Supervised analysis of MS images using Cardinal

Kylie A. Bemis and April Harry

April 26, 2017

Contents

1 Introduction

For experiments in which analyzed samples come from different classes or conditions, a common goal of supervised analysis is to predict the class of a new sample, given a labeled training set for which classes are already known. This task is called classification.

Unlike unsupervised analysis such as clustering, classification requires biological replicates for testing and validation, to avoid biased reporting of accuracy. *Cardinal* implements cross-validation for classification.

In this vignette, an example classification workflow in *Cardinal* is presented, together with plots of the results.

2 Analysis of a renal cell carcinoma (RCC) cancer dataset

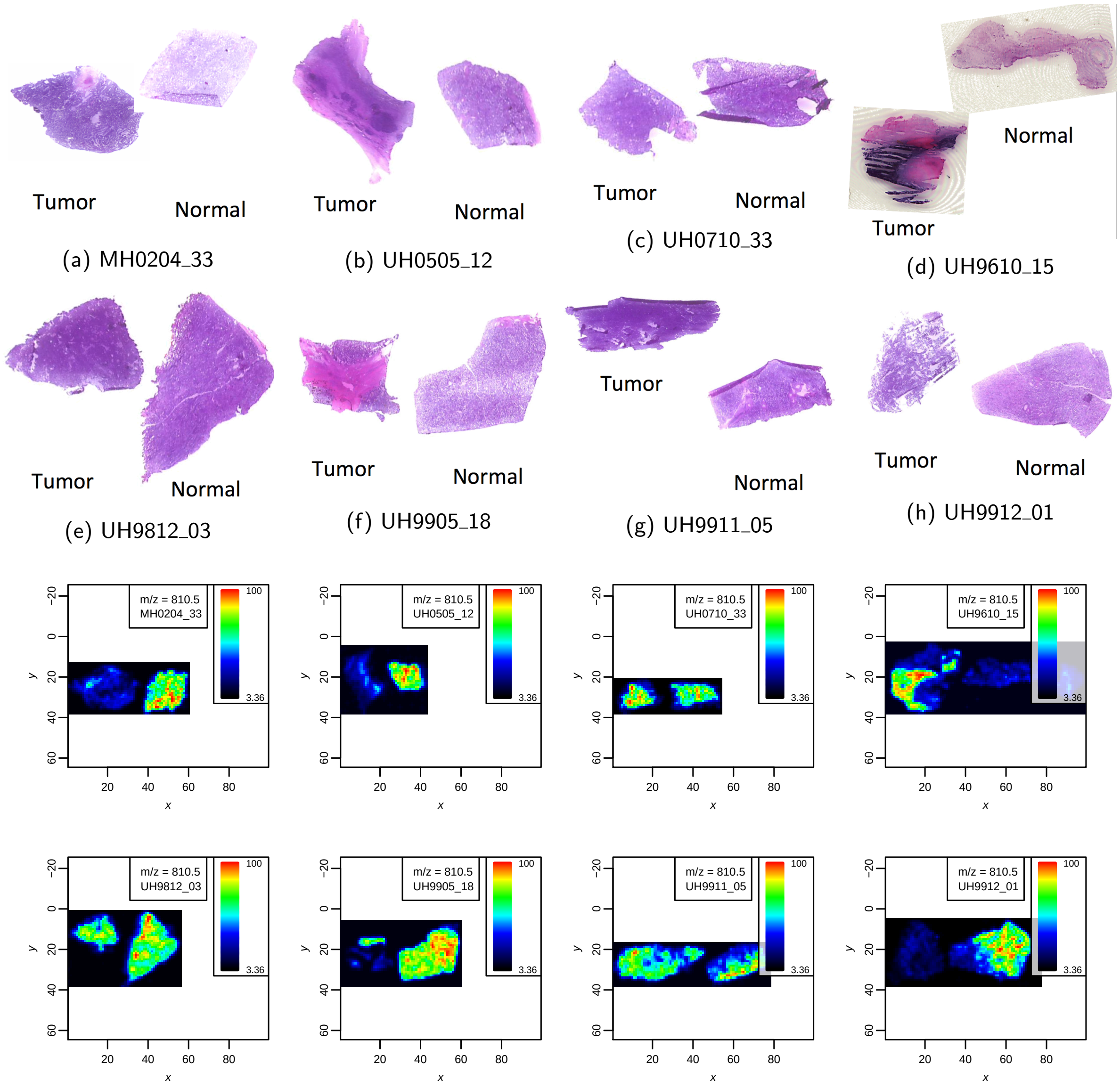
This example uses a renal cell carcinoma (RCC) cancer dataset consisting of 8 matched pairs of human kidney tissue. Each tissue pair consists of a normal tissue sample and a cancerous tissue sample. The goal of the workflow is to develop classifiers for predicting whether a new tissue sample is normal or cancer.

```
> library(CardinalWorkflows)
> data(rcc, rcc_analyses)
```

In this RCC dataset, we expect that normal tissue and cancerous tissue will have unique chemical profiles, which we can use to classify new tissue based on the mass spectra.

In Figure 1 we show the H&E stained tissue samples. In Figure 1i we plot the ion images for m/z 810.5, which we know from previous studies to be abundant in approximately equal intensities in both cancerous and normal tissue [1].

```
> image(rcc, mz = 810.5, normalize.image = "linear", contrast.enhance = "histogram",
+      smooth.image = "gaussian", layout = c(4, 2))
```



(a) MH0204_33

(b) UH0505_12

(c) UH0710_33

(d) UH9610_15

(e) UH9812_03

(f) UH9905_18

(g) UH9911_05

(h) UH9912_01

(i) Corresponding ion images for m/z 810.5

Figure 1: Optical images and ion images for the eight samples showing general morphology.

```
> summary(rcc)
```

```
Class: MSImageSet
Features: m/z = 150.08 ... m/z = 1000 (10200 total)
Pixels: x =  1, y = 13, sample = MH0204_33 ... x = 77, y =  5, sample = UH9912_01 (16000 total)
x: 1 ... 99
y: 1 ... 38
```

```
Size in memory: 629.1 Mb
```

As can be seen in Figure 1, each matched pair of tissues belonging to the same subject are on the same slide. Note also the the cancer tissue is on the left and the normal tissue is on the right on each slide.

The image contains 16000 pixels with 10200 spectral features measured at each location (m/z range from 150 to 1000).

2.1 Pre-processing

For statistical analysis, some form of dimension reduction is necessary so that computation times are reasonable. However, the usual form of dimension reduction for mass spectra – peak-picking – is often unsuitable for classification. This is because classification requires testing and validation to avoid bias in the reported accuracy.

If we perform peak-picking on the whole dataset, then the accuracy reported for the validation set will be biased, because the selected peaks are also coming from the validation set.

Therefore, we recommend resampling or binning as the preferred method of dimension reduction for classification workflows. We will use resampling.

2.1.1 Normalization

Before resampling or binning, normalization is necessary to correct for pixel-to-pixel variation. We will use total ion current (TIC) standardization, which is a popular choice for mass spectrometry imaging datasets.

```
> rcc.norm <- normalize(rcc, method = "tic")
```

2.1.2 Resampling to unit resolution

The normalized data is then resampled to unit resolution. Binning would also be an appropriate alternative, and could be used by setting `method="bin"` in the `reduceDimension` method.

```
> rcc.resample <- reduceDimension(rcc.norm, method = "resample")
```

As discussed above, resampling or binning is preferred to peak-picking for classification. However, if peak-picking is preferred, this can be worked around by performing peak-picking separately on the training set *only*, and using the same peaks in the testing and validation sets. This can become a complex procedure if cross-validation is desired, and will not be covered in this vignette.

2.1.3 Subsetting the dataset

Lastly, we will subset the dataset to drop pixels that contain only the slide background, so that the final dataset will only consist of mass spectra from actual tissue.

To subset the data, we will use the `diagnosis` variable stored in the object's `pixelData`. This variable is a *factor* with the disease condition for each pixel, as annotated by a pathologist.

```
> summary(rcc$diagnosis)
```

```
cancer normal   NA's 
  2775   3302   9923 
```

We drop the 9923 pixels without annotation.

```
> rcc.small <- rcc.resample[, rcc$diagnosis %in% c("cancer", "normal")]
```

```
> summary(rcc.small)
```

```
Class: MSImageSet
Features: m/z = 151 ... m/z = 1000 (850 total)
Pixels: x = 17, y = 15, sample = MH0204_33 ... x = 61, y =  6, sample = UH9912_01 (6077 total)
x: 2 ... 91
```

```
y: 2 ... 37
Size in memory: 41.6 Mb
```

Now the dataset contains only the 6077 mass spectra we need to train and test a classifier.

2.2 Visualizing the dataset

In this section, we will walk through several visualization methods to explore the dataset before training our classifiers.

2.2.1 Visualization of molecular ion images

To begin visualizing the dataset, we will plot ion images for m/z values we already know to be useful in distinguishing normal tissue versus cancer.

First, we plot the ion images for m/z 215.3, known to be more abundant in normal tissue (right) [1], shown in Figure 2a.

```
> image(rcc, mz = 215.3, normalize.image = "linear", contrast.enhance = "histogram",
+     smooth.image = "gaussian", layout = c(4, 2))
```

Likewise, we plot the ion images for m/z 885.7, known to be more abundant in cancerous tissue (left) [1], shown in Figure 2b.

```
> image(rcc, mz = 885.7, normalize.image = "linear", contrast.enhance = "histogram",
+     smooth.image = "gaussian", layout = c(4, 2))
```

From Figure 2a and Figure 2b, we note that there is still a great deal of variation in these images for ions that should be associated with a particular disease condition. For example, m/z 215.3 – which should be more abundant in normal tissue – is also abundant in cancerous tissue for samples UH0505_12 and UH9905_18. This shows that multiple ions will be necessary for classification.

2.2.2 Exploratory analysis using PCA

Although many use principal components analysis (PCA) combined with linear regression for classification, it is a method for unsupervised analysis most appropriately used for exploring a dataset, prior to applying a method designed for classification. We will use PCA for visualization.

Here we fit the first 5 principal components using the PCA method.

```
> rcc.pca <- PCA(rcc.small, ncomp = 5)
```

```
> summary(rcc.pca)
```

```
                        PC1      PC2      PC3      PC4      PC5
Standard deviation 25.02485 17.52704 15.99254 12.49145 9.675649
```

The summary of the first 5 principal components show that PCA is not very useful for this dataset, since the first 5 components cumulatively explain approximately only 51% of the variation in the data.

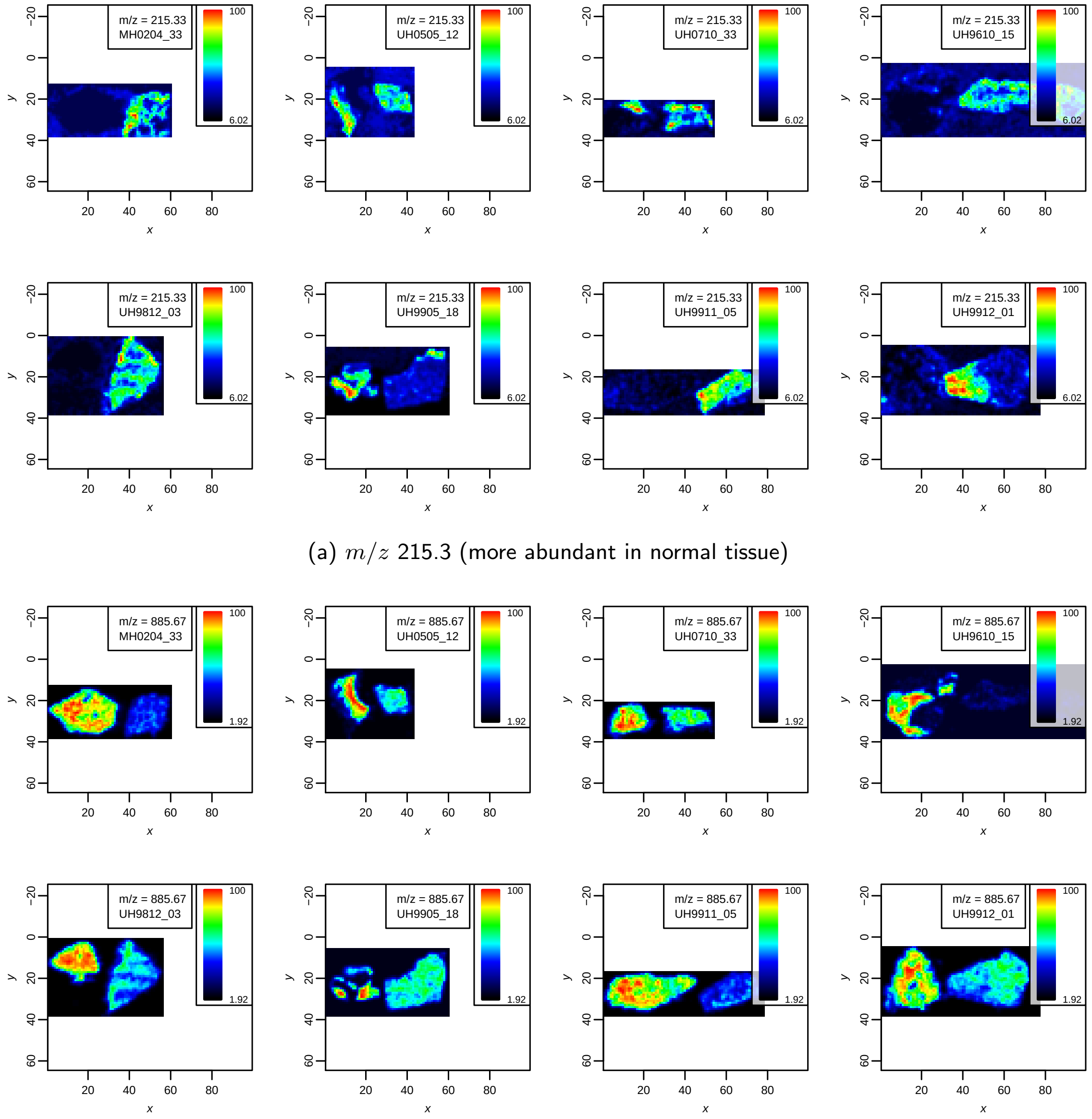
To further explore the dataset with PCA, we plot images of the scores of the first principal component, shown in Figure 3.

```
> image(rcc.pca, column = "PC1", superpose = FALSE, col.regions = risk.colors(100),
+     layout = c(4, 2))
```

Figure 3 show that the images based on the PC1 scores do not seem to show a strong pattern useful for classification of cancer versus normal tissue, although normal tissue seems to have slightly higher PC1 scores.

We also plot the PC loadings for the first 2 principal components, shown in Figure 5.

```
> plot(rcc.pca, column = c("PC1", "PC2", "PC3"), superpose = FALSE, layout = c(3,
+     1))
```



(a) m/z 215.3 (more abundant in normal tissue)

(b) m/z 885.7 (more abundant in cancerous tissue)

Figure 2: Ion images showing ions associated with normal and cancerous tissue.

Another useful PCA plot in a classification setting is to plot the scores of different components against each other, plotting each class separately, which we do below for disease condition.

```
> pca.normal <- as.data.frame(rcc.pca[[1]]$scores[rcc.small$diagnosis == "normal",
+     ])
> pca.cancer <- as.data.frame(rcc.pca[[1]]$scores[rcc.small$diagnosis == "cancer",
+     ])
```

We show PC1 versus PC2 in Figure 5a.

```
> plot(PC2 ~ PC1, data = pca.normal, col = "blue")
> points(PC2 ~ PC1, data = pca.cancer, col = "red")
> legend("top", legend = c("normal", "cancer"), col = c("blue", "red"), pch = 1,
+     bg = rgb(1, 1, 1, 0.75))
```

Now PC1 versus PC3 in Figure 5b.

```
> plot(PC3 ~ PC1, data = pca.normal, col = "blue")
```

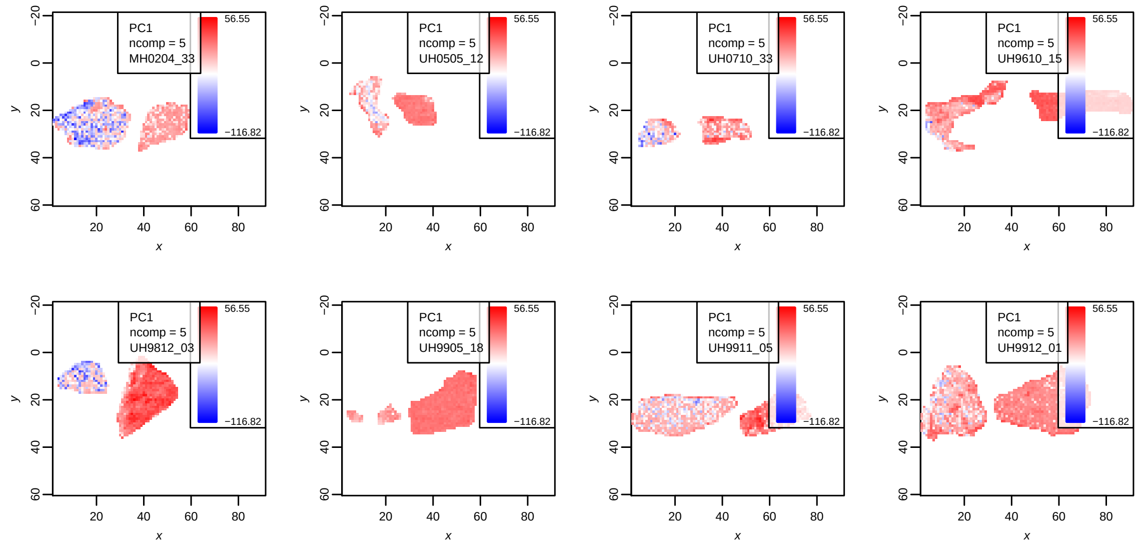


Figure 3: PC scores for the first principal component.

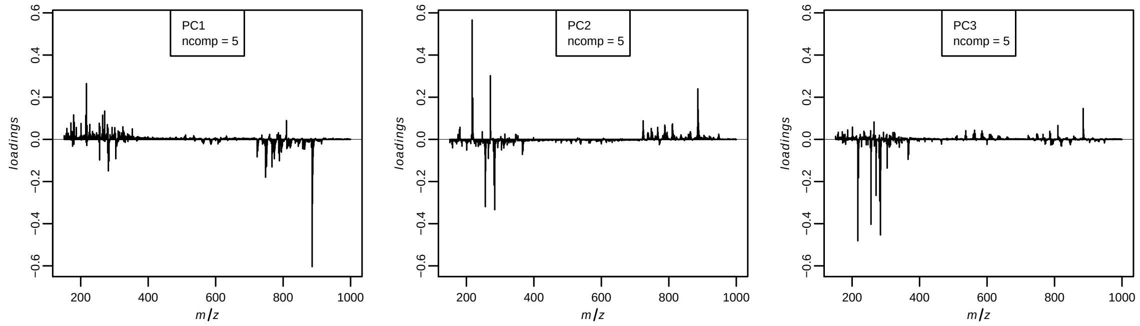


Figure 4: PC loadings for the first three principal components.

```
> points(PC3 ~ PC1, data = pca.cancer, col = "red")
> legend("top", legend = c("normal", "cancer"), col = c("blue", "red"), pch = 1,
+      bg = rgb(1, 1, 1, 0.75))
```

And PC2 versus PC3 in Figure 5c.

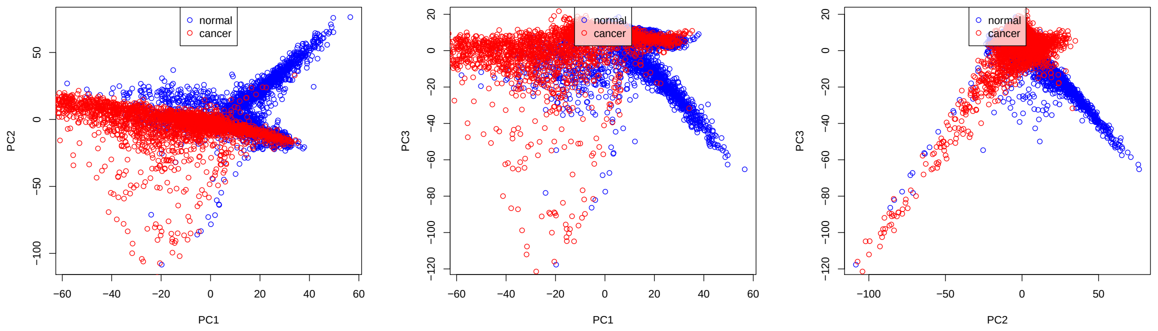
```
> plot(PC3 ~ PC2, data = pca.normal, col = "blue")
> points(PC3 ~ PC2, data = pca.cancer, col = "red")
> legend("top", legend = c("normal", "cancer"), col = c("blue", "red"), pch = 1,
+      bg = rgb(1, 1, 1, 0.75))
```

The PC score plots shown in Figure 5 show that there is indeed separation between the disease conditions in the data. However, it is often difficult to interpret the complex relationship between PC loadings and how the PC scores relate to condition. Therefore, we will now move on to methods designed for classification.

2.3 Classification using PLS-DA

Partial least squares discriminant analysis (PLS-DA) – also known as projection to latent structures – is a multivariate method that has been shown to be useful in the classification of MS images [1]. We will now demonstrate classification of the RCC dataset using PLS-DA.

Note that although we show PLS-DA prediction on two conditions (normal and cancer), it can also be used for classification on more than two conditions.



(a) PC1 versus PC2 scores

(b) PC1 versus PC3 scores

(c) PC2 versus PC3 scores

Figure 5: PCA plots showing PC scores according to disease condition.

2.3.1 Cross-validation with partial least squares

An important step in classification is testing and validation. If the accuracy of a classifier is tested on the same dataset that was used to train the classifier, the reported accuracy will be biased and too optimistic. Therefore, *Cardinal* implements the `cvApply` method, which performs cross-validation for any of the supplied classification methods, including `PLS`.

See `?cvApply` for further details on how to use cross-validation in *Cardinal*.

By default, `cvApply` considers each unique sample (as given by the `sample` variable in an `MSImageSet` object's `pixelData`) as a fold for n-fold cross-validation. In most cases, these should correspond to biological replicates, which is our recommended workflow.

This is the case for the RCC dataset, where each matched pair on a separate slide constitutes a unique sample.

```
> summary(rcc.small$sample)
```

```
MH0204_33 UH0505_12 UH0710_33 UH9610_15 UH9812_03 UH9905_18 UH9911_05 UH9912_01
      811       394       363       801       756       614       937      1401
```

Generally, biological replicates should be used to partition the dataset rather than technical replicates or individual pixels. The only exception would be in the case of a sample size of one, in which case there are no biological replicates. However, a sample size of one is a worst case scenario, and biological replicates should always be preferred.

We now perform cross-validation using PLS-DA as our classification method, using from 1 to 15 PLS components.

```
> rcc.cv.pls <- cvApply(rcc.small, .y = rcc.small$diagnosis, .fun = "PLS", ncomp = 1:15)
```

We plot the cross-validated accuracy to determine the best number of components for prediction, shown in Figure 6.

```
> plot(summary(rcc.cv.pls))
```

As seen in Figure 6, 10 PLS components produce the best prediction rate, with 96.8% cross-validated accuracy.

```
> summary(rcc.cv.pls)$accuracy[["ncomp = 10"]]
```

```
                cancer     normal
Accuracy    0.96813113 0.96813113
Sensitivity 0.93570122 0.97014347
Specificity 0.97014347 0.93570122
FDR         0.02860838 0.02761001
```

Figure 6 tells us that if we wanted to use PLS-DA for prediction on new data, we should train a classifier on this data using 10 PLS components.

Note that `rcc.cv.pls` is a `CrossValidated` object, which contains 8 objects in its `resultData` slot – one for each cross-validation fold – each of which is a `PLS` object containing the results of prediction for that fold.

`CrossValidated` inherits from `ResultSet`. See `?ResultSet` for details.

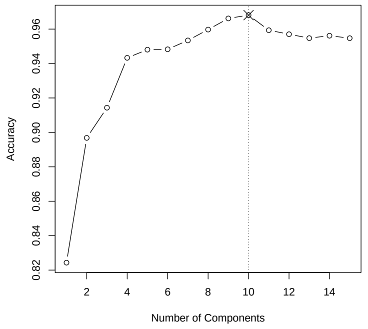


Figure 6: Accuracy of PLS-DA classification for number of components used.

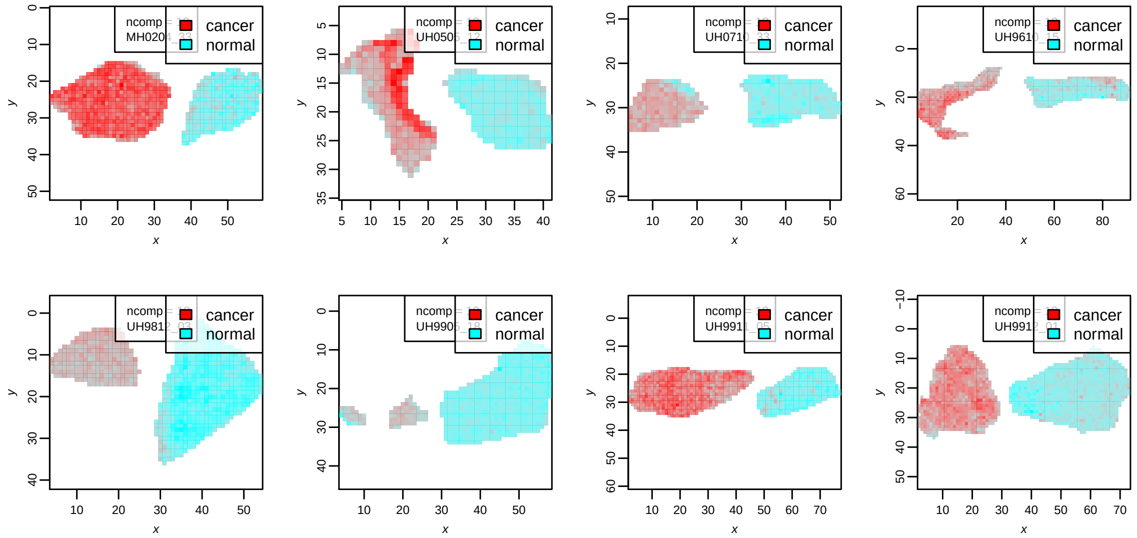


Figure 7: PLS-DA fitted values indicating cancer or normal tissue.

2.3.2 Plotting the classified images

Now we plot the images of the PLS-DA fitted values, to visualize the cross-validated prediction rate, shown in Figure 7.

```
> image(rcc.cv.pls, model = list(ncomp = 10), layout = c(4, 2))
```

For prediction, PLS-DA creates indicator variables (with values 0 or 1) for each condition. The predicted condition is the one with the highest fitted value. (Since the fitted values can fall outside the range 0 to 1, these are not interprettable as probabilities.)

Figure 7 shows the tissues on the left are more predominantly red than blue, indicating that they are predicted to be cancer, which corresponds with the true disease conditions. Since we used cross-validation, the prediction on each matched pair is based only on the data from the other 7 matched pairs of tissue samples.

2.3.3 Plotting and interpretting the coefficients of the m/z values

To interpret the relative importance of the mass features in the classification, we can look at the PLS coefficients used for prediction. To do this, we re-train a PLS classifier on the full dataset using the optical number of PLS components as indicated by the cross-validation.

```
> rcc.pls <- PLS(rcc.small, y = rcc.small$diagnosis, ncomp = 10)
```

Now we plot the PLS coefficients against the m/z values, shown in Figure 8.

```
> plot(rcc.pls)
```

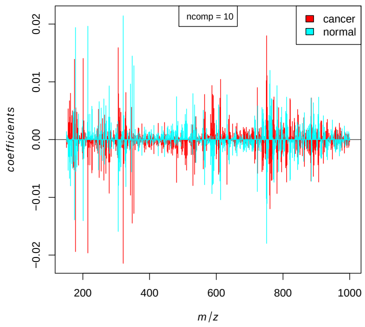


Figure 8: PLS coefficients for cancer and normal.

We can also rank the most important for each condition, based on the PLS coefficients, by using the `topLabels` method.

```
> topLabels(rcc.pls)
```

```
   mz ncomp column coefficients     loadings       weights
1 321    10 normal   0.02143306 -0.043831831 -6.541237e-05
2 215    10 normal   0.01963408  0.037844835  1.357615e-01
3 178    10 normal   0.01940548  0.032555267  3.789179e-02
4 751    10 cancer   0.01798359 -0.126079063 -9.331105e-02
5 306    10 cancer   0.01591385  0.006784054 -5.327659e-02
6 348    10 normal   0.01447231 -0.007467727 -1.009812e-01
```

Among the top-ranked m/z values, we see m/z 215 is listed for normal tissue, which we know to be abundant in normal tissue. The top-ranked ion for cancer is m/z 751, which we plot below in Figure 9.

```
> image(rcc.small, mz = 751, layout = c(4, 2), normalize.image = "linear", contrast.enhance = "histogra
+       smooth.image = "gaussian")
```

We see from Figure 9 that m/z 751 is indeed more abundant in the cancer tissue. Although the PLS coefficients are useful for ranking the relative important of m/z values, they are not indicative of statistical significance.

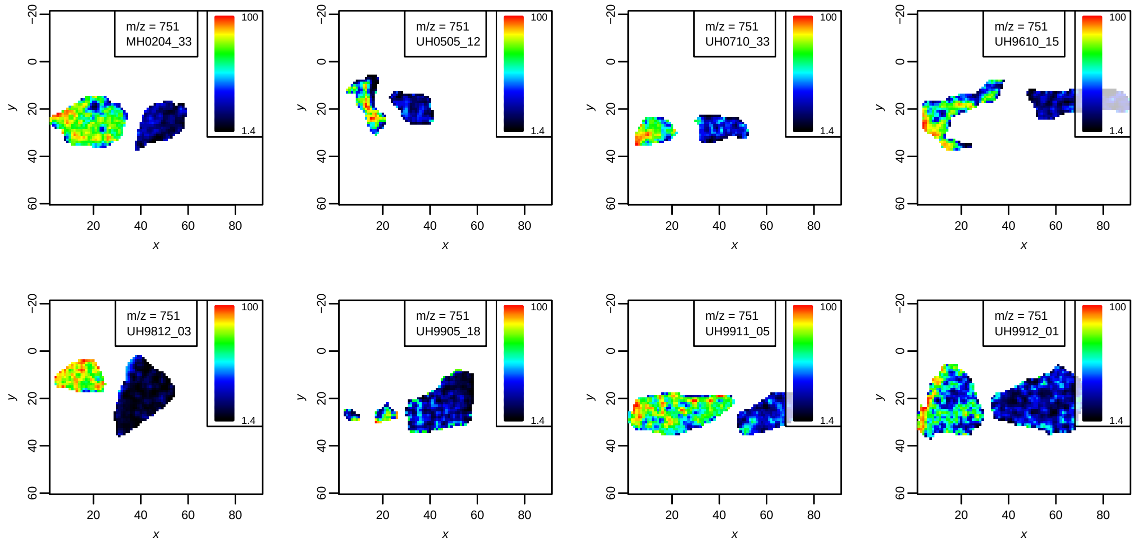


Figure 9: $m/$ 751 (identified by PLS as associated with cancer)

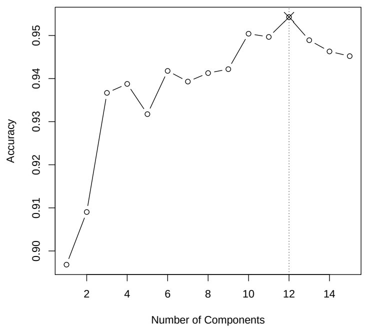


Figure 10: Accuracy of O-PLS-DA classification for number of components used.

2.4 Classification using O-PLS-DA

Orthogonal partial least squares discriminant analysis (O-PLS-DA) is another multivariate method that can be useful for classification of MS images. It is related to PLS, but it removes from the data a number of PLS components orthogonal to the relationship between the data and condition prior to fitting a 1-component PLS model.

O-PLS-DA can often produce comparable accuracy to PLS-DA, but with more stable and easily interprettable coefficients.

2.4.1 Cross-validation with partial least squares

We now use cross-validation to fit O-PLS-DA models for the RCC dataset.

```
> rcc.cv.opls <- cvApply(rcc.small, .y = rcc.small$diagnosis, .fun = "OPLS", ncomp = 1:15,
+     keep.Xnew = FALSE)
```

```
> plot(summary(rcc.cv.opls))
```

As seen in Figure 10, 12 O-PLS components produce the best prediction rate, with 95.7% cross-validated accuracy.

```
> summary(rcc.cv.pls)$accuracy[["ncomp = 12"]]
```

```
                cancer     normal
Accuracy    0.95703555 0.95703555
Sensitivity 0.94163712 0.95103430
Specificity 0.95103430 0.94163712
FDR         0.04267612 0.03080051
```

2.4.2 Plotting the classified images

As with PLS-DA, we now plot the images for the O-PLS-DA fitted values, to visually show the predictions, shown in Figure 11.

```
> image(rcc.cv.opls, model = list(ncomp = 12), layout = c(4, 2))
```

Figure 11 shows good quality prediction comparable with PLS-DA.

2.4.3 Plotting and interpretting the coefficients of the m/z values

Now we consider the O-PLS coefficients by training a classifier on the full dataset with the optimal number of O-PLS components as shown by cross-validation.

```
> rcc.opls <- OPLS(rcc.small, y = rcc.small$diagnosis, ncomp = 12, keep.Xnew = FALSE)
```

And we plot the O-PLS coefficients, as shown in Figure 12.

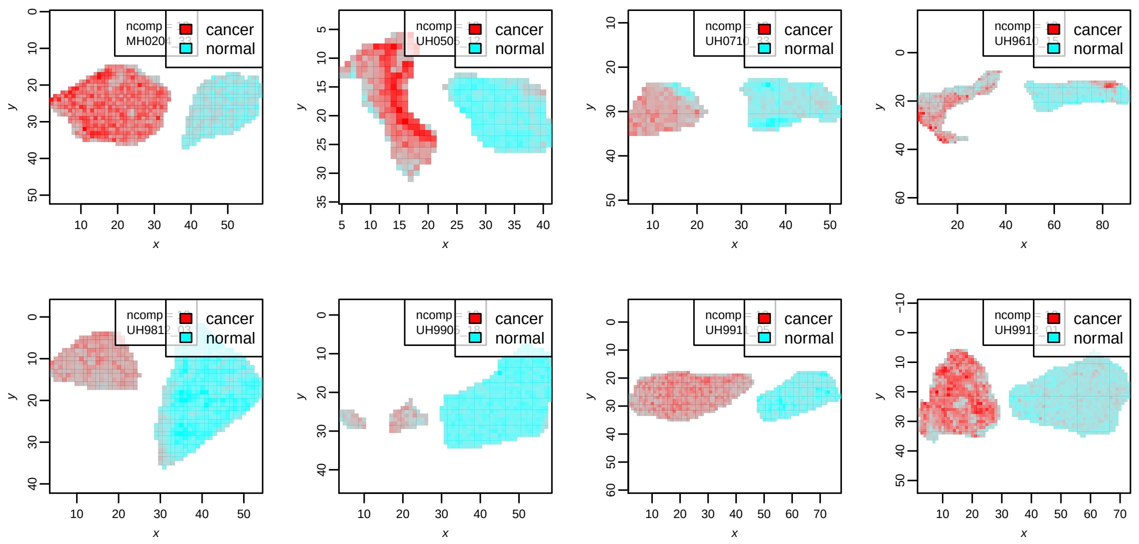


Figure 11: O-PLS-DA fitted values indicating cancer or normal tissue.

```
> plot(rcc.opls)
```

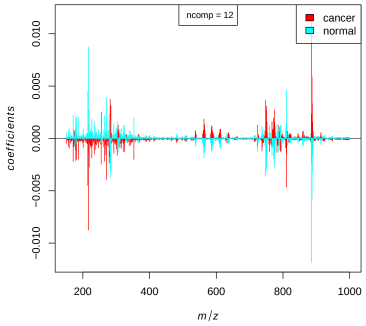


Figure 12: O-PLS coefficients for cancer and normal.

A comparison of the O-PLS coefficients in Figure 12 with the PLS coefficients from Figure 8 shows that the O-PLS coefficients appear more stable and should be easier to interpret.

We get the top-ranked m/z values using `topLabels`

```
> topLabels(rcc.opls)
```

```
   mz ncomp column coefficients   loadings   Oloadings    weights    Oweights
1 886    12 cancer  0.011823832 -0.4728933 -0.10096867 -0.4974430 -0.12552027
2 217    12 normal  0.008737070  0.3864361 -0.22423627  0.3675792 -0.08574331
3 887    12 cancer  0.005885642 -0.2285344  0.05847600 -0.2476161  0.16863198
4 810    12 normal  0.004634575  0.1972194 -0.01996135  0.1949822  0.10523709
5 215    12 normal  0.004515749  0.1800453 -0.15026463  0.1899831  0.05293926
6 271    12 normal  0.003926693  0.1788175 -0.24348253  0.1652008 -0.14747958
```

The O-PLS coefficients rank m/z 886 highly for cancer, which we know to be more abundant in the cancerous tissue. As with the PLS coefficients, the ion at m/z 215, which we know to be more abundant in normal tissue, is also highly ranked for normal.

2.5 Classification using spatial shrunken centroids

This section demonstrates the spatial shrunken centroids classification method for statistical analysis we introduce in *Cardinal* in the `spatialShrunkenCentroids` method.

In this method, we adapt the nearest shrunken centroids classifier [2] with spatial smoothing. This method uses statistical regularization to shrink each condition's mean spectrum toward the global mean spectrum. This shrinkage allows automated feature selection of important masses. It then classifies pixels by comparing their mass spectra to the shrunken mean spectra of each conditions. The spatial smoothing uses weights adapted from spatially-aware clustering [3], including Gaussian weights, and adaptive weights that attempt to account for local structure.

The parameters to be explicitly provided in the `spatialShrunkenCentroids` method are:

- r: The neighborhood smoothing radius
- s: The shrinkage parameter

The s parameter is the shrinkage parameter that enforces sparsity. As s increases, fewer mass features (m/z values) will be used by the classifier, and only the informative mass features will be retained.

For a detailed explanation of the shrinkage parameter s, see [2] and [4].

Clustering can also be performed if no response variable y is given, by providing an additional parameter k for the initial number of clusters. See the clustering workflow for details.

2.5.1 Cross-validation with spatial shrunken centroids

Now we perform cross-validation with spatial shrunken centroids classification and the `method="gaussian"` weights.

```
> rcc.cv.sscg <- cvApply(rcc.small, .y = rcc.small$diagnosis, .fun = "spatialShrunkenCentroids",
+     method = "gaussian", r = c(1, 2, 3), s = c(0, 4, 8, 12, 16, 20, 24, 28))
```

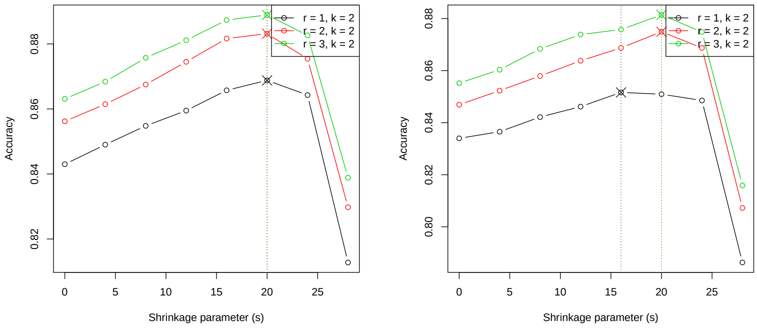
And we perform cross-validation with spatial shrunken centroids classification and the `method="adaptive"` weights.

```
> rcc.cv.ssca <- cvApply(rcc.small, .y = rcc.small$diagnosis, .fun = "spatialShrunkenCentroids",
+     method = "adaptive", r = c(1, 2, 3), s = c(0, 4, 8, 12, 16, 20, 24, 28))
```

Now we plot the cross-validated accuracy for the classifier with Gaussian weights in Figure 13a and adaptive weights in Figure 13b.

```
> plot(summary(rcc.cv.sscg))
```

```
> plot(summary(rcc.cv.ssca))
```



(a) Accuracy for Gaussian weights (b) Accuracy for adaptive weights

Figure 13: Plots of accuracy for spatial shrunken centroids, showing highest accuracy for $s = 20$ and $r = 3$.

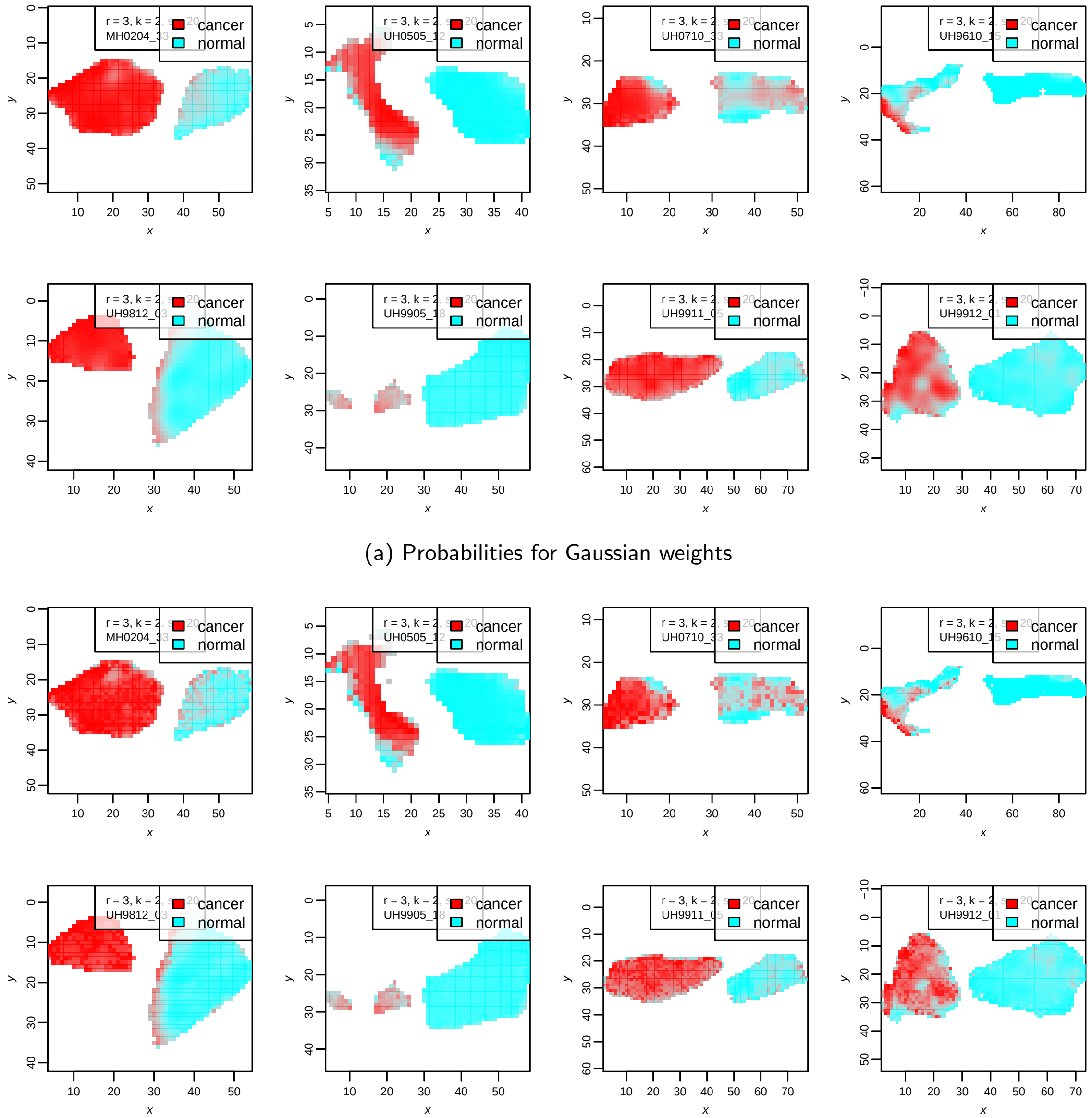
As shown in Figure 13, for both weight types and all smoothing radii r, the highest accuracy occurs with a shrinkage parameter $s = 20$, except for the case with adaptive weights and $r = 1$, for which the highest accuracy occurs at $s = 16$. For Gaussian weights with $r = 3, s = 20$, accuracy was 88.8%.

Note that in general, the accuracy increases with larger smoothing neighborhood radii r. This is likely because rather than heterogenous samples with both normal and cancerous cells on the same tissue, each tissue is relatively homogenous with predominantly normal or cancerous cells. Therefore, greater spatial smoothing increases the accuracy, and adaptive weights have no advantage over Gaussian weights. For classification on more heterogenous tissue, adaptive weights may perform better.

2.5.2 Plotting the classified images

Now we plot the classified images for Gaussian weights in Figure 14a and adaptive weights in Figure 14b.

```
> image(rcc.cv.sscg, model = list(r = 3, s = 20), layout = c(4, 2))
> image(rcc.cv.ssca, model = list(r = 3, s = 20), layout = c(4, 2))
```



(a) Probabilities for Gaussian weights

(b) Probabilities for adaptive weights

Figure 14: Predicted probabilities of cancer and normal, with higher opacity for a condition's color indicating higher probability.

Unlike PLS-DA and O-PLS-DA, spatial shrunken centroids produce probabilities of cancer versus normal, which we plot using higher opacity for higher probability. This makes for more interpretable predicted images.

2.5.3 Plotting and interpreting the t-statistics of the m/z values

A major advantage of spatial shrunken centroids is that it provides t-statistics for each feature for each condition, and it uses statistical regularization to perform automatic feature selection. This allows for easier identification of important m/z values, and a more straightforward and interpretable method of ranking their relative importance.

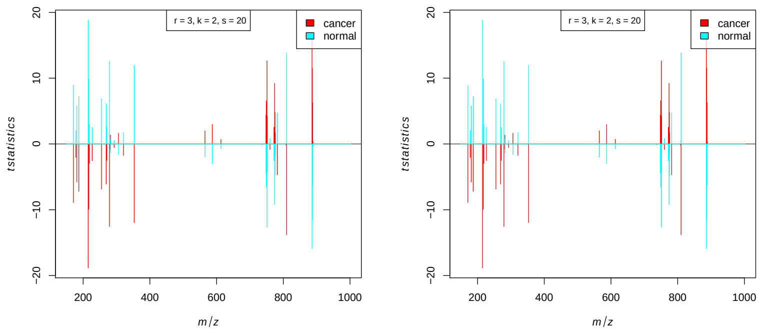
To inspect the t-statistics of the m/z values, we now train classifiers on the full dataset using the parameters $r = 3, s = 20$.

```
> rcc.sscg <- spatialShrunkenCentroids(rcc.small, y = rcc.small$diagnosis, r = 3,
+     s = 20, method = "gaussian")
> rcc.ssca <- spatialShrunkenCentroids(rcc.small, y = rcc.small$diagnosis, r = 3,
+     s = 20, method = "adaptive")
```

Now we plot the t-statistics, shown for Gaussian weights in Figure 15a and for adaptive weights in Figure 15b.

```
> plot(rcc.sscg, mode = "tstatistics", model = list(r = 3, s = 20))
```

```
> plot(rcc.ssca, mode = "tstatistics", model = list(r = 3, s = 20))
```



(a) Shrunken t-statistics for Gaussian weights

(b) Shrunken t-statistics for adaptive weights

Figure 15: Predicted probabilities of cancer and normal, with higher opacity for a condition's color indicating higher probability.

As seen in Figure 15a and Figure 15b, only a few m/z values have non-zero t-statistics.

```
> summary(rcc.sscg)
```

```
  r k  s   method  time Predicted # of Classes Mean # of Features per Class
1 3 2 20 gaussian 3.128                     2                            40
```

```
> summary(rcc.ssca)
```

```
  r k  s   method  time Predicted # of Classes Mean # of Features per Class
1 3 2 20 adaptive 3.233                     2                            40
```

In fact, only 40 of 850 mass features are used in the spatial shrunken centroids classifier.

We identify the top-ranked mass features using the `topLabels` method.

```
> topLabels(rcc.sscg)
```

```
   mz r k  s classes   centers tstatistics p.values adj.p.values
1 215 3 2 20  normal  5.955134    18.83852        0            0
2 886 3 2 20  cancer 19.711863    15.90639        0            0
3 810 3 2 20  normal  8.640044    13.79732        0            0
4 751 3 2 20  cancer  4.030214    12.62721        0            0
5 279 3 2 20  normal  3.596872    12.54607        0            0
6 353 3 2 20  normal  2.261867    11.95248        0            0
```

```
> topLabels(rcc.ssca)
```

```
   mz r k  s classes   centers tstatistics p.values adj.p.values
1 215 3 2 20  normal  5.955134    18.83852        0            0
2 886 3 2 20  cancer 19.711863    15.90639        0            0
3 810 3 2 20  normal  8.640044    13.79732        0            0
4 751 3 2 20  cancer  4.030214    12.62721        0            0
5 279 3 2 20  normal  3.596872    12.54607        0            0
6 353 3 2 20  normal  2.261867    11.95248        0            0
```

Note that the shrunken t-statistics are identical between the Gaussian and adaptive weights, since they are based on the same training data, and the spatial structure is taken into account only for the predicted probabilities.

Spatial shrunken centroids identified m/z 215, m/z 886, and m/z 810, which were also identified by O-PLS-DA, and are all known to be important (as discussed in Section 2.2.1), as well as m/z 751, which was identified by PLS-DA. Of the three methods, spatial shrunken centroids most reliably selected the m/z values known to be associated with cancer and normal, in addition to selecting potentially informative new m/z values.

3 Session info

- R version 3.4.0 (2017-04-21), x86_64-pc-linux-gnu
- Locale: LC_CTYPE=en_US.UTF-8, LC_NUMERIC=C, LC_TIME=en_US.UTF-8, LC_COLLATE=C, LC_MONETARY=en_US.UTF-8, LC_MESSAGES=en_US.UTF-8, LC_PAPER=en_US.UTF-8, LC_NAME=C, LC_ADDRESS=C, LC_TELEPHONE=C, LC_MEASUREMENT=en_US.UTF-8, LC_IDENTIFICATION=C
- Running under: Ubuntu 16.04.2 LTS
- Matrix products: default
- BLAS: /home/biocbuild/bbs-3.5-bioc/R/lib/libRblas.so
- LAPACK: /home/biocbuild/bbs-3.5-bioc/R/lib/libRlapack.so
- Base packages: base, datasets, grDevices, graphics, methods, parallel, stats, utils
- Other packages: Biobase 2.36.0, BiocGenerics 0.22.0, Cardinal 1.7.2, CardinalWorkflows 1.8.0, DBI 0.6-1, ProtGenerics 1.8.0, biglm 0.9-1, matter 1.2.0
- Loaded via a namespace (and not attached): BiocStyle 2.4.0, MASS 7.3-47, Matrix 1.2-9, Rcpp 0.12.10, backports 1.0.5, compiler 3.4.0, digest 0.6.12, evaluate 0.10, grid 3.4.0, htmltools 0.3.5, irlba 2.1.2, knitr 1.15.1, lattice 0.20-35, magrittr 1.5, rmarkdown 1.4, rprojroot 1.2, signal 0.7-6, sp 1.2-4, stats4 3.4.0, stringi 1.1.5, stringr 1.2.0, tools 3.4.0, yaml 2.1.14

References

[1] A. L. Dill, L. S. Eberlin, C. Zheng, A. B. Costa, D. R. Ifa, L. Cheng, T. A. Masterson, M. O. Koch, O. Vitek, and R. G. Cooks. Multivariate statistical differentiation of renal cell carcinomas based on lipidomic analysis by ambient ionization imaging mass spectrometry. *Analytical and Bioanalytical Chemistry*, 398:2969, 2010.

[2] Robert Tibshirani, Trevor Hastie, Balasubramanian Narasimhan, and Gilbert Chu. Diagnosis of multiple cancer types by shrunken centroids of gene expression. *Proceedings of the National Academy of Sciences of the United States of America*, 99(10):6567–6572, 2002.

[3] Theodore Alexandrov and Jan Hendrik Kobarg. Efficient spatial segmentation of large imaging mass spectrometry datasets with spatially aware clustering. *Bioinformatics (Oxford, England)*, 27(13):i230–8, July 2011. URL: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3117346&tool=pmcentrez&rendertype=abstract, doi:10.1093/bioinformatics/btr246.

[4] Robert Tibshirani, Trevor Hastie, Balasubramanian Narasimhan, and Gilbert Chu. Class Prediction by Nearest Shrunken with Applications to DNA Microarrays. *Statistical Science*, 18(1):104–117, 2003.