The seventyGenesData package: annotated gene expression data from the van't Veer and Van de Vijver breast cancer cohorts

Luigi Marchionni[1]

[1]The Sidney Kimmel Comprehensive Cancer Center, Johns Hopkins University School of Medicine

May 20, 2021

Contents

1 Overview

This vignette documents the R code used to retrieve from the original sources, annotate, and then assemble into separate ExpressionSet instances the gene expression data of the van't Veer and Van de Vijver cohorts [2, 1].

2 Original data sources

The original data sets used to develop and validate the 70-genes signature [2, 1] were dowloaded from the website of the Bioinformatics and Statistics group of the Netherlands Cancer Institute (NKI) (http://bioinformatics.nki.nl/data.php), and as supplementary material associated with the original manuscripts, using the following hyperlinks:

- van't Veer cohort (http://bioinformatics.nki.nl/data/van-t-Veer_Nature_2002/):
  - Training set expression data: good prognosis
  - Training set expression data: bad prognosis
  - Test set expression data
  - Expression data for BRCA1 mutated cases
  - Microarray feature annotation
  - Annotation explanation
  - Microarray feature sequences
  - Explanation of files content
  - Explanation of acronyms used in the files
  - Clinical information (from Nature)
  - 70-gene signature data (from Nature)
- Van de Vijveer cohort:
  - Clinical data
  - The expression data for 70-Gene signature across the 295 samples
  - 70-Gene Template for Good Prognosis Signature
  - Genome-Wide Gene Expression Data for 295 Samples

All the files, among those mentioned above, needed to compile this vignette and create the seventyGeneData Rpackage are included inside the inst/extdata directory.

The microarray feature annotation can be obtained from the breastCancerNKI R-Bioconductor package, in which the two cohorts are merged in a unique ExpressionSet instance. On the contrary in the present R-Bioconductor package the two data sets are maintained separate, the training and test sets of sampes used in the original studies are identified, along whith the microarray features constituting the 70-gene signature.

Below is the code chunk used to download the van't Veer's cohort data from NKI and from the Supplementary Information section associated with the original manuscript. This code chunk was not run to

compile this vignette, which was prepared using the local copies of the data stored in the `inst/extdata` directory.

```
> ###Create a working directory
> dir.create("../extdata/vantVeer", showWarnings = FALSE, recursive=TRUE)
> ###Create the url list for all supplementary data on the Nature Website
> nkiUrl <- "http://bioinformatics.nki.nl/data/van-t-Veer_Nature_2002/"
> natureUrl <- "http://www.nature.com/nature/journal/v415/n6871/extref/"
> urlList <- c(
               paste(nkiUrl, sep="",
                     c("ArrayData_greater_than_5yr.zip",
                       "ArrayData_less_than_5yr.zip", "ArrayData_19samples.zip",
                       "ArrayData_BRCA1.zip", "ArrayNomenclature_contig_accession.xls",
                       "ArrayNomenclature_methods.doc", "ProbeSeq.xls",
                       "README-Nature_I.doc", "codeboek_Rosetta.doc")),
               paste(natureUrl, sep="",
                    c("415530a-s7.doc", "415530a-s8.xls",
                      "415530a-s9.xls", "415530a-s10.xls", "415530a-s11.xls"))
              )
> ###Dowload all files from Nature and NKI
> lapply(urlList, function(x) {
         download.file(x, destfile=paste("../extdata/vantVeer/", gsub(".+/", "", x), sep=""),
                       quiet = FALSE, mode = "w", cacheOK = TRUE)
 })
```

Below is the code to download the Van de Vijver's cohort data from NKI. This code chunk was not run to compile this vignette, which was preapred using the local copies of the data stored in the `inst/extdata` directory.

```
> ###Create a working directory
> dir.create("../extdata/vanDeVijver", showWarnings = FALSE, recursive=TRUE)
> ###Create the url list for all supplementary data on the NKI Website
> nkiUrl <- "http://bioinformatics.nki.nl/data/"
> urlList <- paste(nkiUrl, sep="", c("nejm_table1.zip", "ZipFiles295Samples.zip") )
> ###Dowload all files from NKI
> lapply(urlList, function(x) {
         download.file(x, destfile=paste("../extdata/vanDeVijver/", gsub(".+/", "", x), sep=""),
                       quiet = FALSE, mode = "w", cacheOK = TRUE)
 })
```

Below is the code to download and install the `breastCancerNKI` annotation metadata package from `Bioconductor`, along with other required libraries if the are not available (e.g `gdata`). As shown below the `breastCancerNKI` contains a unique instance of class `ExpressionSet` for both cohorts, wich includes the expression data, the phenotypic information, and the feature annotation.

```
> ###Get the list of available packages
> installedPckgs <- installed.packages()[,"Package"]
> ###Define the list of desired libraries
> pckgListBIOC <- c("Biobase", "limma", "breastCancerNKI", "gdata")
> ###Use the BiocManager package from Bioconductor
```

```
> if (!requireNamespace("BiocManager", quietly=TRUE))
    install.packages("BiocManager")
> ###Load the packages, install them from Bioconductor if needed
> for (pckg in pckgListBIOC) {
        if (! pckg %in% installedPckgs) BiocManager::install(pckg)
        require(pckg, character.only=TRUE)
 }
```

3 ExpressionSet preparation

3.1 Feature data preparation

Below is the code used to assemble the microarray feature annotation obtained from all the various sources described above, including the information about gene membership to the original 70-gene signature identified in the van't Veer study [2], as available from the supplementary data section of the original publication. To this end the Excel spreadsheet named `415530a-s9.xls` corresponds to Table S2 of the original manuscript, and contains the identifiers and annotation for the 231 genes that proved to be associated with metastatic recurrence at 5 years. According to the information contained in the legend for Table S2 ("Supplementary Methods" section, file `415530a-s7.doc`) the optimal set of genes constituting the 70-gene signature can be obtained from this table by selecting the top 70 features/reporters with highest absolute correlation coefficients.

The code chunk below shows how to manipulate the annotation contained in the `breastCancerNKI`:

```
> ###Load the library with annotation
> require(breastCancerNKI)
> ###Load the dataset
> data(nki)
> ###Check dataset classes and attributes
> class(nki)

[1] "ExpressionSet"
attr(,"package")
[1] "Biobase"

> dim(nki)

Features  Samples
   24481      337

> ###Check featureData
> str(featureData(nki))

Formal class 'AnnotatedDataFrame' [package "Biobase"] with 4 slots
  ..@ varMetadata      :'data.frame':        10 obs. of  1 variable:
  .. ..$ labelDescription: chr [1:10] "probe" "EntrezGene.ID" "probe.name" "Alignment.score" ...
  ..@ data             :'data.frame':        24481 obs. of  10 variables:
  .. ..$ probe              : Factor w/ 24496 levels "AA017147_RC",..: 10190 10003 13371 13503 6
  .. ..$ EntrezGene.ID      : int [1:24481] 64388 140883 NA 2244 286133 NA 5879 NA NA NA ...
  .. ..$ probe.name         : Factor w/ 14960 levels "AA043429_RC",..: 3465 3383 NA 5118 2353 NA
  .. ..$ Alignment.score    : int [1:24481] 60 60 NA 60 60 NA 60 NA NA NA ...
  .. ..$ Length.of.probe    : int [1:24481] 60 60 NA 60 60 NA 60 NA NA NA ...
```

```
.. ..$ NCBI.gene.symbol   : Factor w/ 13119 levels "2'-PDE","76P",..: 5030 11223 NA 4230 10102
.. ..$ HUGO.gene.symbol   : Factor w/ 12566 levels "-","A1BG","A2M",..: 4824 10702 NA 4156 960
.. ..$ Cytoband           : Factor w/ 2484 levels "-",".|3q28-q29",..: 1086 1205 NA 1689 2156
.. ..$ Alternative.symbols: Factor w/ 12067 levels "(ALPHA)II-SPECTRIN|FLJ44613",..: 3131 160
.. ..$ Description        : Factor w/ 13119 levels "1-acylglycerol-3-phosphate O-acyltransfera
..@ dimLabels        : chr [1:2] "featureNames" "featureColumns"
..@ .__classVersion__:Formal class 'Versions' [package "Biobase"] with 1 slot
.. .. ..@ .Data:List of 1
.. .. .. ..$ : int [1:3] 1 1 0
.. .. ..$ names: chr "AnnotatedDataFrame"
```

```
> nkiAnn <- featureData(nki)
> ###Turn all annotation information into character
> nkiAnn@data <- as.data.frame(apply(nkiAnn@data, 2, as.character),
                               stringsAsFactors=FALSE)
```

The code chunk below shows how to read and manipulate the annotation contained in the files obtained from the Nature and the NKI websites:

```
> ###Load the library
> require(gdata)
> ###Read GBACC information for van't Veer dataset
> myFile<- system.file("extdata/vantVeer", "ArrayNomenclature_contig_accession.xls",
                       package = "seventyGeneData")
> featAcc <- read.xls(myFile, skip=0, header=TRUE, stringsAsFactors=FALSE)
> ###Read seq information for van't Veer dataset
> myFile <- system.file("extdata/vantVeer", "ProbeSeq.xls",
                       package = "seventyGeneData")
> featSeq <- read.xls(myFile, skip=0, header=TRUE, stringsAsFactors=FALSE)
> ###Read 70-genes signature information for van't Veer dataset
> myFile <- system.file("extdata/vantVeer", "415530a-s9.xls",
                       package = "seventyGeneData")
> gns231 <- read.xls(myFile, skip=0, header=TRUE, stringsAsFactors=FALSE)
> ###Remove special characters in the colums header,
> ###which are due to white spaces present in the Excel files
> colnames(gns231) <- gsub("\\.\\.", "", colnames(gns231))
> ###Remove GO annotation
> gns231 <- gns231[, -grep("sp_xref_keyword_list", colnames(gns231))]
> ###Reorder the genes in decreasing order by absolute correlation
> gns231 <- gns231[order(abs(gns231$correlation), decreasing=TRUE),]
> ###Select the feature identifiers corresponding to the top 231 and 70 genes
> gns231$genes231 <- TRUE
> gns231$genes70  <- gns231$accession %in% gns231$accession[1:70]
> ###Merge all information (including 70-gene signature information)
> ###with the annotation obtained from the breastCancerNKI package
> newAnn <- nkiAnn@data
> newAnn <- merge(newAnn, featAcc, by.x=1, by.y=1, all=TRUE, sort=FALSE)
> newAnn <- merge(newAnn, featSeq, by.x=1, by.y=1, all=TRUE, sort=FALSE)
> newAnn <- merge(newAnn, gns231, by.x=1, by.y=1, all=TRUE, sort=FALSE)
```

All the available annotation information is shown below:

```
> ###Check the structure of the new annotation data.frame
> newAnn <- newAnn[order(newAnn[,1]),]
> str(newAnn)

'data.frame':        24481 obs. of  17 variables:
 $ probe                   : chr  "AA017147_RC" "AA024493_RC" "AA043429_RC" "AA470152_RC" ...
 $ EntrezGene.ID           : chr  NA NA " 57674" NA ...
 $ probe.name              : chr  NA NA "AA043429_RC" NA ...
 $ Alignment.score         : chr  NA NA "60" NA ...
 $ Length.of.probe         : chr  NA NA "60" NA ...
 $ NCBI.gene.symbol        : chr  NA NA "C17orf27" NA ...
 $ HUGO.gene.symbol        : chr  NA NA "C17orf27" NA ...
 $ Cytoband                : chr  NA NA "17q25.3" NA ...
 $ Alternative.symbols     : chr  NA NA "KIAA1554" NA ...
 $ Description             : chr  NA NA "chromosome 17 open reading frame 27" NA ...
 $ Genbank.Accession.Number: chr  NA NA NA NA ...
 $ Oligo.probe.sequence    : chr  "TTTCTAATCCACATTTCACAATTCTCTCAAAACTCAATCCTAACACTCTCTCACAAACAC'
 $ correlation             : num  NA NA NA NA NA ...
 $ gene.name               : chr  NA NA NA NA ...
 $ description             : chr  NA NA NA NA ...
 $ genes231                : logi  NA NA NA NA NA NA ...
 $ genes70                 : logi  NA NA NA NA NA NA ...
```

3.2 Assemble the `vantVeer ExpressionSet`

For the van't Veer cohort the gene expression data for 117 samples is contained in the following four different compressed files.

- `ArrayData_19samples.zip`, containing data for 19 samples (test set);
- `ArrayData_BRCA1.zip`, containing data for 20 samples (BRCA1 and 2 mutation).
- `ArrayData_greater_than_5yr.zip`, containing data for 51 samples;
- `ArrayData_less_than_5yr.zip`, containing data for 46 samples;

Once decompressed such archives correspond to 4 distinct Excel spreadsheets containing three (3) columns for each sample, organized as follows:

- Log ratio;
- P-value for log ratio significance;
- Log intensity;

However the annotation information contained in these spreadsheets has different format, with unmatched quotes and a variable number of columns corresponding to the systematic feature name, the gene name, and its annotation. Hence such spreadsheets are somewhat difficult to read directly into `R` with the `read.xls` function contained in the `gdata`. For this reason, after extracting the corresponding compressed Excel files, we have saved all the spreadsheets as TAB-delimited text files externally, and we have provided them with the `seventyGeneData` package inside the `inst/extdata` directory. Finally, the Sample IDs

used in such gene expression tables correspond to those used in the clinical data table (archived in the `415530a-s8.xls`).

The Rcode used to assemble the `ExpressionSet` instance for the van't Veer cohort contained in the `seventyGenesData` R-Bioconductor package is shown below. Since gene expression from the original van't Veer and Van de Vijver cohorts was provided in the form of pre-processed, normalized, and summarized values, pre-processing is not required.

The code chunk below shows how to list the available files containing gene expression measurements:

```
> ###Load the library
> require(Biobase)
> require(gdata)
> ###Check presence of dowloaded file
> filesVtVloc <- system.file("extdata/vantVeer", package = "seventyGeneData")
> dir(filesVtVloc)
```

```
[1] "415530a-s8.xls"
[2] "415530a-s9.xls"
[3] "ArrayData_19samples_.txt.gz"
[4] "ArrayData_BRCA1.txt.gz"
[5] "ArrayData_greater_than_5yr.txt.gz"
[6] "ArrayData_less_than_5yr.txt.gz"
[7] "ArrayNomenclature_contig_accession.xls"
[8] "ProbeSeq.xls"
```

```
> ###Create list of files to be read in
> filesVtV <- dir(filesVtVloc, full.names=TRUE, pattern="^ArrayData")
> filesVtV
```

```
[1] "/tmp/Rtmp6pKEaH/Rinst273c716b68b778/seventyGeneData/extdata/vantVeer/ArrayData_19samples_.t
[2] "/tmp/Rtmp6pKEaH/Rinst273c716b68b778/seventyGeneData/extdata/vantVeer/ArrayData_BRCA1.txt.gz
[3] "/tmp/Rtmp6pKEaH/Rinst273c716b68b778/seventyGeneData/extdata/vantVeer/ArrayData_greater_than
[4] "/tmp/Rtmp6pKEaH/Rinst273c716b68b778/seventyGeneData/extdata/vantVeer/ArrayData_less_than_5y
```

The code chunk below shows how to read and the phenotypic information associated with the samples analyzed in the van't Veer cohort:

```
> myFile <- system.file("extdata/vantVeer", "415530a-s8.xls",
                        package = "seventyGeneData")
> ###Read phenotypic information
> phenoVtV <- read.xls(myFile, skip=0, header=TRUE, stringsAsFactors=FALSE)
> ###Show Phenotypic information
> str(phenoVtV)
```

```
'data.frame':        117 obs. of  11 variables:
 $ Sample..             : int  1 2 3 4 5 6 7 8 9 10 ...
 $ age                  : int  43 44 41 41 48 49 46 48 48 38 ...
 $ diameter.mm.         : int  25 20 45 20 20 13 20 28 15 15 ...
 $ followup.time.yr.    : num  12.53 6.44 10.66 13 11.98 ...
 $ metastases           : int  0 0 0 0 0 0 0 0 0 0 ...
 $ grade                : int  2 1 3 3 3 2 1 3 3 2 ...
```

```
$ Angioinvasion         : int  1 0 1 0 1 0 0 0 0 0 ...
$ ERp                   : int  80 50 10 50 100 80 80 0 60 100 ...
$ PRp                   : int  80 50 5 70 80 80 50 0 80 10 ...
$ Lymphocytic.Infiltrate: int  0 0 0 1 0 0 0 1 0 0 ...
$ Brca1.mutation        : chr  "0" "0" "0" "0" ...
```

The code chunk below shows how to process the phenotypic information associated with the samples analyzed in the van't Veer cohort. In particular it is important to check that samples order is the same in the expression data and phenotypic information tables.

```
> ###Remove the special characters in the colums headers
> ###due to white spaces present in the Excel file
> colnames(phenoVtV) <- gsub("\\.$", "", gsub("\\.$", "", colnames(phenoVtV)))
> ####Remove columns that do not contain useful information
> phenoVtV <- phenoVtV[ , apply(phenoVtV, 2, function(x) length(unique(x)) > 1 )]
> phenoVtV$SampleName <- paste("Sample", phenoVtV$Sample)
> rownames(phenoVtV) <- phenoVtV$SampleName
> ###Read sample names from the 6 expression data tables
> samplesVtV <- lapply(filesVtV, read.table, nrow=1, header=FALSE, sep="\t",
                       stringsAsFactors=FALSE, fill=TRUE, strip.white=TRUE)
> ###Format the samples strings
> samplesVtV <- lapply(samplesVtV, function(x) x[ grep("^Sample", x) ])
> headerDesc <- samplesVtV
> samplesVtV <- lapply(samplesVtV, function(x) gsub(",.+", "", x) )
```

The code chunk below shows how to compare the order of the samples in the expression data spreadsheets and the phenotypic information table.

```
> ###Check sample lables obtained from expression data files
> str(samplesVtV)
```

```
List of 4
 $ : chr [1:19] "Sample 111" "Sample 112" "Sample 113" "Sample 114" ...
 $ : chr [1:20] "Sample 80" "Sample 81" "Sample 83" "Sample 84" ...
 $ : chr [1:44] "Sample 1" "Sample 2" "Sample 3" "Sample 4" ...
 $ : chr [1:34] "Sample 45" "Sample 46" "Sample 47" "Sample 48" ...
```

```
> ###Combine the lables in one unique vector
> allSamplesVtV <- do.call("c", samplesVtV)
> ###Compare order the order the samples between the expression data
> ###and phenotypic information data.frames
> if (all(rownames(phenoVtV) %in% allSamplesVtV)) {
         print("All sample names match phenoData")
         if (all(rownames(phenoVtV) == allSamplesVtV)) {
                 print("All sample names match phenoData")
         } else {
                 print("Sample names from tables and phenoData need reordering")
                 phenoVtV <- phenoVtV[order(phenoVtV$SampleName), ]
         }
 } else {
         print("Sample names DO NOT match phenoData")
```

```
 }
```

```
[1] "All sample names match phenoData"
[1] "Sample names from tables and phenoData need reordering"
```

The code chunk below shows how to read the expression data from the spreadsheets:

```
> ###Read expression data from the 4 converted TAB-delimited text files
> dataVtV <- lapply(filesVtV, read.table, skip=1, sep="\t", quote="",
                header=TRUE, row.names=NULL,
                stringsAsFactors=FALSE, fill=FALSE, strip.white=FALSE)
> sapply(dataVtV, dim)
```

```
      [,1]  [,2]  [,3]  [,4]
[1,] 24481 24481 24481 24481
[2,]    59    62   134   104
```

```
> ###Extract annotation: note that column headers are slightly different
> sapply(dataVtV, function(x) head(colnames(x)) )
```

```
     [,1]                [,2]                [,3]
[1,] "Systematic.name"   "Systematic.name"   "Systematic.name"
[2,] "Gene.name"         "Gene.name"         "Gene.name"
[3,] "log10.Intensity."  "log10.Intensity."  "log10.Intensity."
[4,] "Log10.ratio."      "Log10.ratio."      "Log10.ratio."
[5,] "P.value"           "P.value"           "P.value"
[6,] "log10.Intensity..1" "log10.Intensity..1" "log10.Intensity..1"
     [,4]
[1,] "Systematic.name"
[2,] "Gene.name"
[3,] "log10.Intensity."
[4,] "Log10.ratio."
[5,] "P.value"
[6,] "log10.Intensity..1"
```

```
> sapply(dataVtV, function(x) tail(colnames(x)) )
```

```
     [,1]                 [,2]                 [,3]
[1,] "log10.Intensity..17" "log10.Intensity..18" "log10.Intensity..42"
[2,] "Log10.ratio..17"     "Log10.ratio..18"     "Log10.ratio..42"
[3,] "P.value.17"          "P.value.18"          "P.value.42"
[4,] "log10.Intensity..18" "log10.Intensity..19" "log10.Intensity..43"
[5,] "Log10.ratio..18"     "Log10.ratio..19"     "Log10.ratio..43"
[6,] "P.value.18"          "P.value.19"          "P.value.43"
     [,4]
[1,] "log10.Intensity..32"
[2,] "Log10.ratio..32"
[3,] "P.value.32"
[4,] "log10.Intensity..33"
[5,] "Log10.ratio..33"
[6,] "P.value.33"
```

```
> ###Extract the associated annotation
> annVtV <- lapply(dataVtV, function(x) x[,c("Systematic.name", "Gene.name")])
> annVtV <- lapply(annVtV, function(x) {x[x==""] <- NA ; x })
> annVtV <- do.call("cbind", annVtV)
```

The code chunk below shows how to check the annotation information contained in the expression data spreadsheets:

```
> ###Check annotation order in all data files
> if ( all(apply(annVtV[, seq(1, 8, by=2)], 1, function(x) length(unique(x)) == 1 )) ) {
         print("OK")
         annVtV <- annVtV[,1:2]
 } else {
         print("Check annotation")
 }
```

```
[1] "OK"
```

Create a function that extracts columns based on UNIQUE patterns and sets rownames using annotation and reorder the rows:

```
> ###Define the function
> extractColumns <- function(x, pattern, ann) {
         sel <- grep(pattern, colnames(x), value=TRUE)
         x <- x[,sel ]
         rownames(x) <- ann
         x <- x[order(rownames(x)), ]
 }
```

The code chunk below shows how to read the four files and assemble the log-ratio expression data:

```
> ###Extract log ratio data from all the spreadsheets
> logRat <- lapply(dataVtV, extractColumns, pattern="Log10\\.ratio", ann=annVtV[,1])
> logRat <- do.call("cbind", logRat)
> ###Assign colnames and reorder the columns
> colnames(logRat) <- allSamplesVtV
> logRat <- logRat[, order(colnames(logRat)),]
```

The code chunk below shows how to check that the samples are similarly ordered in the assembled gene expression and phenotypes data.frames:

```
> ###Check order
> all(phenoVtV$SampleName == colnames(logRat))
```

```
[1] TRUE
```

The code chunk below shows how to read the four files and assemble the p-values associated with the expression data:

```
> ###Extract p-values from all the spreadsheets
> pVal <- lapply(dataVtV, extractColumns, pattern="value", ann=annVtV[,1])
> pVal <- do.call("cbind", pVal)
> ###Assign colnames and reorder the columns
```

```
> colnames(pVal) <- allSamplesVtV
> pVal <- pVal[, order(colnames(pVal)),]
```

The code chunk below shows how to check that the samples are similarly ordered in the assembled p-values and phenotypes data.frames:

```
> ###Check order
> all(phenoVtV$SampleName == colnames(pVal))
```

```
[1] TRUE
```

The code chunk below shows how to read the four files and assemble the expression intensities provided with the data:

```
> ###Extract expression intensity from all the spreadsheets
> intensity <- lapply(dataVtV, extractColumns, pattern="Intensity", ann=annVtV[,1])
> intensity <- do.call("cbind", intensity)
> ###Assign colnames and reorder the columns
> colnames(intensity) <- allSamplesVtV
> intensity <- intensity[, order(colnames(intensity)),]
```

The code chunk below shows how to check that the samples are similarly ordered in the assembled expression intensity and phenotypes data.frames:

```
> ###Check order
> all(phenoVtV$SampleName == colnames(intensity))
```

```
[1] TRUE
```

The code chunk below shows how to create an instance of class `ExpressionSet` for the van't Veer cohort.

```
> ###Merge annotation objects and check order
> annVtV <- merge(annVtV, newAnn, by=1, all=TRUE, sort=TRUE)
> rownames(annVtV) <- annVtV[,1]
> all(rownames(annVtV) == rownames(logRat))
```

```
[1] TRUE
```

```
> all(rownames(annVtV) == rownames(pVal))
```

```
[1] TRUE
```

```
> all(rownames(annVtV) == rownames(intensity))
```

```
[1] TRUE
```

```
> ###Create the new assayData
> myAssayData <- assayDataNew(exprs=logRat, pValue=pVal, intensity=intensity)
> ###Create the new phenoData
> myPhenoData <- new("AnnotatedDataFrame", phenoVtV)
> ###Create the new featureData
> myFeatureData <- new("AnnotatedDataFrame", annVtV)
> ###Create the new experimentData
> myExperimentData <- new("MIAME", name = "Marc J Van De Vijver, Hongyue Dai, and Laura J van't
                          lab = "The Netherland Cancer Institute, Amsterdam, The Netherlands",
```

```
                        contact = "Luigi Marchionni <marchion@gmail.com>",
                        title = "Gene expression profiling predicts clinical outcome of breast
                        abstract = "Breast cancer patients with the same stage of disease can h
  The strongest predictors for metastases (for example, lymph node status and histological grade)
  Chemotherapy or hormonal therapy reduces the risk of distant metastases by approximately one-th
  None of the signatures of breast cancer gene expression reported to date allow for patient-tail
  Here we used DNA microarray analysis on primary breast tumours of 117 young patients, and appli
  In addition, we established a signature that identifies tumours of BRCA1 carriers. The poor pro
  This gene expression profile will outperform all currently used clinical parameters in predicti
                        url = "http://www.ncbi.nlm.nih.gov/pubmed/?term=11823860",
                        pubMedIds = "11823860" )
> ###Create the expression set
> vantVeer <- new("ExpressionSet",
                    assayData = myAssayData,
                    phenoData = myPhenoData,
                    featureData = myFeatureData,
                    experimentData = myExperimentData)
```

3.3 Assemble the `vanDeVijver ExpressionSet`

For the Van de Vijver cohort the gene expression data for 295 samples is contained in the `ZipFiles295Samples.zip` compressed archive. Such data is organized in six separate text files:

-
- Table_NKI_295_1.txt, containing data for 50 samples;
- `Table_NKI_295_2.txt`, containing data for 50 samples.
- `Table_NKI_295_3.txt`, containing data for 50 samples;
- `Table_NKI_295_4.txt`, containing data for 50 samples;
- `Table_NKI_295_5.txt`, containing data for 50 samples;
- `Table_NKI_295_6.txt`, containing data for 45 samples;

All the files, amon those mentioned above, needed to compile this vignette and create the `seventyGeneData` Rpackage are included inside the `inst/extdata` directory.

Each row in each expression data spreadsheet corresponds to a specific feature on the microrray. The first two columns provide the systematic substance name and the gene name when available. Starting from the third column each sample accounts for five distinct and consecutive columns:

- Log ratio;
- Log ratio error;
- P-value for log ratio significance;
- Log intensity;
- Flag for each feature(0 for control or bad spot, = 1 for valid measurement);

The Sample ID used in the gene expression tables correspond to those used in the clinical data table (archived in the `nejm_table1.zip` compressed file).

The code used to assemble the `ExpressionSet` instance for the van't Veer cohort contained in the `seventyGenesData` `R-Bioconductor` package is shown below. Since gene expression from the original van't Veer and Van de Vijver cohorts was provided in the form of pre-processed, normalized, and summarized values, pre-processing is not required.

The code chunk below shows how to check the availability of all raw data files obtained from the NKI website.

```
> ##################################################
> ###Load the library
> require(Biobase)
> require(gdata)
> ##################################################
> ###Check presence of dowloaded files
> dir("../inst/extdata/vanDeVijver")
```

```
[1] "ZipFiles295Samples.zip" "nejm_table1.zip"
```

The code chunk below shows how to decompress and extract the zipped archives containing the expression data for the Van de Vijver cohort.

```
> ###Check presence of dowloaded file
> filesVdVloc <- system.file("extdata/vanDeVijver", package = "seventyGeneData")
> dir(filesVdVloc)
```

```
[1] "ZipFiles295Samples.zip" "nejm_table1.zip"
```

```
> ###Create list of files to be unzipped and read in
> filesVdVzip <- dir(filesVdVloc, full.names=TRUE)
> filesVdVzip
```

```
[1] "/tmp/Rtmp6pKEaH/Rinst273c716b68b778/seventyGeneData/extdata/vanDeVijver/ZipFiles295Samples.
[2] "/tmp/Rtmp6pKEaH/Rinst273c716b68b778/seventyGeneData/extdata/vanDeVijver/nejm_table1.zip"
```

```
> ###Create output directory
> myTmpDir <- paste(filesVdVloc, "/tmp", sep="")
> ###Decompress expression
> unzip(filesVdVzip[1], exdir=myTmpDir)
> ###Decompress phenoData
> unzip(filesVdVzip[2], exdir=myTmpDir)
> ###List of files in "ZipFiles295Samples.zip" containing expression
> filesVdV <- dir(myTmpDir, full.names=TRUE, pattern="NKI")
> ###Show file list content
> filesVdV
```

```
[1] "/tmp/Rtmp6pKEaH/Rinst273c716b68b778/seventyGeneData/extdata/vanDeVijver/tmp/Table_NKI_295_1
[2] "/tmp/Rtmp6pKEaH/Rinst273c716b68b778/seventyGeneData/extdata/vanDeVijver/tmp/Table_NKI_295_2
[3] "/tmp/Rtmp6pKEaH/Rinst273c716b68b778/seventyGeneData/extdata/vanDeVijver/tmp/Table_NKI_295_3
[4] "/tmp/Rtmp6pKEaH/Rinst273c716b68b778/seventyGeneData/extdata/vanDeVijver/tmp/Table_NKI_295_4
[5] "/tmp/Rtmp6pKEaH/Rinst273c716b68b778/seventyGeneData/extdata/vanDeVijver/tmp/Table_NKI_295_5
[6] "/tmp/Rtmp6pKEaH/Rinst273c716b68b778/seventyGeneData/extdata/vanDeVijver/tmp/Table_NKI_295_6
```

The code chunk below shows how to process the phenotypic information associated with the samples

analyzed in the Van de Vijver cohort. In particular it is important to check that samples order is the same in the expression data and phenotypic information tables.

```
> ###Read phenotypic information
> myFile <- dir(myTmpDir, full.names=TRUE, pattern="Table1_ClinicalData_Table.xls")
> phenoVdV <- read.xls(myFile, skip=2, header=TRUE, stringsAsFactors=FALSE)
> ####Remove columns that do not contain useful information
> phenoVdV <- phenoVdV[ , apply(phenoVdV, 2, function(x) length(unique(x)) > 1 )]
> phenoVdV$SampleName <- paste("Sample", phenoVdV$SampleID)
> rownames(phenoVdV) <- phenoVdV$SampleName
> ###Read sample names from the expression data spreadsheets
> samplesVdV <- lapply(filesVdV, scan, what="character", nlines=1, sep="\t", strip.white=FALSE)
> samplesVdV <- lapply(samplesVdV, function(x) x[x!=""])
> allSamplesVdV <- do.call("c", samplesVdV)
> ###Read all data contained in the expression data spreadsheets
> dataVdV <- lapply(filesVdV, read.table, header=TRUE, skip=1, sep="\t", quote="",
                stringsAsFactors=FALSE, fill=TRUE, strip.white=TRUE)
> ###Extract feature annotation
> annVdV <- lapply(dataVdV, function(x) x[,c("Substance", "Gene")])
> annVdV <- lapply(annVdV, function(x) {x[x==""] <- NA ; x })
> annVdV <- do.call("cbind", annVdV)
```

The code chunk below shows how to check the annotation information contained in the expression data spreadsheets:

```
> ###Check annotation order in all data files
> if ( all(apply(annVdV[, seq(1, 12, by=2)], 1, function(x) length(unique(x)) == 1 )) ) {
         print("OK")
         annVdV <- annVdV[,1:2]
 } else {
         print("Check annotation")
 }
[1] "OK"
```

Create a function that extracts columns based on UNIQUE patterns and sets rownames using annotation and reorder the rows.

```
> ###Define the function
> extractColumns <- function(x, pattern, annVdV) {
         colnames(x) <- gsub("Log\\.Ratio\\.Error", "Error", colnames(x))
         sel <- grep(pattern, colnames(x), value=TRUE)
         x <- x[,sel ]
         rownames(x) <- annVdV
         x <- x[order(rownames(x)), ]
 }
```

The code chunk below shows how to read the four files and assemble the log-ratio expression data:

```
> ###Extract and assemble the log ratio values
> logRat <- lapply(dataVdV, extractColumns, pattern="Log\\.Ratio", ann=annVdV[,1])
> logRat <- do.call("cbind", logRat)
```

```
> ###Set the column names
> colnames(logRat) <- allSamplesVdV
```

The code chunk below shows how to check that the samples are similarly ordered in the assembled gene expression and phenotypes data.frames:

```
> ###Check order
> all(phenoVdV$SampleName == colnames(logRat))
```

```
[1] TRUE
```

The code chunk below shows how to read the four files and assemble the log-ratio error associated with the expression data:

```
> ###Extract log ratio error
> logRatError <- lapply(dataVdV, extractColumns, pattern="Error", ann=annVdV[,1])
> logRatError <- do.call("cbind", logRatError)
> ###Set the column names
> colnames(logRatError) <- allSamplesVdV
```

The code chunk below shows how to check that the samples are similarly ordered in the assembled gene expression error and phenotypes data.frames:

```
> ###Check order
> all(phenoVdV$SampleName == colnames(logRatError))
```

```
[1] TRUE
```

The code chunk below shows how to read the four files and assemble the p-values associated with the expression data:

```
> ###Extract P-value
> pVal <- lapply(dataVdV, extractColumns, pattern="alue", ann=annVdV[,1])
> pVal <- do.call("cbind", pVal)
> ###Set the column names
> colnames(pVal) <- allSamplesVdV
```

The code chunk below shows how to check that the samples are similarly ordered in the assembled p-values and phenotypes data.frames:

```
> ###Check order
> all(phenoVdV$SampleName == colnames(pVal))
```

```
[1] TRUE
```

The code chunk below shows how to read the four files and assemble the expression intensities provided with the data:

```
> ###Extract Intensity
> intensity <- lapply(dataVdV, extractColumns, pattern="Intensity", ann=annVdV[,1])
> intensity <- do.call("cbind", intensity)
> ###Set the column names
> colnames(intensity) <- allSamplesVdV
```

The code chunk below shows how to check that the samples are similarly ordered in the assembled p-values and phenotypes data.frames:

```
> ###Check order
> all(phenoVdV$SampleName == colnames(intensity))

[1] TRUE
```

The code chunk below shows how to create an instance of class `ExpressionSet` for the van't Veer cohort.

```
> ###Merge and check order
> annVdV <- merge(annVdV, newAnn, by=1, all=TRUE, sort=TRUE)
> rownames(annVdV) <- annVdV[,1]
> all(rownames(annVdV) == rownames(logRat))

[1] TRUE

> all(rownames(annVdV) == rownames(logRatError))

[1] TRUE

> all(rownames(annVdV) == rownames(pVal))

[1] TRUE

> all(rownames(annVdV) == rownames(intensity))

[1] TRUE

> ###Create the new assayData
> myAssayData <- assayDataNew(exprs=logRat, exprsError=logRatError,
                              pValue=pVal, intensity=intensity)
> ###Create the new phenoData
> myPhenoData <- new("AnnotatedDataFrame", phenoVdV)
> ###Create the new featureData
> myFeatureData <- new("AnnotatedDataFrame", annVdV)
> ###Create the new experimentData
> myExperimentData <- new("MIAME", name = "Marc J Van De Vijver, Yudong D He, and Laura J van't
                          lab = "The Netherland Cancer Institute, Amsterdam, The Netherlands",
                          contact = "Luigi Marchionni <marchion@gmail.com>",
                          title = "A gene-expresion signature as a predictor  of survival in brea
                          abstract = "Background: A more accurate means of prognostication in bre
 Methods: Using microarray analysis to evaluate our previously established 70-gene prognosis pro
 All patients had stage I or II breast cancer and were younger than 53 years old; 151 had lymph-
 Results: Among the 295 patients, 180 had a poor-prognosis signature and 115 had a good-prognosi
 At 10 years, the probability of remaining free of distant metastases was 50.6+/-4.5 percent in
 The estimated hazard ratio for distant metastases in the group with a poor-prognosis signature,
 This ratio remained significant when the groups were analyzed according to lymph-node status. M
 Conclusions: The gene-expression profile we studied is a more powerful predictor of the outcome
                          url = "http://www.ncbi.nlm.nih.gov/pubmed/?term=12490681",
                          pubMedIds = "12490681" )
> ###Create the expression set
> vanDeVijver <- new("ExpressionSet",
                     assayData = myAssayData,
                     phenoData = myPhenoData,
```

```
                    featureData = myFeatureData,
                    experimentData = myExperimentData)
> ###Remove temporary folder
> file.remove(dir(myTmpDir, full.names=TRUE))
```

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

```
> file.remove(myTmpDir)
```

```
[1] TRUE
```

4 Phenotypic information processing

4.1 Patients' phenotypes in the `vantVeer ExpressionSet`

The code below is to add data set information to the `vantVeer ExpressionSet`.

```
> ###Define the data set type from file of origin
> type <- gsub("..txt", "", gsub(".+ArrayData_", "", filesVtV))
> dataSetType <- mapply(x=samplesVtV, y=type, FUN=function(x,y) {
        rep(y, length(x))
        })
> ###Combine with sample information
> dataSetType <- do.call("c", dataSetType)
> names(dataSetType) <- allSamplesVtV
> ###Reorder
> dataSetType <- dataSetType[order(names(dataSetType))]
```

```
> ###Add the information to pData(vantVeer)
> if (all(rownames(pData(vantVeer)) == names(dataSetType) )) {
        pData(vantVeer)$DataSetType <- dataSetType
        print("Adding information about data set type to pData")
 } else {
        print("Check order pData and data set type information")
 }
```

```
[1] "Adding information about data set type to pData"
```

The `R` code below is to create prognostic groups using recurrence information defined as the occurrence of a metastasis as first event within five years from surgery ("bad" prognosis), as opposed to patients who remained disease free for at least five years ("good" prognosis).

```
> ###Process time metastases (TTM)
> pData(vantVeer)$TTM <- pData(vantVeer)$followup.time.yr
> ####Process TTM event
> pData(vantVeer)$TTMevent <- pData(vantVeer)$metastases
> ####Create binary TTM at 5 years groups
> pData(vantVeer)$FiveYearMetastasis <- pData(vantVeer)$TTM < 5 & pData(vantVeer)$TTMevent == 1
> ###Show structure of updated phenotypes
> str(pData(vantVeer))
```

```
'data.frame':        117 obs. of  16 variables:
 $ Sample               : int  1 10 100 102 103 104 105 106 107 108 ...
```

```
 $ age                   : int  43 38 48 50 42 34 51 35 38 48 ...
 $ diameter.mm           : int  25 15 20 15 30 5 27 15 2 17 ...
 $ followup.time.yr      : num  12.53 6.64 NA 3.29 4.95 ...
 $ metastases            : int  0 0 NA 1 1 1 1 1 1 1 ...
 $ grade                 : int  2 2 3 3 3 2 2 3 3 2 ...
 $ Angioinvasion         : int  1 0 0 1 0 0 1 0 0 1 ...
 $ ERp                   : int  80 100 0 50 0 0 90 0 70 90 ...
 $ PRp                   : int  80 10 0 20 0 0 90 0 70 60 ...
 $ Lymphocytic.Infiltrate: int  0 0 1 0 1 0 0 0 0 0 ...
 $ Brca1.mutation        : chr  "0" "0" "1" "0" ...
 $ SampleName            : chr  "Sample 1" "Sample 10" "Sample 100" "Sample 102" ...
 $ DataSetType           : chr  "greater_than_5y.gz" "greater_than_5y.gz" "BRCA.gz" "19samples.g
 $ TTM                   : num  12.53 6.64 NA 3.29 4.95 ...
 $ TTMevent              : int  0 0 NA 1 1 1 1 1 1 1 ...
 $ FiveYearMetastasis    : logi  FALSE FALSE NA TRUE TRUE TRUE ...
```

```
> ###Save the final ExpressionSet object
> dataDirLoc <- system.file("data", package = "seventyGeneData")
> save(vantVeer, file=paste(dataDirLoc, "/vantVeer.rda", sep=""))
```

4.2 Patients' phenotypes in the `vanDeVijver ExpressionSet`

The `R` code below is to create prognostic groups using recurrence information defined as the occurrence of a metastasis as first event within five years from surgery ("bad" prognosis), as opposed to patients who remained disease free for at least five years ("good" prognosis).x

```
> ###Select  new cases not included in the van't Veer study
> pVDV <- pData(vanDeVijver)
> ###Rename columns
> selNames <- c("TIMEmeta", "EVENTmeta", "TIMEsurvival", "EVENTdeath", "TIMErecurrence")
> newNames <- c("TTM", "TTMevent", "OS", "OSevent", "RFS")
> colnames(pVDV)[ sapply(selNames, grep, colnames(pVDV)) ] <- newNames
> ###Process time metastases (TTM)
> pVDV$TTM[is.nan(pVDV$TTM)] <- pVDV$OS[is.nan(pVDV$TTM)]
> ###Process recurrence free survival (RFS) adding RFSevent
> pVDV$RFSevent <- pVDV$RFS < pVDV$OS
> ###Create binary TTM at 5 years groups selecting:
> ###1) the cases with metastases as first event within 5 years
> badCases <- which(
                    pVDV$TTM <= pVDV$RFS ###Met is 1st recurrence
                    & pVDV$TTMevent == 1 ### Metastases occurred
                    & pVDV$TTM < 5 ### Recurrence within 5 years
                    )
> ###2) the cases disease free for at least 5 years
> goodCases <- which(
                    pVDV$TTM > 5 ### No metastasis before 5 years
                    & pVDV$RFS > 5 ###No recurrence before 5 years
                    & pVDV$TTMevent == 0 ### Metastases did notoccurred
                    )
```

The code chunk below is to check that no patients are duplicated:

```
> ###Check if there are duplicated cased present in both prognostic groups
> all (!goodCases %in% badCases)
```

```
[1] TRUE
```

The code chunk below is to add the prognostic group information to the `pData` component of the ExpressionSet.

```
> ###Create groups by setting all cases to NA and then identifying bad cases
> pVDV$FiveYearMetastasis <- NA
> pVDV$FiveYearMetastasis[badCases] <- TRUE
> ###And then excluding patients with a relapse before a metastasis within 5 years
> pVDV$FiveYearMetastasis[goodCases] <- FALSE
> ###Assign updated phenotypic data
> pData(vanDeVijver) <- pVDV
> ###Show structure of updated phenotypes
> str(pData(vanDeVijver))
```

```
'data.frame':        295 obs. of  18 variables:
 $ SampleID          : int  122 123 124 125 126 127 128 129 130 131 ...
 $ FirstSeriesID     : num  1 5 10 NaN NaN NaN NaN NaN NaN NaN ...
 $ Posnodes          : chr  "n" "n" "n" "y" ...
 $ TTMevent          : int  0 0 0 0 1 1 0 0 0 0 ...
 $ OSevent           : int  0 0 0 0 0 0 0 0 0 1 ...
 $ OS                : num  14.82 14.26 6.64 7.75 6.44 ...
 $ RFS               : num  14.82 14.26 6.64 7.75 6.32 ...
 $ TTM               : num  14.82 14.26 6.64 7.75 6.32 ...
 $ ESR1              : int  1 1 1 1 1 1 1 0 0 ...
 $ NIH               : int  0 0 0 0 0 0 0 0 0 0 ...
 $ StGallen          : int  0 0 0 0 0 0 0 0 0 0 ...
 $ conservFlag       : int  -1 -1 -1 0 0 0 0 0 0 0 ...
 $ C1fromData        : num  0.782 0.636 0.568 0.646 0.342 ...
 $ C1crossvalid      : num  0.78 0.58 0.61 NaN NaN NaN NaN NaN NaN NaN ...
 $ C1used            : num  0.78 0.58 0.61 0.646 0.342 ...
 $ SampleName        : chr  "Sample 122" "Sample 123" "Sample 124" "Sample 125" ...
 $ RFSevent          : logi  FALSE FALSE FALSE FALSE TRUE TRUE ...
 $ FiveYearMetastasis: logi  FALSE FALSE FALSE FALSE NA NA ...
```

```
> ###Save the final ExpressionSet object
> dataDirLoc <- system.file("data", package = "seventyGeneData")
> save(vanDeVijver, file=paste(dataDirLoc, "/vanDeVijver.rda", sep=""))
```

5 System Information

Session information:

```
> sessionInfo()
```

```
R version 4.1.0 (2021-05-18)
Platform: x86_64-pc-linux-gnu (64-bit)
```

```
Running under: Ubuntu 20.04.2 LTS

Matrix products: default
BLAS:   /home/biocbuild/bbs-3.13-bioc/R/lib/libRblas.so
LAPACK: /home/biocbuild/bbs-3.13-bioc/R/lib/libRlapack.so

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
 [3] LC_TIME=en_US.UTF-8        LC_COLLATE=C
 [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
 [9] LC_ADDRESS=C               LC_TELEPHONE=C
[11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

attached base packages:
[1] parallel  stats     graphics  grDevices utils     datasets  methods
[8] base

other attached packages:
[1] gdata_2.18.0           breastCancerNKI_1.30.0 limma_3.48.0
[4] Biobase_2.52.0         BiocGenerics_0.38.0

loaded via a namespace (and not attached):
[1] BiocManager_1.30.15 compiler_4.1.0      tools_4.1.0
[4] gtools_3.8.2
```

6 References

References

[1] M. J. van de Vijver, Y. D. He, L. J. van't Veer, H. Dai, A. A. Hart, D. W. Voskuil, G. J. Schreiber, J. L. Peterse, C. Roberts, M. J. Marton, M. Parrish, D. Atsma, A. Witteveen, A. Glas, L. Delahaye, T. van der Velde, H. Bartelink, S. Rodenhuis, E. T. Rutgers, S. H. Friend, and R. Bernards. A gene-expression signature as a predictor of survival in breast cancer. *N Engl J Med*, 347(25):1999–2009, 2002. 1533-4406 (Electronic) Evaluation Studies Journal Article.

[2] L. J. van't Veer, H. Dai, M. J. van de Vijver, Y. D. He, A. A. Hart, M. Mao, H. L. Peterse, K. van der Kooy, M. J. Marton, A. T. Witteveen, G. J. Schreiber, R. M. Kerkhoven, C. Roberts, P. S. Linsley, R. Bernards, and S. H. Friend. Gene expression profiling predicts clinical outcome of breast cancer. *Nature*, 415(6871):530–6, 2002. 0028-0836 (Print) Journal Article.