Untitled

Anishkumar Senthil

2025-11-05

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

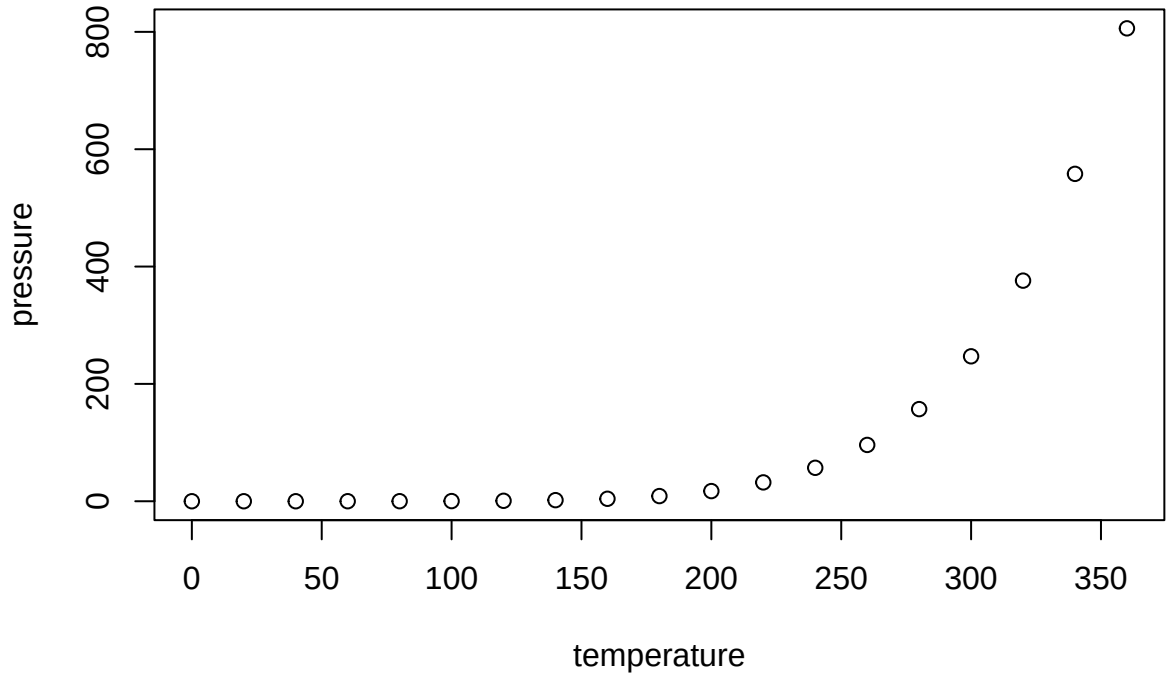
When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
summary(cars)
```

```
##      speed           dist       
##  Min.   : 4.0   Min.   :  2.00  
##  1st Qu.:12.0   1st Qu.: 26.00  
##  Median :15.0   Median : 36.00  
##  Mean   :15.4   Mean   : 42.98  
##  3rd Qu.:19.0   3rd Qu.: 56.00  
##  Max.   :25.0   Max.   :120.00
```

Including Plots

You can also embed plots, for example:



Select top predictive proteins (choose a larger number than 10)

The chunk below loads the processed data created by `preprocessing.R` and selects a configurable number of top predictive proteins using three selection methods: (1) t-test ranking by p-value, (2) Random Forest variable importance (MeanDecreaseGini), and (3) LASSO (glmnet) coefficient magnitude. Change `top_n` to any number > 10.

```
# choose how many top proteins to select (default: 20)
top_n <- 20

# load processed data (adjust relative path if you move this Rmd)
load('../data/biomarker-clean.RData') # creates object `biomarker_clean`

library(dplyr)

dat <- biomarker_clean
dat$group <- factor(dat$group)

proteins <- setdiff(names(dat), c('group','ados'))

# 1) t-test ranking
tt_res <- sapply(proteins, function(p){
    x <- dat[[p]]
    grp <- dat$group
    ok <- !is.na(x) & !is.na(grp)
```

```
    if(sum(ok) < 3) return(NA)
    t <- try(t.test(x[ok] ~ grp[ok]), silent=TRUE)
    if(inherits(t,'try-error')) return(NA)
    t$p.value
})
tt_df <- tibble::tibble(protein = proteins,
                                            pvalue = as.numeric(tt_res)) %>%
    arrange(pvalue) %>%
    slice_head(n = top_n)

# 2) Random Forest importance
if(!requireNamespace('randomForest', quietly = TRUE)) {
    stop("Package 'randomForest' required. Please install it before running this chunk.")
}
rf_dat <- dat %>% dplyr::select(dplyr::all_of(proteins))
rf_resp <- dat$group
rf_fit <- randomForest::randomForest(x = rf_dat, y = rf_resp, ntree = 1000, importance = TRUE)
rf_imp_mat <- randomForest::importance(rf_fit, type = 2)
# importance returns a matrix; use the first column (MeanDecreaseGini) if present
rf_imp_val <- if(is.matrix(rf_imp_mat)) rf_imp_mat[,1] else rf_imp_mat
rf_df <- tibble::tibble(protein = names(rf_imp_val), importance = as.numeric(rf_imp_val)) %>%
    arrange(desc(importance)) %>%
    slice_head(n = top_n)

# 3) LASSO (glmnet)
if(!requireNamespace('glmnet', quietly = TRUE)) {
    stop("Package 'glmnet' required. Please install it before running this chunk.")
}
X <- as.matrix(rf_dat)
# convert factor to 0/1
y <- as.numeric(dat$group) - 1
# fit LASSO with cross-validation
cv <- glmnet::cv.glmnet(X, y, family = 'binomial', alpha = 1, nfolds = 5)
# extract coefficients at the lambda that minimizes CV error
# use the generic coef() so S3 dispatch finds the cv.glmnet method
coef_min <- as.matrix(coef(cv, s = 'lambda.min'))
# coef matrix includes intercept in row 1; align proteins accordingly
coefs <- coef_min[-1,1]
lasso_df <- tibble::tibble(protein = proteins, coef = as.numeric(coefs)) %>%
    mutate(abscoef = abs(coef)) %>%
    arrange(desc(abscoef)) %>%
    slice_head(n = top_n)

# collect top lists
tt_top <- tt_df$protein
rf_top <- rf_df$protein
lasso_top <- lasso_df$protein

cat(glue::glue("Selected top {top_n} proteins by each method:\n"))
```

```
## Selected top 20 proteins by each method:
```

```r
cat("- t-test (by p-value):\n")
```

```
## - t-test (by p-value):
```

```r
print(tt_top)
```

```
##  [1] "DERM"            "RELT"            "FSTL1"           "C1QR1"
##  [5] "Calcineurin"     "CXCL16, soluble" "IgD"             "MRC2"
##  [9] "PTN"             "Cadherin-5"      "MAPK2"           "TGF-b R III"
## [13] "DAF"             "MIA"             "Notch 1"         "gp130, soluble"
## [17] "MMP-2"           "ALCAM"           "ROR1"            "MATN2"
```

```r
cat('\n- Random Forest (importance):\n')
```

```
##
## - Random Forest (importance):
```

```r
print(rf_top)
```

```
##  [1] "DERM"        "ERBB1"       "IgD"         "MAPK14"      "RELT"
##  [6] "SOST"        "MMP-2"       "TGF-b R III" "M2-PK"       "Cadherin-5"
## [11] "ALCAM"       "FSTL1"       "Notch 1"     "CSK"         "MAPK2"
## [16] "TSP4"        "PTN"         "FSTL3"       "CK-MB"       "EPHB2"
```

```r
cat('\n- LASSO (coef magnitude):\n')
```

```
##
## - LASSO (coef magnitude):
```

```r
print(lasso_top)
```

```
##  [1] "IgD"                       "DERM"
##  [3] "14-3-3 protein zeta/delta" "Epo"
##  [5] "MAPK2"                     "ENTP5"
##  [7] "Protein S"                 "IL-17 RC"
##  [9] "SRCN1"                     "FSTL1"
## [11] "CD59"                      "TWEAKR"
## [13] "IL-6 sRa"                  "PAI-1"
## [15] "ITI heavy chain H4"        "PYY"
## [17] "CSRP3"                     "HGFA"
## [19] "hnRNP K"                   "FAM3D"
```

```r
# summary: how many unique proteins across methods
unique_proteins <- unique(c(tt_top, rf_top, lasso_top))
cat('\nTotal unique proteins across methods: ', length(unique_proteins), '\n')
```

```
##
## Total unique proteins across methods:  45
```

```r
# also provide tables for downstream use
selected_lists <- list(tt = tt_df, rf = rf_df, lasso = lasso_df)

selected_lists
```

```
## $tt
## # A tibble: 20 x 2
##    protein               pvalue
##    <chr>                  <dbl>
##  1 DERM            0.00000000827
##  2 RELT            0.0000000782
##  3 FSTL1           0.000000466
##  4 C1QR1           0.000000479
##  5 Calcineurin     0.000000537
##  6 CXCL16, soluble 0.000000875
##  7 IgD             0.000000933
##  8 MRC2            0.00000103
##  9 PTN             0.00000135
## 10 Cadherin-5      0.00000175
## 11 MAPK2           0.00000204
## 12 TGF-b R III     0.00000330
## 13 DAF             0.00000397
## 14 MIA             0.00000483
## 15 Notch 1         0.00000500
## 16 gp130, soluble  0.00000530
## 17 MMP-2           0.00000552
## 18 ALCAM           0.00000664
## 19 ROR1            0.00000786
## 20 MATN2           0.00000799
##
## $rf
## # A tibble: 20 x 2
##    protein     importance
##    <chr>            <dbl>
##  1 DERM             0.763
##  2 ERBB1            0.529
##  3 IgD              0.526
##  4 MAPK14           0.518
##  5 RELT             0.500
##  6 SOST             0.500
##  7 MMP-2            0.444
##  8 TGF-b R III      0.439
##  9 M2-PK            0.397
## 10 Cadherin-5       0.390
## 11 ALCAM            0.381
## 12 FSTL1            0.376
## 13 Notch 1          0.373
## 14 CSK              0.365
## 15 MAPK2            0.362
## 16 TSP4             0.360
## 17 PTN              0.343
## 18 FSTL3            0.326
## 19 CK-MB            0.312
```

```
## 20 EPHB2            0.295
##
## $lasso
## # A tibble: 20 x 3
##    protein                     coef absoef
##    <chr>                      <dbl>  <dbl>
##  1 IgD                        0.593  0.593
##  2 DERM                       0.538  0.538
##  3 14-3-3 protein zeta/delta  0.337  0.337
##  4 Epo                        0.303  0.303
##  5 MAPK2                      0.295  0.295
##  6 ENTP5                     -0.287  0.287
##  7 Protein S                  0.251  0.251
##  8 IL-17 RC                  -0.241  0.241
##  9 SRCN1                      0.235  0.235
## 10 FSTL1                      0.229  0.229
## 11 CD59                      -0.223  0.223
## 12 TWEAKR                    -0.182  0.182
## 13 IL-6 sRa                  -0.181  0.181
## 14 PAI-1                     -0.168  0.168
## 15 ITI heavy chain H4         0.168  0.168
## 16 PYY                        0.167  0.167
## 17 CSRP3                     -0.156  0.156
## 18 HGFA                       0.142  0.142
## 19 hnRNP K                    0.137  0.137
## 20 FAM3D                     -0.136  0.136
```

Note that the `echo = FALSE` parameter was added to the code chunk above to prevent printing of the R code that generated the plot.