Vignette for the *sampleClassifier* R Package

Khadija El Amrani [*†]   Andreas Kurtz[*]

April 24, 2017

Contents

1 Introduction

Discrimination between different classes of samples such as different cell types or tissues using gene expression profiles is an important problem in genetic and cell research. It has several implications and can contribute to our understanding of cell phenotype differences and will allow precise identification of various cell types and tissues. *sampleClassifier* provides functions for the classification of samples using their gene expression profiles. The package supports the classification

[*]Charité - Universitätsmedizin Berlin, Berlin Brandenburg Center for Regenerative Therapies (BCRT)
[†]Package maintainer, Email: `khadija.el-amrani@charite.de`

of microarray and RNA-seq gene expression profiles. The tool requires a reference and a test data set, and uses a simple algorithm called Shared Marker Genes (SMG). As the name suggests, the number of shared marker genes between a reference and a query sample is used as a similarity measure. Marker genes are detected using the tools *MGFM* [1, 2] or *MGFR* [3] that we have developed previously. *sampleClassifier* can be applied: i) to evaluate the similarity of experimentally derived cells with their desired target cell type (e.g. to compare primary hepatocytes with induced hepatocytes); ii) to compare in vitro derived organoids to their in vivo counterparts; iii) to classify different types of diseases.
This vignette provides an introduction to gene expression profile based sample classification using the *sampleClassifier* R package and the accompanying data package, *sampleClassifierData*. It contains guidelines on how to select all inputs to the tool (such as reference and test matrices), and instructions on running *sampleClassifier* using microarray and RNA-seq data from the *sampleClassifierData*.

2 Contents of the package

The *sampleClassifier* package contains the following functions:

```
> library("sampleClassifier")
> ls("package:sampleClassifier")
```

```
[1] "classifyProfile"            "classifyProfile.rnaseq"     "classifyProfile.rnaseq.svm"
[4] "classifyProfile.svm"        "get.heatmap"
```

2.1 `classifyProfile()`

`classifyProfile()` is a function to classify microarray gene expression profiles.

2.1.1 Parameter Settings

1. ref_matrix: Normalized microarray data matrix to be used as reference, with probe sets corresponding to rows and samples corresponding to columns.
2. $query\_mat$: Normalized microarray query matrix with query samples to be classified, with probe sets corresponding to rows and samples corresponding to columns.
3. *chip1* : Chip name of the reference matrix (e.g. 'hgu133plus2').
4. *chip2*: Chip name of the query matrix. This parameter can be ignored if the reference and query matrix are from the same chip.
5. *fun1*: `mean` or `median`. This will specify the number of marker genes that will be used for classification. Default is `median`.
6. *fun2*: `mean` or `median`. This will be used to summarize the expression values of probe sets that belong to the same gene. This parameter can be ignored if the reference and query matrix are from the same chip. Default is `mean`.
7. *write2File*: If TRUE, the classification results for each query profile will be written to a file.
8. *out.dir*: Path to a directory to write the classification results, default is the current working directory.

2.1.2 Output

The function `classifyProfile()` returns a list with hits for each of the query samples in the query matrix. The hits are sorted according to their similarity to the query.

2.2 `classifyProfile.rnaseq()`

`classifyProfile.rnaseq()` is a function to classify RNA-seq gene expression profiles.

2.2.1 Parameter Settings

1. ref_matrix: RNA-seq data matrix to be used as reference, with genes corresponding to rows and samples corresponding to columns.
2. $query\_mat$: RNA-seq query matrix with query samples to be classified, with genes corresponding to rows and samples corresponding to columns.
3. *gene.ids.type* : Type of the used gene identifiers, the following gene identifiers are supported: ensembl, refseq and ucsc gene ids. Default is ensembl.
4. *fun1*: `mean` or `median`. This will specify the number of marker genes that will be used for classification. Default is `median`.
5. *write2File*: If TRUE, the classification results for each query profile will be written to a file.
6. *out.dir*: Path to a directory to write the classification results, default is the current working directory.

2.2.2 Output

The function `classifyProfile.rnaseq()` returns a list with top hits for each query profile, sorted according to a similarity score.

2.3 `classifyProfile.svm()`

`classifyProfile.svm()` is a function to classify microarray gene expression profiles using Support Vector Machines (SVM).

2.3.1 Parameter Settings

1. ref_matrix: Normalized microarray data matrix to be used as reference, with probe sets corresponding to rows and samples corresponding to columns.
2. $query\_mat$: Normalized microarray query matrix to be classified, with probe sets corresponding to rows and samples corresponding to columns.
3. *chip1* : Chip name of the reference matrix (e.g. 'hgu133plus2').
4. *chip2*: Chip name of the query matrix. This parameter can be ignored if the reference and query matrix are from the same chip.
5. *fun1*: `mean` or `median`. This will specify the number of marker genes that will be used for classification. Default is `median`.
6. *fun2*: `mean` or `median`. This will be used to summarize the expression values of probe sets that belong to the same gene. This parameter can be ignored if the reference and query matrix are from the same chip.

2.3.2 Output

The function `classifyProfile.svm()` returns a data frame with the predicted classes for each query profile.

2.4 `classifyProfile.rnaseq.svm()`

`classifyProfile.rnaseq.svm()` is a function to classify RNA-seq gene expression profiles using Support Vector Machines (SVM).

2.4.1 Parameter Settings

1. ref_matrix: RNA-seq data matrix to be used as reference, with genes corresponding to rows and samples corresponding to columns.
2. $query\_mat$: RNA-seq query matrix with query samples to be classified, with genes corresponding to rows and samples corresponding to columns.
3. *gene.ids.type* : Type of the used gene identifiers, the following gene identifiers are supported: ensembl, refseq and ucsc gene ids. Default is ensembl.
4. *fun1*: `mean` or `median`. This will specify the number of marker genes that will be used for classification. Default is `median`.

2.4.2 Output

The function `classifyProfile.rnaseq.svm()` returns a data frame with the predicted classes for each query profile.

Please note that all replicates of a sample type should have the same label in the reference matrix.

2.5 `get.heatmap()`

`get.heatmap()` is a function to display the classification predictions as a heatmap.

2.5.1 Parameter Settings

1. `res.list`: the result list returned by the function `classifyProfile()` or `classifyProfile.rnaseq()`

2.5.2 Output

The function `get.heatmap()` is used only for the side effect of creating a heatmap.

3 Getting Started

The *sampleClassifier* package can be downloaded and installed by running the following code from within R:

```
> source("http://bioconductor.org/biocLite.R")
> biocLite("sampleClassifier")
```

We also recommend installing the accompanying data package, *sampleClassifierData*, which contains pre-processed microarray and RNA-seq data, which are derived from normal human tissues, and are available from public repositories. More details about the data can be found in the Vignette of the data package.

After installing and loading *sampleClassifierData*, the individual sampleClassifierData datasets can be loaded using the function `data()`. For instance, the RNA-seq dataset named se_rnaseq_refmat is stored as a `SummarizedExperiment` object. The numeric matrix can be extracted using the function `assay()` from the *SummarizedExperiment* R package:

```
> library(sampleClassifierData)
> data("se_rnaseq_refmat")
> rnaseq_refmat <- assay(se_rnaseq_refmat)
```

4 Classification using sampleClassifier

We will use the data provided in the data package *sampleClassifierData* to demonstrate how to classify samples using *sampleClassifier*.

4.1 Classification of microarray data

To classify microarray gene expression profiles, we use the function `classifyProfile()`. It expects a reference and a test matrix. We recommend using 3 replicates for each sample type in the reference matrix. Please note that replicates of the same sample type should have the same name in the reference matrix. We also recommend processing the reference and the test matrix in the same way.
As a reference matrix we use here a microarray dataset derived from the study GSE3526 [4], which is available from GEO [5]. This dataset is named se_micro_refmat and can be loaded with the following code:

```
> library(sampleClassifierData)
> data("se_micro_refmat")
> micro_refmat <- assay(se_micro_refmat)
> dim(micro_refmat)
```

```
[1] 54675    78
```

As a test matrix we use a microarray dataset derived from the study GSE2361 [6], which is available from GEO [5]. This dataset is named $se\_micro\_testmat$ and can be loaded with the following code:

```
> data("se_micro_testmat")
> micro_testmat <- assay(se_micro_testmat)
> dim(micro_testmat)
```

```
[1] 22283    16
```

Now, we can call the function `classifyProfile()`:

```
> res1.list <- classifyProfile(ref_matrix=micro_refmat, query_mat=micro_testmat,
+ chip1="hgu133plus2",chip2="hgu133a", write2File=FALSE)
```

```
The reference matrix and the query are from different platforms...
Collapse rows ...
detecting marker genes...
16 profiles to be classified...
done!
```

For simplicity, we show only the two top hits for each query sample:

```
> lapply(res1.list,"[",c(1,2),,drop=FALSE)
```

```
$`GSM44671 : Heart`
                Hits Score   Ratio
1       heart_atrium 0.569 37 / 65
2 trigeminal_ganglia 0.231  3 / 13

$`GSM44673 : Spleen`
         Hits Score   Ratio
1 bone_marrow 0.282 22 / 78
2      spleen 0.256 20 / 78

$`GSM44674 : Ovary`
            Hits Score   Ratio
1          ovary 0.364 20 / 55
```

```
2 saphenous_vein 0.137 10 / 73

$`GSM44675 : Kidney`
                Hits Score   Ratio
1      kidney_cortex 0.462 36 / 78
2 trigeminal_ganglia 0.231  3 / 13

$`GSM44676 : Skeletal Muscle`
             Hits Score   Ratio
1 skeletal_muscle 0.667 52 / 78
2     bone_marrow 0.141 11 / 78

$`GSM44678 : Prostate`
                Hits Score   Ratio
1     prostate_gland 0.256 20 / 78
2 trigeminal_ganglia 0.154  2 / 13

$`GSM44689 : Cerebellum`
        Hits Score   Ratio
1 cerebellum 0.321 25 / 78
2   thalamus   0.3  3 / 10

$`GSM44693 : Bone Marrow`
         Hits Score   Ratio
1 bone_marrow 0.679 53 / 78
2 lymph_nodes 0.122  6 / 49

$`GSM44698 : Thalamus`
                Hits Score  Ratio
1           thalamus   0.7 7 / 10
2 trigeminal_ganglia 0.231 3 / 13

$`GSM44699 : Pituitary Gland`
                Hits Score   Ratio
1    pituitary_gland   0.5 39 / 78
2 trigeminal_ganglia 0.154  2 / 13

$`GSM44700 : Spinal Cord`
                 Hits Score   Ratio
1         spinal_cord 0.278 20 / 72
2 dorsal_root_ganglia 0.194  6 / 31

$`GSM44701 : Testis`
                Hits Score   Ratio
1             testes  0.91 71 / 78
2 trigeminal_ganglia 0.231  3 / 13

$`GSM44702 : Liver`
      Hits Score   Ratio
1    liver 0.859 67 / 78
2 thalamus   0.2  2 / 10

$`GSM44704 : Lung`
                Hits Score   Ratio
```

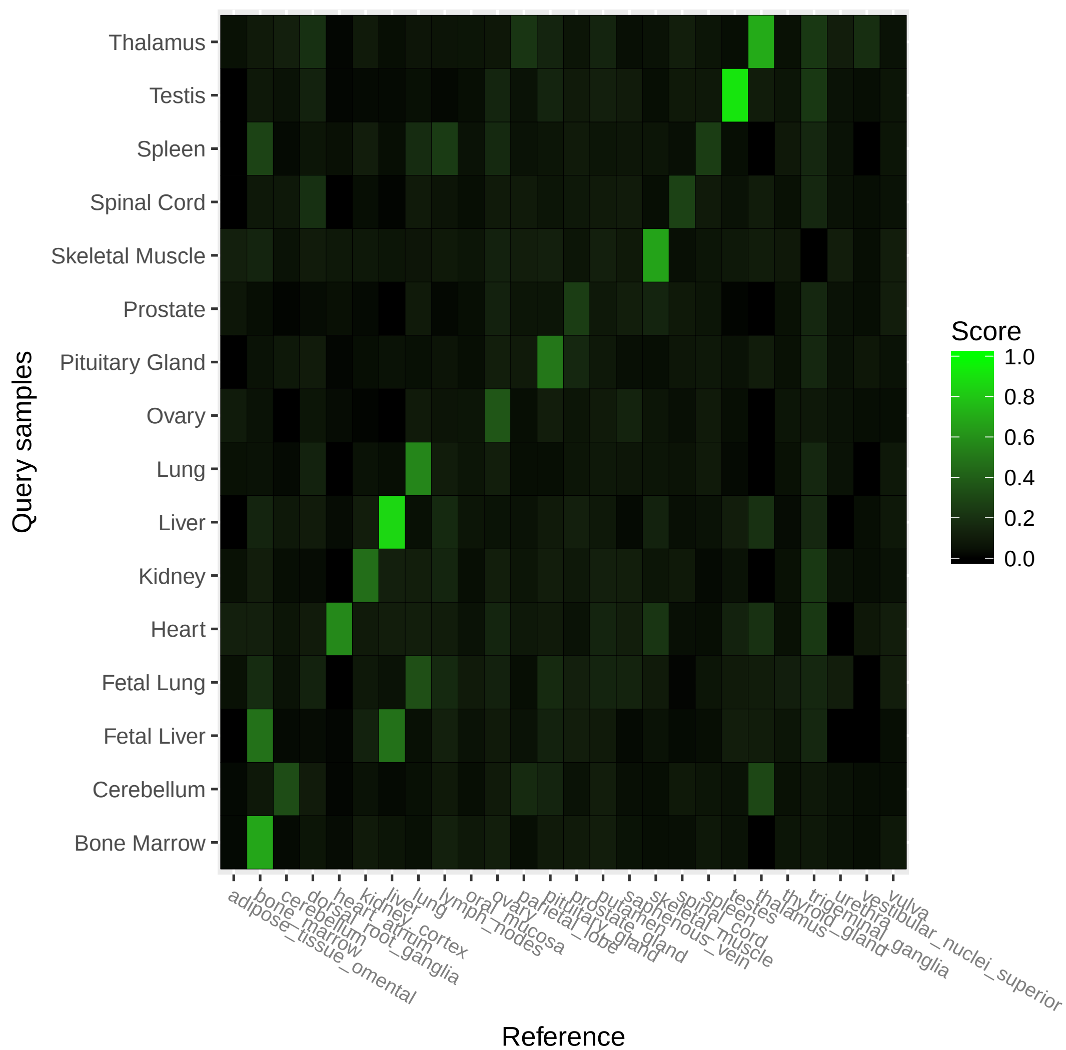
```
1                lung 0.556 25 / 45
2 trigeminal_ganglia 0.154  2 / 13

$`GSM44705 : Fetal Lung`
         Hits Score   Ratio
1         lung 0.333 15 / 45
2 bone_marrow 0.179 14 / 78

$`GSM44706 : Fetal Liver`
         Hits Score   Ratio
1 bone_marrow 0.474 37 / 78
2       liver 0.474 37 / 78
```

To display the classification results as a heatmap, we call the function `get.heatmap()` with the resulted list as input.

```
> get.heatmap(res1.list)
```



In order to compare the classification predictions for microarray query profiles to those of Support Vector Machines (SVM), the function `classifyProfile.svm()` was implemented based on functions from the R-package *e1071*.

```
> res1.svm.df <- classifyProfile.svm(ref_matrix=micro_refmat, query_mat=micro_testmat,
+ chip1="hgu133plus2",chip2="hgu133a")
```

```
The reference matrix and the query are from different platforms...
Collapse rows ...
detecting marker genes...
building an SVM model...
16 profiles to be classified...
done!
```

```
> res1.svm.df
```

```
                  query_name        predicted_class
1            GSM44671 : Heart           heart_atrium
2           GSM44673 : Spleen adipose_tissue_omental
3            GSM44674 : Ovary                  ovary
4           GSM44675 : Kidney          kidney_cortex
5  GSM44676 : Skeletal Muscle        skeletal_muscle
6         GSM44678 : Prostate                urethra
7       GSM44689 : Cerebellum             cerebellum
8      GSM44693 : Bone Marrow            bone_marrow
9         GSM44698 : Thalamus               thalamus
10 GSM44699 : Pituitary Gland        pituitary_gland
11     GSM44700 : Spinal Cord            spinal_cord
12          GSM44701 : Testis                 testes
13           GSM44702 : Liver                  liver
14            GSM44704 : Lung                   lung
15      GSM44705 : Fetal Lung                   lung
16     GSM44706 : Fetal Liver            bone_marrow
```

Our method `classifyProfile()` classified 14 samples correctly, whereas SVM classified 13 of the 16 query samples correctly. The sample 'GSM44678' was classified correctly by `classifyProfile()` as prostate and miclassified by SVM as urethra. The sample 'GSM44673' (tissue: spleen) was misclassified as bone marrow or adipose tissue omental by `classifyProfile()` or `classifyProfile.svm()`, respectively.
The sample 'GSM44706' (tissue: fetal liver) was misclassified as bone marrow by SVM. `classifyProfile()` predicted both bone marrow and liver as top hits with the same similarity score.

4.2 Classification of RNA-seq data

To classify RNA-seq gene expression profiles, we use the function `classifyProfile.rnaseq()`. It expects a reference and a test matrix. As a reference matrix we use an RNA-seq dataset derived from the study E-MTAB-1733 [7], which is available from ArrayExpress [8]. This dataset is named se_rnaseq_refmat and can be loaded with the following code:

```
> library(sampleClassifierData)
> data("se_rnaseq_refmat")
> rnaseq_refmat <- assay(se_rnaseq_refmat)
> dim(rnaseq_refmat)
```

```
[1] 43819    71
```

As a test matrix we use an RNA-seq dataset derived from the study E-MTAB-513 [9], which is available from ArrayExpress. This dataset is named $se\_rnaseq\_testmat$ and can be loaded with the following code:

```
> data("se_rnaseq_testmat")
> rnaseq_testmat <- assay(se_rnaseq_testmat)
> dim(rnaseq_testmat)
```

```
[1] 43819    12
```

Now, we can call the function `classifyProfile.rnaseq()` to predict the classes of the samples in the test matrix:

```
> res2.list <- classifyProfile.rnaseq(ref_matrix=rnaseq_refmat, query_mat=rnaseq_testmat,
+ gene.ids.type="ensembl",write2File=FALSE)
```

```
Detecting marker genes...
Done!
12 profiles to be classified...
Done!
```

For simplicity, we show only the two top hits for each query sample:

```
> lapply(res2.list,"[",c(1,2),,drop=FALSE)
```

```
$adrenal
    Hits Score    Ratio
1 testis 0.141 26 / 184
2 spleen 0.121  12 / 99

$brain
      Hits Score     Ratio
1    brain  0.75 138 / 184
2 appendix 0.143     1 / 7

$colon
         Hits Score  Ratio
1    appendix 0.286  2 / 7
2 endometrium 0.135 5 / 37

$heart
   Hits Score     Ratio
1 heart 0.543 100 / 184
2   fat 0.103  19 / 184

$kidney
    Hits Score    Ratio
1 kidney 0.467 85 / 182
2   lung 0.091   7 / 77

$liver
         Hits Score     Ratio
1       liver 0.652 120 / 184
2 gallbladder 0.174    4 / 23

$lung
        Hits Score    Ratio
1       lung 0.325  25 / 77
2 bonemarrow 0.201 37 / 184

$lymph.node
       Hits Score    Ratio
1 lymphnode 0.196 36 / 184
2  appendix 0.143    1 / 7

$ovary
         Hits Score    Ratio
1       ovary  0.19 35 / 184
2 endometrium 0.108   4 / 37
```

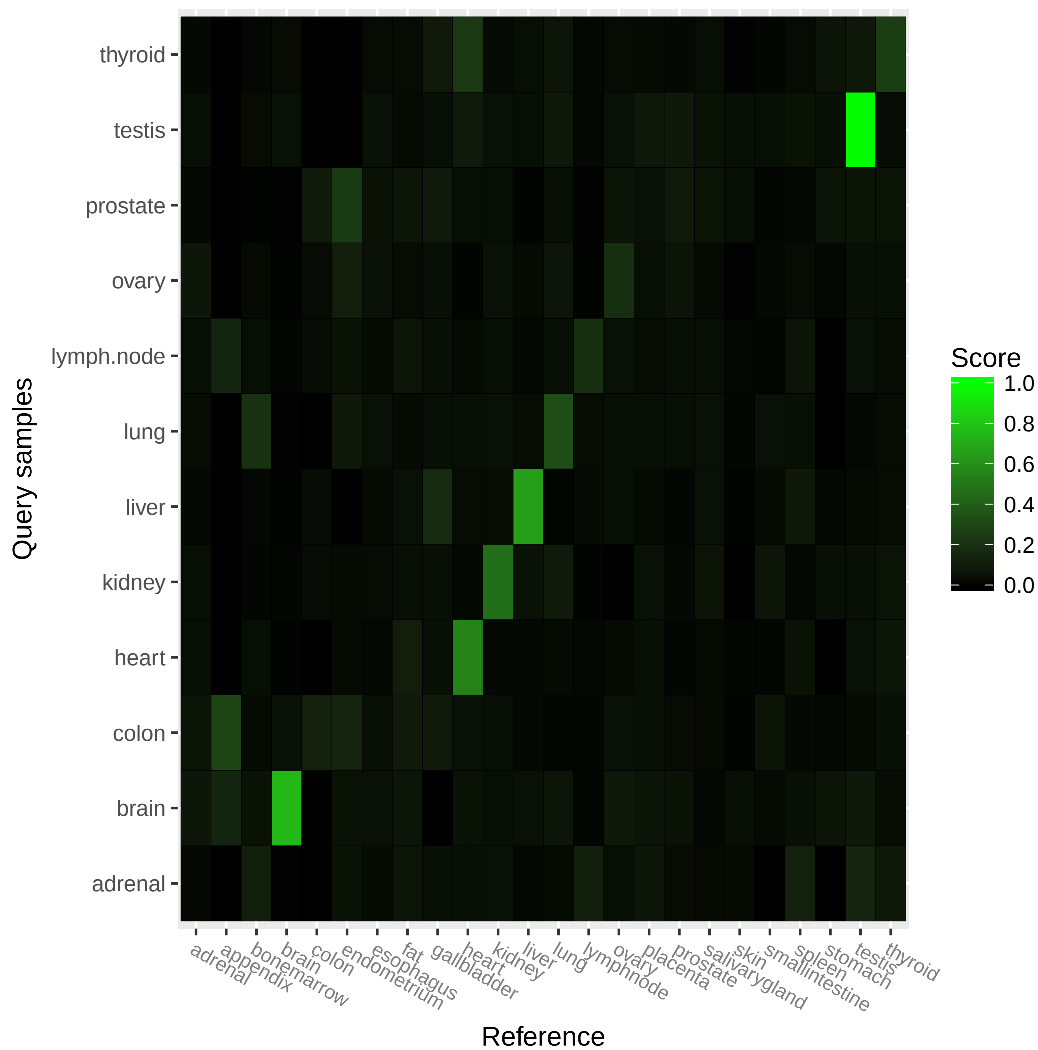
```
$prostate
         Hits Score  Ratio
1 endometrium 0.243 9 / 37
2       colon 0.091 3 / 33

$testis
    Hits Score     Ratio
1 testis 0.989 182 / 184
2  heart 0.092  17 / 184

$thyroid
     Hits Score    Ratio
1 thyroid 0.255 35 / 137
2   heart 0.239 44 / 184
```

To display the classification results as a heatmap, we call the function `get.heatmap()` with the resulted list as input.

```
> get.heatmap(res2.list)
```



In order to compare the classification predictions for RNA-seq query profiles to those of SVM, the function `classi-`

fyProfile.rnaseq.svm() was implemented based on functions from the R-package *e1071*.

```
> res2.svm.df <- classifyProfile.rnaseq.svm(ref_matrix=rnaseq_refmat, query_mat=rnaseq_testmat,
+ gene.ids.type="ensembl")
```

```
Detecting marker genes...
Done!
building an SVM model...
12 profiles to be classified...
done!
```

```
> res2.svm.df
```

```
   query_name predicted_class
1     adrenal        appendix
2       brain           brain
3       colon     gallbladder
4       heart           heart
5      kidney          kidney
6       liver           liver
7        lung            lung
8   lymph.node       appendix
9       ovary     endometrium
10   prostate     endometrium
11     testis          testis
12    thyroid         thyroid
```

Our method classifyProfile.rnaseq() classified 9 samples correctly, whereas SVM classified 7 of the 12 query samples correctly. To show the query samples that were misclassified by our method or SVM, we run the following code:

```
> misclas.inds <- which(as.character(res2.svm.df[,1])!=as.character(res2.svm.df[,2]))
> colnames(res2.svm.df) <- c("query_real_class","predicted_class_by_SVM")
> pred.classifyProfile.rnaseq <- as.character(unlist(lapply(res2.list[which(names(res2.list) %in%
+ as.character(res2.svm.df[misclas.inds,1]))],"[",1,1,drop=TRUE)))
> comp.df <- cbind(res2.svm.df[misclas.inds,],
+ predicted_by_classifyProfile.rnaseq=pred.classifyProfile.rnaseq)
> comp.df
```

```
   query_real_class predicted_class_by_SVM predicted_by_classifyProfile.rnaseq
1           adrenal               appendix                              testis
3             colon            gallbladder                            appendix
8        lymph.node               appendix                           lymphnode
9             ovary            endometrium                               ovary
10         prostate            endometrium                         endometrium
```

5 sampleClassifier algorithm details

The algorithm used in sampleClassifier is a simple algorithm called Shared Marker Genes (SMG). As the name suggests, the number of shared marker genes between a query and a reference is used as similarity measure. The tool requires a reference matrix with at least three replicates for each sample type. This matrix is used for marker gene detection using *MGFM* or *MGFR*. Since the number of detected markers differs depending on the sample types, we filter the list of marker genes of each sample type. Using the complete list of markers of each sample type, will result in a bias towards the sample type with the most marker genes. For example, if testis is the tissue with the most marker genes, using all marker genes for classification will result in classifying query samples often as testis. Suppose the reference matrix contains four tissues: liver, lung, kidney and midbrain, and X=(16, 20, 100, 500) is the vector of lengths of predicted marker genes for these tissues. The filtering is based on the median number of marker genes, in this case median(X) = 60. If the number of predicted markers for a tissue > 60, then only the top 60 marker genes will be used for classification. If the number of predicted markers for a tissue <= 60, then all markers are used for classification. After the filtering step, each query sample will be compared to all sample types in the reference and the number of marker genes shared between the query and each sample type in the reference is calculated. A query shares a marker gene with a reference sample if this marker gene is highly expressed in the query sample compared to all other sample types in the reference. The ratio of the number of shared marker genes and the total number of markers used for classification is used as a similarity score. This score has a value between 0 and 1. A value of 1 means that the query shares all marker genes with the reference, and a value of 0 means that no marker genes are shared between the query and the reference. For each query, the hits are sorted according to this score. The class of the first top hit is predicted as a class for the query.

6 R sessionInfo

The results in this file were generated using the following packages:

```
> sessionInfo()
```

```
R version 3.4.0 (2017-04-21)
Platform: x86_64-apple-darwin15.6.0 (64-bit)
Running under: OS X El Capitan 10.11.6

Matrix products: default
BLAS: /Library/Frameworks/R.framework/Versions/3.4/Resources/lib/libRblas.0.dylib
LAPACK: /Library/Frameworks/R.framework/Versions/3.4/Resources/lib/libRlapack.dylib

locale:
[1] C/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8

attached base packages:
[1] parallel  stats4    stats     graphics  grDevices utils     datasets 
[8] methods   base     

other attached packages:
 [1] hgu133a.db_3.2.3           hgu133plus2.db_3.2.3      
 [3] org.Hs.eg.db_3.4.1         sampleClassifierData_0.99.2
 [5] SummarizedExperiment_1.6.0 DelayedArray_0.2.0        
 [7] matrixStats_0.52.2         GenomicRanges_1.28.0      
 [9] GenomeInfoDb_1.12.0        sampleClassifier_1.0.0    
[11] MGFR_1.2.0                 MGFM_1.10.0               
[13] annotate_1.54.0            XML_3.98-1.6              
[15] AnnotationDbi_1.38.0       IRanges_2.10.0            
[17] S4Vectors_0.14.0           Biobase_2.36.0            
[19] BiocGenerics_0.22.0       

loaded via a namespace (and not attached):
 [1] Rcpp_0.12.10            XVector_0.16.0          compiler_3.4.0         
 [4] plyr_1.8.4              zlibbioc_1.22.0         class_7.3-14           
 [7] bitops_1.0-6            tools_3.4.0             biomaRt_2.32.0         
[10] digest_0.6.12           lattice_0.20-35         RSQLite_1.1-2          
[13] evaluate_0.10           memoise_1.1.0           tibble_1.3.0           
[16] gtable_0.2.0            Matrix_1.2-9            DBI_0.6-1              
[19] yaml_2.1.14             GenomeInfoDbData_0.99.0 e1071_1.6-8            
[22] stringr_1.2.0           knitr_1.15.1            rprojroot_1.2          
[25] grid_3.4.0              rmarkdown_1.4           ggplot2_2.2.1          
```

```
[28] magrittr_1.5          backports_1.0.5        scales_0.4.1
[31] htmltools_0.3.5       BiocStyle_2.4.0        xtable_1.8-2
[34] colorspace_1.3-2      stringi_1.1.5          RCurl_1.95-4.8
[37] lazyeval_0.2.0        munsell_0.4.3
```

References

[1] Khadija El Amrani, Harald Stachelscheid, Fritz Lekschas, Andreas Kurtz, and Miguel A Andrade-Navarro. MGFM: a novel tool for detection of tissue and cell specific marker genes from microarray gene expression data. *BMC genomics*, 16(1):645, jan 2015. URL: http://bmcgenomics.biomedcentral.com/articles/10.1186/s12864-015-1785-9, doi:10.1186/s12864-015-1785-9.

[2] Khadija El Amrani. *MGFM: Marker Gene Finder in Microarray gene expression data*, 2014. R package version 1.8.0.

[3] Khadija El Amrani. *MGFR: Marker Gene Finder in RNA-seq data*, 2016. R package version 1.0.0.

[4] Richard B Roth, Peter Hevezi, Jerry Lee, Dorian Willhite, Sandra M Lechner, Alan C Foster, and Albert Zlotnik. Gene expression analyses reveal molecular relationships among 20 regions of the human CNS. *Neurogenetics*, 7(2):67–80, may 2006. URL: http://www.ncbi.nlm.nih.gov/pubmed/16572319, doi:10.1007/s10048-006-0032-6.

[5] Tanya Barrett, Dennis B Troup, Stephen E Wilhite, Pierre Ledoux, Dmitry Rudnev, Carlos Evangelista, Irene F Kim, Alexandra Soboleva, Maxim Tomashevsky, and Ron Edgar. NCBI GEO: mining tens of millions of expression profiles–database and tools update. *Nucleic acids research*, 35(Database issue):D760–5, jan 2007. URL: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=1669752{&}tool=pmcentrez{&}rendertype=abstract, doi:10.1093/nar/gkl887.

[6] Xijin Ge, Shogo Yamamoto, Shuichi Tsutsumi, Yutaka Midorikawa, Sigeo Ihara, San Ming Wang, and Hiroyuki Aburatani. Interpreting expression profiles of cancers by genome-wide survey of breadth of expression in normal tissues. *Genomics*, 86(2):127–41, aug 2005. URL: http://www.sciencedirect.com/science/article/pii/S0888754305001114, doi:10.1016/j.ygeno.2005.04.008.

[7] Linn Fagerberg, Björn M Hallström, Per Oksvold, Caroline Kampf, Dijana Djureinovic, Jacob Odeberg, Masato Habuka, Simin Tahmasebpoor, Angelika Danielsson, Karolina Edlund, Anna Asplund, Evelina Sjöstedt, Emma Lundberg, Cristina Al-Khalili Szigyarto, Marie Skogs, Jenny Ottosson Takanen, Holger Berling, Hanna Tegel, Jan Mulder, Peter Nilsson, Jochen M Schwenk, Cecilia Lindskog, Frida Danielsson, Adil Mardinoglu, Asa Sivertsson, Kalle von Feilitzen, Mattias Forsberg, Martin Zwahlen, IngMarie Olsson, Sanjay Navani, Mikael Huss, Jens Nielsen, Fredrik Ponten, and Mathias Uhlén. Analysis of the human tissue-specific expression by genome-wide integration of transcriptomics and antibody-based proteomics. *Molecular & cellular proteomics : MCP*, 13(2):397–406, feb 2014. URL: http://www.ncbi.nlm.nih.gov/pubmed/24309898http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC3916642, doi:10.1074/mcp.M113.035600.

[8] Alvis Brazma, Helen Parkinson, Ugis Sarkans, Mohammadreza Shojatalab, Jaak Vilo, Niran Abeygunawardena, Ele Holloway, Misha Kapushesky, Patrick Kemmeren, Gonzalo Garcia Lara, Ahmet Oezcimen, Philippe Rocca-Serra, and Susanna-Assunta Sansone. ArrayExpress–a public repository for microarray gene expression data at the EBI. *Nucleic acids research*, 31(1):68–71, jan 2003. URL: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=165538{&}tool=pmcentrez{&}rendertype=abstract.

[9] The illumina body map 2.0 data. https://www.ebi.ac.uk/arrayexpress/experiments/E-MTAB-513/.

[10] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2007. ISBN 3-900051-07-0. URL: http://www.R-project.org.