rep2 size:  50
## -(rfe) fit Fold1.Rep2 size:  50
## +(rfe) fit Fold1.Rep2 size:  40
## -(rfe) fit Fold1.Rep2 size:  40
## +(rfe) fit Fold1.Rep2 size:  30
## -(rfe) fit Fold1.Rep2 size:  30
## +(rfe) fit Fold1.Rep2 size:  25
## -(rfe) fit Fold1.Rep2 size:  25
## +(rfe) fit Fold1.Rep2 size:  20
## -(rfe) fit Fold1.Rep2 size:  20
## +(rfe) fit Fold1.Rep2 size:  15
## -(rfe) fit Fold1.Rep2 size:  15
## +(rfe) fit Fold1.Rep2 size:  10
## -(rfe) fit Fold1.Rep2 size:  10
## +(rfe) fit Fold1.Rep2 size:   5
## -(rfe) fit Fold1.Rep2 size:   5
## +(rfe) fit Fold1.Rep2 size:   2
## -(rfe) fit Fold1.Rep2 size:   2
## +(rfe) fit Fold2.Rep2 size: 414
## -(rfe) fit Fold2.Rep2 size: 414
## +(rfe) imp Fold2.Rep2
## -(rfe) imp Fold2.Rep2
## +(rfe) fit Fold2.Rep2 size:  50
## -(rfe) fit Fold2.Rep2 size:  50
## +(rfe) fit Fold2.Rep2 size:  40
## -(rfe) fit Fold2.Rep2 size:  40
## +(rfe) fit Fold2.Rep2 size:  30
## -(rfe) fit Fold2.Rep2 size:  30
## +(rfe) fit Fold2.Rep2 size:  25
## -(rfe) fit Fold2.Rep2 size:  25
## +(rfe) fit Fold2.Rep2 size:  20
## -(rfe) fit Fold2.Rep2 size:  20
## +(rfe) fit Fold2.Rep2 size:  15
## -(rfe) fit Fold2.Rep2 size:  15
## +(rfe) fit Fold2.Rep2 size:  10
## -(rfe) fit Fold2.Rep2 size:  10
## +(rfe) fit Fold2.Rep2 size:   5
## -(rfe) fit Fold2.Rep2 size:   5
## +(rfe) fit Fold2.Rep2 size:   2
## -(rfe) fit Fold2.Rep2 size:   2
## +(rfe) fit Fold3.Rep2 size: 414
## -(rfe) fit Fold3.Rep2 size: 414
## +(rfe) imp Fold3.Rep2
## -(rfe) imp Fold3.Rep2
## +(rfe) fit Fold3.Rep2 size:  50
## -(rfe) fit Fold3.Rep2 size:  50
## +(rfe) fit Fold3.Rep2 size:  40
## -(rfe) fit Fold3.Rep2 size:  40
## +(rfe) fit Fold3.Rep2 size:  30
## -(rfe) fit Fold3.Rep2 size:  30
## +(rfe) fit Fold3.Rep2 size:  25
## -(rfe) fit Fold3.Rep2 size:  25
## +(rfe) fit Fold3.Rep2 size:  20
## -(rfe) fit Fold3.Rep2 size:  20
## +(rfe) fit Fold3.Rep2 size:  15
## -(rfe) fit Fold3.Rep2 size:  15
## +(rfe) fit Fold3.Rep2 size:  10
## -(rfe) fit Fold3.Rep2 size:  10
## +(rfe) fit Fold3.Rep2 size:   5
## -(rfe) fit Fold3.Rep2 size:   5
## +(rfe) fit Fold3.Rep2 size:   2
## -(rfe) fit Fold3.Rep2 size:   2
## +(rfe) fit Fold4.Rep2 size: 414
## -(rfe) fit Fold4.Rep2 size: 414
## +(rfe) imp Fold4.Rep2
## -(rfe) imp Fold4.Rep2
## +(rfe) fit Fold4.Rep2 size:  50
## -(rfe) fit Fold4.Rep2 size:  50
## +(rfe) fit Fold4.Rep2 size:  40
## -(rfe) fit Fold4.Rep2 size:  40
## +(rfe) fit Fold4.Rep2 size:  30
## -(rfe) fit Fold4.Rep2 size:  30
## +(rfe) fit Fold4.Rep2 size:  25
## -(rfe) fit Fold4.Rep2 size:  25
## +(rfe) fit Fold4.Rep2 size:  20
## -(rfe) fit Fold4.Rep2 size:  20
## +(rfe) fit Fold4.Rep2 size:  15
## -(rfe) fit Fold4.Rep2 size:  15
## +(rfe) fit Fold4.Rep2 size:  10
## -(rfe) fit Fold4.Rep2 size:  10
## +(rfe) fit Fold4.Rep2 size:   5
## -(rfe) fit Fold4.Rep2 size:   5
## +(rfe) fit Fold4.Rep2 size:   2
## -(rfe) fit Fold4.Rep2 size:   2
## +(rfe) fit Fold5.Rep2 size: 414
## -(rfe) fit Fold5.Rep2 size: 414
## +(rfe) imp Fold5.Rep2
## -(rfe) imp Fold5.Rep2
## +(rfe) fit Fold5.Rep2 size:  50
## -(rfe) fit Fold5.Rep2 size:  50
## +(rfe) fit Fold5.Rep2 size:  40
## -(rfe) fit Fold5.Rep2 size:  40
## +(rfe) fit Fold5.Rep2 size:  30
## -(rfe) fit Fold5.Rep2 size:  30
## +(rfe) fit Fold5.Rep2 size:  25
## -(rfe) fit Fold5.Rep2 size:  25
## +(rfe) fit Fold5.Rep2 size:  20
## -(rfe) fit Fold5.Rep2 size:  20
## +(rfe) fit Fold5.Rep2 size:  15
## -(rfe) fit Fold5.Rep2 size:  15
## +(rfe) fit Fold5.Rep2 size:  10
## -(rfe) fit Fold5.Rep2 size:  10
## +(rfe) fit Fold5.Rep2 size:   5
## -(rfe) fit Fold5.Rep2 size:   5
## +(rfe) fit Fold5.Rep2 size:   2
## -(rfe) fit Fold5.Rep2 size:   2
## +(rfe) fit Fold1.Rep3 size: 414
## -(rfe) fit Fold1.Rep3 size: 414
## +(rfe) imp Fold1.Rep3
## -(rfe) imp Fold1.Rep3
## +(rfe) fit Fold1.Rep3 size:  50
## -(rfe) fit Fold1.Rep3 size:  50
## +(rfe) fit Fold1.Rep3 size:  40
## -(rfe) fit Fold1.Rep3 size:  40
## +(rfe) fit Fold1.Rep3 size:  30
## -(rfe) fit Fold1.Rep3 size:  30
## +(rfe) fit Fold1.Rep3 size:  25
## -(rfe) fit Fold1.Rep3 size:  25
## +(rfe) fit Fold1.Rep3 size:  20
## -(rfe) fit Fold1.Rep3 size:  20
## +(rfe) fit Fold1.Rep3 size:  15
## -(rfe) fit Fold1.Rep3 size:  15
## +(rfe) fit Fold1.Rep3 size:  10
## -(rfe) fit Fold1.Rep3 size:  10
## +(rfe) fit Fold1.Rep3 size:   5
## -(rfe) fit Fold1.Rep3 size:   5
## +(rfe) fit Fold1.Rep3 size:   2
## -(rfe) fit Fold1.Rep3 size:   2
## +(rfe) fit Fold2.Rep3 size: 414
## -(rfe) fit Fold2.Rep3 size: 414
## +(rfe) imp Fold2.Rep3
## -(rfe) imp Fold2.Rep3
## +(rfe) fit Fold2.Rep3 size:  50
## -(rfe) fit Fold2.Rep3 size:  50
## +(rfe) fit Fold2.Rep3 size:  40
## -(rfe) fit Fold2.Rep3 size:  40
## +(rfe) fit Fold2.Rep3 size:  30
## -(rfe) fit Fold2.Rep3 size:  30
## +(rfe) fit Fold2.Rep3 size:  25
## -(rfe) fit Fold2.Rep3 size:  25
## +(rfe) fit Fold2.Rep3 size:  20
## -(rfe) fit Fold2.Rep3 size:  20
## +(rfe) fit Fold2.Rep3 size:  15
## -(rfe) fit Fold2.Rep3 size:  15
## +(rfe) fit Fold2.Rep3 size:  10
## -(rfe) fit Fold2.Rep3 size:  10
## +(rfe) fit Fold2.Rep3 size:   5
## -(rfe) fit Fold2.Rep3 size:   5
## +(rfe) fit Fold2.Rep3 size:   2
## -(rfe) fit Fold2.Rep3 size:   2
## +(rfe) fit Fold3.Rep3 size: 414
## -(rfe) fit Fold3.Rep3 size: 414
## +(rfe) imp Fold3.Rep3
## -(rfe) imp Fold3.Rep3
## +(rfe) fit Fold3.Rep3 size:  50
## -(rfe) fit Fold3.Rep3 size:  50
## +(rfe) fit Fold3.Rep3 size:  40
## -(rfe) fit Fold3.Rep3 size:  40
## +(rfe) fit Fold3.Rep3 size:  30
## -(rfe) fit Fold3.Rep3 size:  30
## +(rfe) fit Fold3.Rep3 size:  25
## -(rfe) fit Fold3.Rep3 size:  25
## +(rfe) fit Fold3.Rep3 size:  20
## -(rfe) fit Fold3.Rep3 size:  20
## +(rfe) fit Fold3.Rep3 size:  15
## -(rfe) fit Fold3.Rep3 size:  15
## +(rfe) fit Fold3.Rep3 size:  10
## -(rfe) fit Fold3.Rep3 size:  10
## +(rfe) fit Fold3.Rep3 size:   5
## -(rfe) fit Fold3.Rep3 size:   5
## +(rfe) fit Fold3.Rep3 size:   2
## -(rfe) fit Fold3.Rep3 size:   2
## +(rfe) fit Fold4.Rep3 size: 414
## -(rfe) fit Fold4.Rep3 size: 414
## +(rfe) imp Fold4.Rep3
## -(rfe) imp Fold4.Rep3
## +(rfe) fit Fold4.Rep3 size:  50
## -(rfe) fit Fold4.Rep3 size:  50
## +(rfe) fit Fold4.Rep3 size:  40
## -(rfe) fit Fold4.Rep3 size:  40
## +(rfe) fit Fold4.Rep3 size:  30
## -(rfe) fit Fold4.Rep3 size:  30
## +(rfe) fit Fold4.Rep3 size:  25
## -(rfe) fit Fold4.Rep3 size:  25
## +(rfe) fit Fold4.Rep3 size:  20
## -(rfe) fit Fold4.Rep3 size:  20
## +(rfe) fit Fold4.Rep3 size:  15
## -(rfe) fit Fold4.Rep3 size:  15
## +(rfe) fit Fold4.Rep3 size:  10
## -(rfe) fit Fold4.Rep3 size:  10
## +(rfe) fit Fold4.Rep3 size:   5
## -(rfe) fit Fold4.Rep3 size:   5
## +(rfe) fit Fold4.Rep3 size:   2
## -(rfe) fit Fold4.Rep3 size:   2
## +(rfe) fit Fold5.Rep3 size: 414
## -(rfe) fit Fold5.Rep3 size: 414
## +(rfe) imp Fold5.Rep3
## -(rfe) imp Fold5.Rep3
## +(rfe) fit Fold5.Rep3 size:  50
## -(rfe) fit Fold5.Rep3 size:  50
## +(rfe) fit Fold5.Rep3 size:  40
## -(rfe) fit Fold5.Rep3 size:  40
## +(rfe) fit Fold5.Rep3 size:  30
## -(rfe) fit Fold5.Rep3 size:  30
## +(rfe) fit Fold5.Rep3 size:  25
## -(rfe) fit Fold5.Rep3 size:  25
## +(rfe) fit Fold5.Rep3 size:  20
## -(rfe) fit Fold5.Rep3 size:  20
## +(rfe) fit Fold5.Rep3 size:  15
## -(rfe) fit Fold5.Rep3 size:  15
## +(rfe) fit Fold5.Rep3 size:  10
## -(rfe) fit Fold5.Rep3 size:  10
## +(rfe) fit Fold5.Rep3 size:   5
## -(rfe) fit Fold5.Rep3 size:   5
## +(rfe) fit Fold5.Rep3 size:   2
## -(rfe) fit Fold5.Rep3 size:   2
## +(rfe) fit Fold1.Rep4 size: 414
## -(rfe) fit Fold1.Rep4 size: 414
## +(rfe) imp Fold1.Rep4
## -(rfe) imp Fold1.Rep4
## +(rfe) fit Fold1.Rep4 size:  50
## -(rfe) fit Fold1.Rep4 size:  50
## +(rfe) fit Fold1.Rep4 size:  40
## -(rfe) fit Fold1.Rep4 size:  40
## +(rfe) fit Fold1.Rep4 size:  30
## -(rfe) fit Fold1.Rep4 size:  30
## +(rfe) fit Fold1.Rep4 size:  25
## -(rfe) fit Fold1.Rep4 size:  25
## +(rfe) fit Fold1.Rep4 size:  20
## -(rfe) fit Fold1.Rep4 size:  20
## +(rfe) fit Fold1.Rep4 size:  15
## -(rfe) fit Fold1.Rep4 size:  15
## +(rfe) fit Fold1.Rep4 size:  10
## -(rfe) fit Fold1.Rep4 size:  10
## +(rfe) fit Fold1.Rep4 size:   5
## -(rfe) fit Fold1.Rep4 size:   5
## +(rfe) fit Fold1.Rep4 size:   2
## -(rfe) fit Fold1.Rep4 size:   2
## +(rfe) fit Fold2.Rep4 size: 414
## -(rfe) fit Fold2.Rep4 size: 414
## +(rfe) imp Fold2.Rep4
## -(rfe) imp Fold2.Rep4
## +(rfe) fit Fold2.Rep4 size:  50
## -(rfe) fit Fold2.Rep4 size:  50
## +(rfe) fit Fold2.Rep4 size:  40
## -(rfe) fit Fold2.Rep4 size:  40
## +(rfe) fit Fold2.Rep4 size:  30
## -(rfe) fit Fold2.Rep4 size:  30
## +(rfe) fit Fold2.Rep4 size:  25
## -(rfe) fit Fold2.Rep4 size:  25
## +(rfe) fit Fold2.Rep4 size:  20
## -(rfe) fit Fold2.Rep4 size:  20
## +(rfe) fit Fold2.Rep4 size:  15
## -(rfe) fit Fold2.Rep4 size:  15
## +(rfe) fit Fold2.Rep4 size:  10
## -(rfe) fit Fold2.Rep4 size:  10
## +(rfe) fit Fold2.Rep4 size:   5
## -(rfe) fit Fold2.Rep4 size:   5
## +(rfe) fit Fold2.Rep4 size:   2
## -(rfe) fit Fold2.Rep4 size:   2
## +(rfe) fit Fold3.Rep4 size: 414
## -(rfe) fit Fold3.Rep4 size: 414
## +(rfe) imp Fold3.Rep4
## -(rfe) imp Fold3.Rep4
## +(rfe) fit Fold3.Rep4 size:  50
## -(rfe) fit Fold3.Rep4 size:  50
## +(rfe) fit Fold3.Rep4 size:  40
## -(rfe) fit Fold3.Rep4 size:  40
## +(rfe) fit Fold3.Rep4 size:  30
## -(rfe) fit Fold3.Rep4 size:  30
## +(rfe) fit Fold3.Rep4 size:  25
## -(rfe) fit Fold3.Rep4 size:  25
## +(rfe) fit Fold3.Rep4 size:  20
## -(rfe) fit Fold3.Rep4 size:  20
## +(rfe) fit Fold3.Rep4 size:  15
## -(rfe) fit Fold3.Rep4 size:  15
## +(rfe) fit Fold3.Rep4 size:  10
## -(rfe) fit Fold3.Rep4 size:  10
## +(rfe) fit Fold3.Rep4 size:   5
## -(rfe) fit Fold3.Rep4 size:   5
## +(rfe) fit Fold3.Rep4 size:   2
## -(rfe) fit Fold3.Rep4 size:   2
## +(rfe) fit Fold4.Rep4 size: 414
## -(rfe) fit Fold4.Rep4 size: 414
## +(rfe) imp Fold4.Rep4
## -(rfe) imp Fold4.Rep4
## +(rfe) fit Fold4.Rep4 size:  50
## -(rfe) fit Fold4.Rep4 size:  50
## +(rfe) fit Fold4.Rep4 size:  40
## -(rfe) fit Fold4.Rep4 size:  40
## +(rfe) fit Fold4.Rep4 size:  30
## -(rfe) fit Fold4.Rep4 size:  30
## +(rfe) fit Fold4.Rep4 size:  25
## -(rfe) fit Fold4.Rep4 size:  25
## +(rfe) fit Fold4.Rep4 size:  20
## -(rfe) fit Fold4.Rep4 size:  20
## +(rfe) fit Fold4.Rep4 size:  15
## -(rfe) fit Fold4.Rep4 size:  15
## +(rfe) fit Fold4.Rep4 size:  10
## -(rfe) fit Fold4.Rep4 size:  10
## +(rfe) fit Fold4.Rep4 size:   5
## -(rfe) fit Fold4.Rep4 size:   5
## +(rfe) fit Fold4.Rep4 size:   2
## -(rfe) fit Fold4.Rep4 size:   2
## +(rfe) fit Fold5.Rep4 size: 414
## -(rfe) fit Fold5.Rep4 size: 414
## +(rfe) imp Fold5.Rep4
## -(rfe) imp Fold5.Rep4
## +(rfe) fit Fold5.Rep4 size:  50
## -(rfe) fit Fold5.Rep4 size:  50
## +(rfe) fit Fold5.Rep4 size:  40
## -(rfe) fit Fold5.Rep4 size:  40
## +(rfe) fit Fold5.Rep4 size:  30
## -(rfe) fit Fold5.Rep4 size:  30
## +(rfe) fit Fold5.Rep4 size:  25
## -(rfe) fit Fold5.Rep4 size:  25
## +(rfe) fit Fold5.Rep4 size:  20
## -(rfe) fit Fold5.Rep4 size:  20
## +(rfe) fit Fold5.Rep4 size:  15
## -(rfe) fit Fold5.Rep4 size:  15
## +(rfe) fit Fold5.Rep4 size:  10
## -(rfe) fit Fold5.Rep4 size:  10
## +(rfe) fit Fold5.Rep4 size:   5
## -(rfe) fit Fold5.Rep4 size:   5
## +(rfe) fit Fold5.Rep4 size:   2
## -(rfe) fit Fold5.Rep4 size:   2
## +(rfe) fit Fold1.Rep5 size: 414
## -(rfe) fit Fold1.Rep5 size: 414
## +(rfe) imp Fold1.Rep5
## -(rfe) imp Fold1.Rep5
## +(rfe) fit Fold1.Rep5 size:  50
## -(rfe) fit Fold1.Rep5 size:  50
## +(rfe) fit Fold1.Rep5 size:  40
## -(rfe) fit Fold1.Rep5 size:  40
## +(rfe) fit Fold1.Rep5 size:  30
## -(rfe) fit Fold1.Rep5 size:  30
## +(rfe) fit Fold1.Rep5 size:  25
## -(rfe) fit Fold1.Rep5 size:  25
## +(rfe) fit Fold1.Rep5 size:  20
## -(rfe) fit Fold1.Rep5 size:  20
## +(rfe) fit Fold1.Rep5 size:  15
## -(rfe) fit Fold1.Rep5 size:  15
## +(rfe) fit Fold1.Rep5 size:  10
## -(rfe) fit Fold1.Rep5 size:  10
## +(rfe) fit Fold1.Rep5 size:   5
## -(rfe) fit Fold1.Rep5 size:   5
## +(rfe) fit Fold1.Rep5 size:   2
## -(rfe) fit Fold1.Rep5 size:   2
## +(rfe) fit Fold2.Rep5 size: 414
## -(rfe) fit Fold2.Rep5 size: 414
## +(rfe) imp Fold2.Rep5
## -(rfe) imp Fold2.Rep5
## +(rfe) fit Fold2.Rep5 size:  50
## -(rfe) fit Fold2.Rep5 size:  50
## +(rfe) fit Fold2.Rep5 size:  40
## -(rfe) fit Fold2.Rep5 size:  40
## +(rfe) fit Fold2.Rep5 size:  30
## -(rfe) fit Fold2.Rep5 size:  30
## +(rfe) fit Fold2.Rep5 size:  25
## -(rfe) fit Fold2.Rep5 size:  25
## +(rfe) fit Fold2.Rep5 size:  20
## -(rfe) fit Fold2.Rep5 size:  20
## +(rfe) fit Fold2.Rep5 size:  15
## -(rfe) fit Fold2.Rep5 size:  15
## +(rfe) fit Fold2.Rep5 size:  10
## -(rfe) fit Fold2.Rep5 size:  10
## +(rfe) fit Fold2.Rep5 size:   5
## -(rfe) fit Fold2.Rep5 size:   5
## +(rfe) fit Fold2.Rep5 size:   2
## -(rfe) fit Fold2.Rep5 size:   2
## +(rfe) fit Fold3.Rep5 size: 414
## -(rfe) fit Fold3.Rep5 size: 414
## +(rfe) imp Fold3.Rep5
## -(rfe) imp Fold3.Rep5
## +(rfe) fit Fold3.Rep5 size:  50
## -(rfe) fit Fold3.Rep5 size:  50
## +(rfe) fit Fold3.Rep5 size:  40
## -(rfe) fit Fold3.Rep5 size:  40
## +(rfe) fit Fold3.Rep5 size:  30
## -(rfe) fit Fold3.Rep5 size:  30
## +(rfe) fit Fold3.Rep5 size:  25
## -(rfe) fit Fold3.Rep5 size:  25
## +(rfe) fit Fold3.Rep5 size:  20
## -(rfe) fit Fold3.Rep5 size:  20
## +(rfe) fit Fold3.Rep5 size:  15
## -(rfe) fit Fold3.Rep5 size:  15
## +(rfe) fit Fold3.Rep5 size:  10
## -(rfe) fit Fold3.Rep5 size:  10
## +(rfe) fit Fold3.Rep5 size:   5
## -(rfe) fit Fold3.Rep5 size:   5
## +(rfe) fit Fold3.Rep5 size:   2
## -(rfe) fit Fold3.Rep5 size:   2
## +(rfe) fit Fold4.Rep5 size: 414
## -(rfe) fit Fold4.Rep5 size: 414
## +(rfe) imp Fold4.Rep5
## -(rfe) imp Fold4.Rep5
## +(rfe) fit Fold4.Rep5 size:  50
## -(rfe) fit Fold4.Rep5 size:  50
## +(rfe) fit Fold4.Rep5 size:  40
## -(rfe) fit Fold4.Rep5 size:  40
## +(rfe) fit Fold4.Rep5 size:  30
## -(rfe) fit Fold4.Rep5 size:  30
## +(rfe) fit Fold4.Rep5 size:  25
## -(rfe) fit Fold4.Rep5 size:  25
## +(rfe) fit Fold4.Rep5 size:  20
## -(rfe) fit Fold4.Rep5 size:  20
## +(rfe) fit Fold4.Rep5 size:  15
## -(rfe) fit Fold4.Rep5 size:  15
## +(rfe) fit Fold4.Rep5 size:  10
## -(rfe) fit Fold4.Rep5 size:  10
## +(rfe) fit Fold4.Rep5 size:   5
## -(rfe) fit Fold4.Rep5 size:   5
## +(rfe) fit Fold4.Rep5 size:   2
## -(rfe) fit Fold4.Rep5 size:   2
## +(rfe) fit Fold5.Rep5 size: 414
## -(rfe) fit Fold5.Rep5 size: 414
## +(rfe) imp Fold5.Rep5
## -(rfe) imp Fold5.Rep5
## +(rfe) fit Fold5.Rep5 size:  50
## -(rfe) fit Fold5.Rep5 size:  50
## +(rfe) fit Fold5.Rep5 size:  40
## -(rfe) fit Fold5.Rep5 size:  40
## +(rfe) fit Fold5.Rep5 size:  30
## -(rfe) fit Fold5.Rep5 size:  30
## +(rfe) fit Fold5.Rep5 size:  25
## -(rfe) fit Fold5.Rep5 size:  25
## +(rfe) fit Fold5.Rep5 size:  20
## -(rfe) fit Fold5.Rep5 size:  20
## +(rfe) fit Fold5.Rep5 size:  15
## -(rfe) fit Fold5.Rep5 size:  15
## +(rfe) fit Fold5.Rep5 size:  10
## -(rfe) fit Fold5.Rep5 size:  10
## +(rfe) fit Fold5.Rep5 size:   5
## -(rfe) fit Fold5.Rep5 size:   5
## +(rfe) fit Fold5.Rep5 size:   2
## -(rfe) fit Fold5.Rep5 size:   2
```

```
# Extract optimal features
optimal.nfeatures.rfe <- rfe_results$bestSubset
optimal_features_rfe <- rfe_results$optVariables

cat("Optimal number of features:", optimal.nfeatures.rfe, "\n")
```

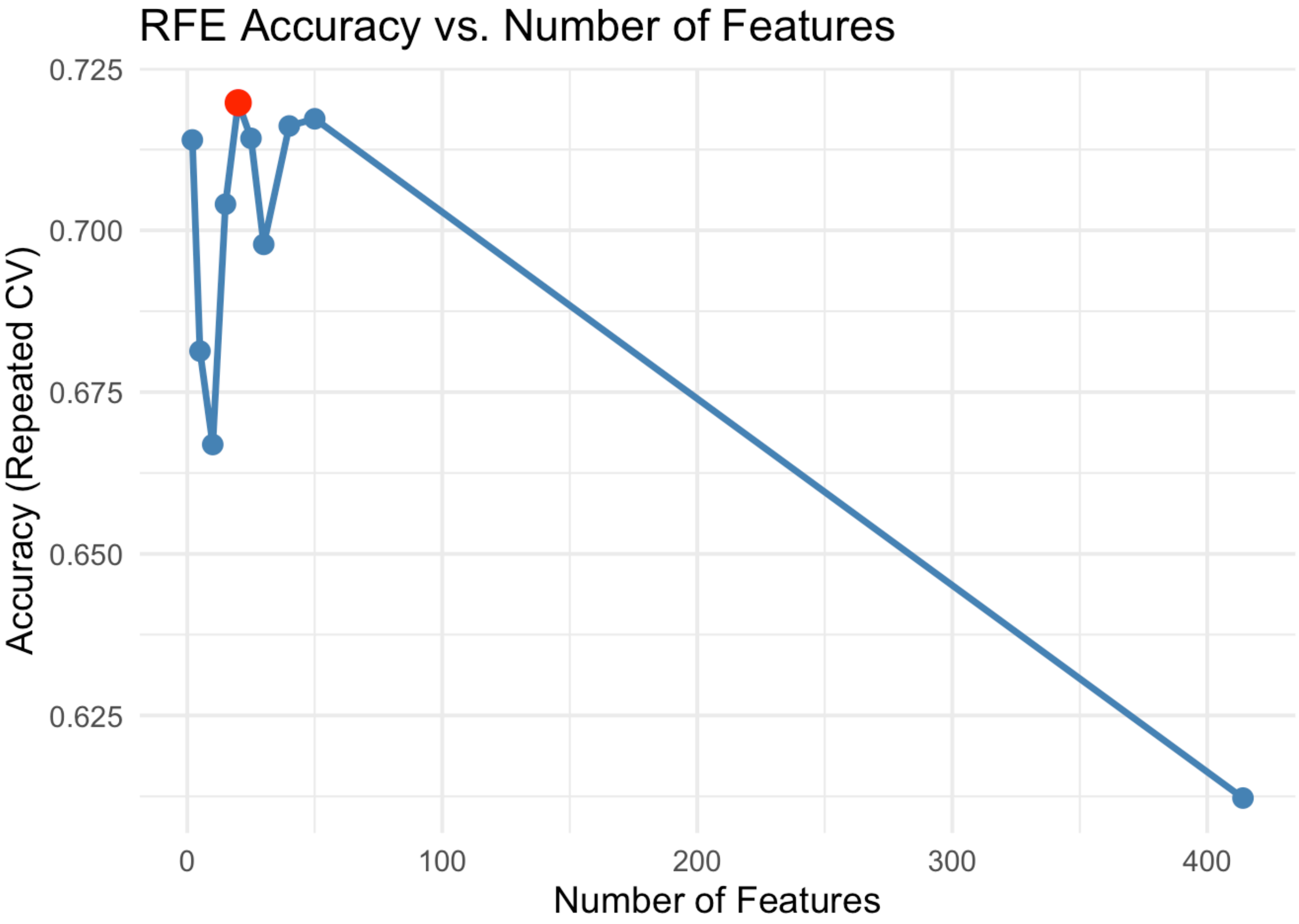
```
## Optimal number of features: 20
```

```
cat("Top selected features:", head(optimal_features_rfe, 10), "\n")
```

```
## Top selected features: X22 X399 X51 X250 X84 X57 X113 X291 X10 X180
```

Plot the results of the RFE function and print the table

```
# Extract resampling results from RFE
rfe_df <- rfe_results$results

# Plot accuracy vs. number of variables
ggplot(rfe_df, aes(x = Variables, y = Accuracy)) +
  geom_line(color = "steelblue", size = 1.2) +
  geom_point(size = 3, color = "steelblue") +
  geom_point(data = subset(rfe_df, Variables == rfe_results$bestSubset),
             aes(x = Variables, y = Accuracy),
             color = "red", size = 4, shape = 21, fill = "red") +
  theme_minimal(base_size = 14) +
  labs(title = "RFE Accuracy vs. Number of Features",
       x = "Number of Features",
       y = "Accuracy (Repeated CV)")
```



```
# Print the table with the results to validate the plot representation
kable(round(rfe_df, 3), caption = "RFE Accuracy by Number of Features")
```

RFE Accuracy by Number of Features

Variables	Accuracy	Kappa	AccuracySD	KappaSD

2	0.714	0.425	0.134	0.270
5	0.681	0.362	0.090	0.179
10	0.667	0.331	0.095	0.194
15	0.704	0.408	0.097	0.192
20	0.720	0.439	0.098	0.195
25	0.714	0.428	0.083	0.165
30	0.698	0.395	0.096	0.194
40	0.716	0.432	0.095	0.190
50	0.717	0.436	0.114	0.228
414	0.612	0.225	0.103	0.206

Final Model Training and Evaluation with Optimal Features

```
# 1. Define the number of components (LVs)
# Use the previously optimized ncomp = 3.
optimal_ncomp <- 3

# 2. Subset the training and test data to only the optimal features
X_train_optimal <- X_train[, optimal_features_rfe]
X_test_optimal  <- X_test[, optimal_features_rfe]

# 3. Train the final PLS-DA model using ONLY the optimal features
pls_final_optimal <- mixOmics::plsda(X_train_optimal, y_train, ncomp = optimal_ncomp)

cat("Final PLS-DA model trained with:", optimal.nfeatures.rfe, "metabolites.\n")
```

```
## Final PLS-DA model trained with: 20 metabolites.
```

```
# 4. Predict on the Test Set
pred_test_optimal <- predict(pls_final_optimal, X_test_optimal)

# Extract predicted class using max distance for the chosen ncomp
pred_class_optimal <- pred_test_optimal$class$max.dist[, optimal_ncomp]

# 5. Evaluate Performance
cat("\n--- Performance on Test Set ---\n")
```

```
##
## --- Performance on Test Set ---
```

```
# Confusion Matrix
confusion_matrix_optimal <- table(True = y_test, Predicted = pred_class_optimal)
print(confusion_matrix_optimal)
```

```
##         Predicted
## True     Female Male
##   Female     16    2
##   Male        5   13
```

```
# Overall Accuracy
accuracy_optimal <- mean(pred_class_optimal == y_test)
cat("\nOverall Accuracy:", round(accuracy_optimal, 4), "\n")
```

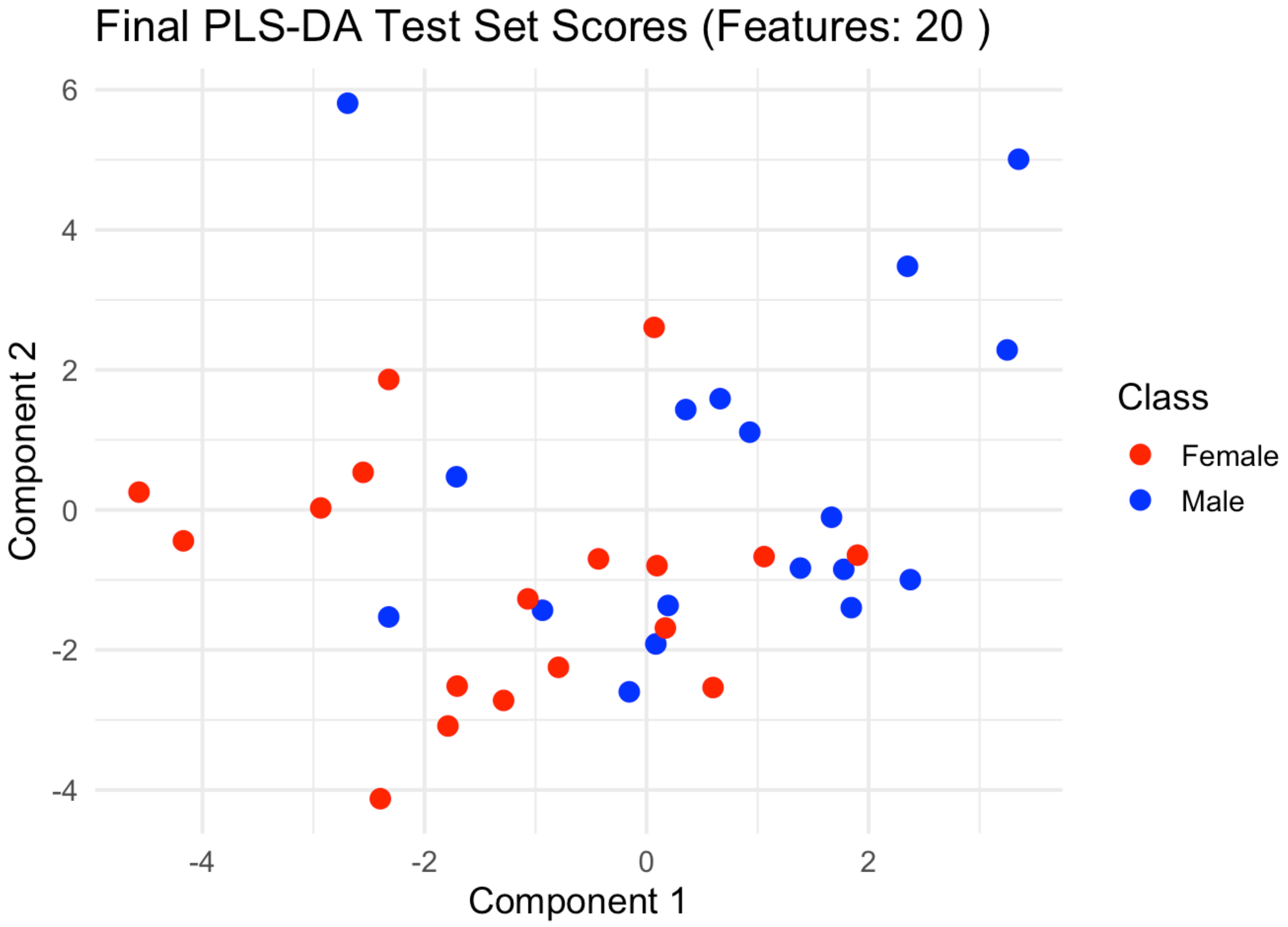
```
##
## Overall Accuracy: 0.8056
```

```
# Custom BER function (can't be exported directly)
BER <- function(conf_matrix) {
  # Convert to matrix
  cm <- as.matrix(conf_matrix)

  # Assume binary classification: rows = true, cols = predicted
  # Sensitivity for class 1
  sens1 <- cm[1,1] / sum(cm[1,])
  # Sensitivity for class 2
  sens2 <- cm[2,2] / sum(cm[2,])

  # BER = 1 - mean(sensitivities)
  ber <- 1 - mean(c(sens1, sens2))
  return(ber)
}

ber_optimal <- BER(confusion_matrix_optimal)
cat("Balanced Error Rate (BER):", round(ber_optimal, 4), "\n")
```

```
## Balanced Error Rate (BER): 0.1944
```

Visualization of Test Set Scores

```
# Visualization of Test Set Scores
# Extract the sample scores (variates) for the optimal model
test_variates_optimal <- pred_test_optimal$variates

# Create a data frame for plotting
test_df_optimal <- data.frame(
  Comp1 = test_variates_optimal[,1],
  Comp2 = if (optimal_ncomp >= 2) test_variates_optimal[,2] else rep(0, nrow(test_v
ariates_optimal)),
  Class = y_test
)

# Generate the Score Plot
ggplot(test_df_optimal, aes(Comp1, Comp2, color = Class)) +
  geom_point(size = 3) +
  theme_minimal(base_size = 14) +
  scale_color_manual(values = c("Female" = "red", "Male" = "blue")) +
  ggtitle(paste("Final PLS-DA Test Set Scores (Features:", optimal.nfeatures.rfe,
")")) +
  labs(x = "Component 1", y = "Component 2")
```



External validation of the results

```
# --- External Validation on Test Set with Optimal Features ---

# Ensure test and train sets have the same features
X_test_optimal <- X_test[, optimal_features_rfe]

# Predict with the final PLS-DA model trained on optimal features
pred_test_optimal <- predict(pls_final_optimal, X_test_optimal)

# Extract predicted classes using max distance
pred_class_optimal <- pred_test_optimal$class$max.dist[, optimal_ncomp]

# Confusion Matrix
conf_matrix <- table(True = y_test, Predicted = pred_class_optimal)
cat("Confusion Matrix:\n")
```

```
## Confusion Matrix:
```

```
print(conf_matrix)
```

```
##         Predicted
## True     Female Male
##   Female     16    2
##   Male        5   13
```

```
# Overall Accuracy
accuracy <- mean(pred_class_optimal == y_test)
cat("\nOverall Accuracy:", round(accuracy, 4), "\n")
```

```
##
## Overall Accuracy: 0.8056
```

```
# Balanced Error Rate (BER)
BER <- function(cm) {
  cm <- as.matrix(cm)
  sens <- diag(cm) / rowSums(cm)
  1 - mean(sens)
}
ber <- BER(conf_matrix)
cat("Balanced Error Rate (BER):", round(ber, 4), "\n")
```

```
## Balanced Error Rate (BER): 0.1944
```

```
# ROC & AUC for binary classification
y_test_num <- as.numeric(y_test) - 1
pred_scores <- pred_test_optimal$predict[,1,optimal_ncomp]

roc_obj <- roc(y_test_num, pred_scores)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls > cases
```

```
auc_val <- auc(roc_obj)
cat("AUC:", round(auc_val, 4), "\n")
```

```
## AUC: 0.8333
```

```
# --- Bootstrap Confidence Intervals ---
set.seed(123)
n_boot <- 1000
boot_acc <- numeric(n_boot)
boot_auc <- numeric(n_boot)

for (i in 1:n_boot) {
  idx <- sample(seq_along(y_test), replace = TRUE)
  Xb <- X_test_optimal[idx, ]
  yb <- y_test[idx]

  pred_b <- predict(pls_final_optimal, Xb)
  class_b <- pred_b$class$max.dist[, optimal_ncomp]
  boot_acc[i] <- mean(class_b == yb)

  scores_b <- pred_b$predict[,1,optimal_ncomp]
  yb_num <- as.numeric(yb) - 1

  if (length(unique(yb_num)) == 2) {
    roc_b <- roc(yb_num, scores_b, quiet = TRUE)
    boot_auc[i] <- auc(roc_b)
  } else {
    boot_auc[i] <- NA
  }
}

# Report mean and 95% CI
acc_ci <- quantile(boot_acc, probs = c(0.025, 0.975))
auc_ci <- quantile(boot_auc, probs = c(0.025, 0.975), na.rm = TRUE)

cat("\n--- Bootstrap Confidence Intervals (n =", n_boot, ") ---\n")
```

```
##
## --- Bootstrap Confidence Intervals (n = 1000 ) ---
```

```
cat("Accuracy: Mean =", round(mean(boot_acc), 4),
    " | 95% CI =", round(acc_ci[1], 4), "-", round(acc_ci[2], 4), "\n")
```

```
## Accuracy: Mean = 0.8064  | 95% CI = 0.6667 - 0.9167
```

```
cat("AUCROC:  Mean =", round(mean(boot_auc, na.rm = TRUE), 4),
    " | 95% CI =", round(auc_ci[1], 4), "-", round(auc_ci[2], 4), "\n")
```

```
## AUCROC:  Mean = 0.8332  | 95% CI = 0.6889 - 0.9524
```

```
# --- Figure of Merit Uncertainty (Bootstrap 95% CI) ---

# Point estimates on the held-out test set
conf_matrix <- table(True = y_test, Predicted = pred_class_optimal)

# Helper metrics (binary, rows=True: Female, Male; cols=Predicted)
metrics_point <- (function(cm) {
  cm <- as.matrix(cm)
  TP1 <- cm[1,1]; FN1 <- cm[1,2]; FP1 <- cm[2,1]; TN1 <- cm[2,2]
  acc <- (TP1 + TN1) / sum(cm)
  sens_fem <- TP1 / (TP1 + FN1)
  spec_fem <- TN1 / (TN1 + FP1)
  sens_male <- TN1 / (TN1 + FP1)        # sensitivity for "Male" if treating Male a
s positive in its own class
  spec_male <- TP1 / (TP1 + FN1)
  ber <- 1 - mean(c(sens_fem, sens_male))
  po <- acc
  pe <- ((sum(cm[1,]) * sum(cm[,1])) + (sum(cm[2,]) * sum(cm[,2]))) / (sum(cm)^2)
  kappa <- (po - pe) / (1 - pe)
  list(accuracy = acc, BER = ber,
       sensitivity_female = sens_fem, specificity_female = spec_fem,
       sensitivity_male = sens_male, specificity_male = spec_male,
       kappa = kappa)
})(conf_matrix)

# AUC point estimate (scores from your model)
y_test_num <- as.numeric(y_test) - 1
pred_scores <- pred_test_optimal$predict[,1,optimal_ncomp]
auc_point <- as.numeric(auc(roc(y_test_num, pred_scores)))
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls > cases
```

```
cat("\n--- Point estimates (Test Set) ---\n")
```

```
##
## --- Point estimates (Test Set) ---
```

```
cat(sprintf("Accuracy = %.4f\n", metrics_point$accuracy))
```

```
## Accuracy = 0.8056
```

```
cat(sprintf("BER      = %.4f\n", metrics_point$BER))
```

```
## BER      = 0.1944
```

```
cat(sprintf("Kappa    = %.4f\n", metrics_point$kappa))
```

```
## Kappa    = 0.6111
```

```
cat(sprintf("Sens(F)  = %.4f | Spec(F) = %.4f\n", metrics_point$sensitivity_female,
metrics_point$specificity_female))
```

```
## Sens(F)  = 0.8889 | Spec(F) = 0.7222
```

```
cat(sprintf("Sens(M)  = %.4f | Spec(M) = %.4f\n", metrics_point$sensitivity_male, m
etrics_point$specificity_male))
```

```
## Sens(M)  = 0.7222 | Spec(M) = 0.8889
```

```
cat(sprintf("AUCROC   = %.4f\n", auc_point))
```

```
## AUCROC   = 0.8333
```

```
# --- Bootstrap CIs for Accuracy, BER, Kappa, Sens/Spec (both classes), and AUC ---
set.seed(123)
n_boot <- 1000

boot_acc  <- numeric(n_boot)
boot_ber  <- numeric(n_boot)
```

```
boot_kappa<- numeric(n_boot)
boot_sensF<- numeric(n_boot)
boot_specF<- numeric(n_boot)
boot_sensM<- numeric(n_boot)
boot_specM<- numeric(n_boot)
boot_auc  <- numeric(n_boot)

for (i in 1:n_boot) {
  idx <- sample(seq_along(y_test), replace = TRUE)
  Xb <- X_test_optimal[idx, ]
  yb <- y_test[idx]

  pred_b <- predict(pls_final_optimal, Xb)
  class_b <- pred_b$class$max.dist[, optimal_ncomp]
  cm <- table(True = yb, Predicted = class_b)

  # Guard: ensure 2x2 matrix even if a class is missing
  all_lvls <- levels(y_test)
  cm <- as.matrix(cm)
  # Rebuild complete 2x2 in fixed order (Female, Male)
  full_cm <- matrix(0, nrow = 2, ncol = 2,
                    dimnames = list(True = all_lvls, Predicted = all_lvls))
  for (tr in rownames(cm)) for (pr in colnames(cm)) full_cm[tr, pr] <- cm[tr, pr]
  cm <- full_cm

  TP1 <- cm["Female","Female"]; FN1 <- cm["Female","Male"]
  FP1 <- cm["Male","Female"];   TN1 <- cm["Male","Male"]

  acc <- (TP1 + TN1) / sum(cm)
  sensF <- if ((TP1 + FN1) > 0) TP1 / (TP1 + FN1) else NA
  specF <- if ((TN1 + FP1) > 0) TN1 / (TN1 + FP1) else NA
  sensM <- if ((TN1 + FP1) > 0) TN1 / (TN1 + FP1) else NA
  specM <- if ((TP1 + FN1) > 0) TP1 / (TP1 + FN1) else NA
  ber   <- 1 - mean(c(sensF, sensM), na.rm = TRUE)

  po <- acc
  pe <- ((sum(cm["Female",]) * sum(cm[,"Female"])) + (sum(cm["Male",]) * sum(cm[,"M
ale"]))) / (sum(cm)^2)
  kappa <- if ((1 - pe) > 0) (po - pe) / (1 - pe) else NA

  # AUC from scores (skip if a single class)
  scores_b <- pred_b$predict[,1,optimal_ncomp]
  yb_num <- as.numeric(yb) - 1
  if (length(unique(yb_num)) == 2) {
    boot_auc[i] <- as.numeric(auc(roc(yb_num, scores_b, quiet = TRUE)))
  } else {
    boot_auc[i] <- NA
  }

  boot_acc[i]   <- acc
  boot_ber[i]   <- ber
  boot_kappa[i] <- kappa
  boot_sensF[i] <- sensF
  boot_specF[i] <- specF
  boot_sensM[i] <- sensM
  boot_specM[i] <- specM
}

q <- function(x) quantile(x, probs = c(0.025, 0.975), na.rm = TRUE)

acc_ci   <- q(boot_acc)
ber_ci   <- q(boot_ber)
kappa_ci <- q(boot_kappa)
sensF_ci <- q(boot_sensF)
specF_ci <- q(boot_specF)
sensM_ci <- q(boot_sensM)
specM_ci <- q(boot_specM)
auc_ci   <- q(boot_auc)

cat("\n--- Bootstrap 95% CIs (n = ", n_boot, ") ---\n", sep = "")
```

```
##
## --- Bootstrap 95% CIs (n = 1000) ---
```

```
cat(sprintf("Accuracy: Mean = %.4f | 95%% CI = %.4f – %.4f\n", mean(boot_acc, na.rm
= TRUE), acc_ci[1], acc_ci[2]))
```

```
## Accuracy: Mean = 0.8064 | 95% CI = 0.6667 – 0.9167
```

```
cat(sprintf("BER:      Mean = %.4f | 95%% CI = %.4f – %.4f\n", mean(boot_ber, na.rm
= TRUE), ber_ci[1], ber_ci[2]))
```

```
## BER:      Mean = 0.1933 | 95% CI = 0.0789 – 0.3287
```

```
cat(sprintf("Kappa:    Mean = %.4f | 95%% CI = %.4f – %.4f\n", mean(boot_kappa, na.
rm = TRUE), kappa_ci[1], kappa_ci[2]))
```

```
## Kappa:    Mean = 0.6072 | 95% CI = 0.3388 – 0.8344
```

```
cat(sprintf("Sens(F):  Mean = %.4f | 95%% CI = %.4f – %.4f\n", mean(boot_sensF, na.
rm = TRUE), sensF_ci[1], sensF_ci[2]))
```

```
## Sens(F):  Mean = 0.8890 | 95% CI = 0.7368 – 1.0000
```

```
cat(sprintf("Spec(F):  Mean = %.4f | 95%% CI = %.4f – %.4f\n", mean(boot_specF, na.rm = TRUE), specF_ci[1], specF_ci[2]))
```

```
## Spec(F):  Mean = 0.7244 | 95% CI = 0.5000 – 0.9231
```

```
cat(sprintf("Sens(M):  Mean = %.4f | 95%% CI = %.4f – %.4f\n", mean(boot_sensM, na.rm = TRUE), sensM_ci[1], sensM_ci[2]))
```

```
## Sens(M):  Mean = 0.7244 | 95% CI = 0.5000 – 0.9231
```

```
cat(sprintf("Spec(M):  Mean = %.4f | 95%% CI = %.4f – %.4f\n", mean(boot_specM, na.rm = TRUE), specM_ci[1], specM_ci[2]))
```

```
## Spec(M):  Mean = 0.8890 | 95% CI = 0.7368 – 1.0000
```

```
cat(sprintf("AUCROC:   Mean = %.4f | 95%% CI = %.4f – %.4f\n", mean(boot_auc, na.rm = TRUE), auc_ci[1], auc_ci[2]))
```

```
## AUCROC:   Mean = 0.8332 | 95% CI = 0.6889 – 0.9524
```