Session 2: Cross-validation

Prof Jean Yang, Dr Dario Strbenac, Dr Ellis Patrick, Dr Shila Ghazanfar

29 June 2018

THE UNIVERSITY OF SYDNEY

Roadmap

- Part 1: Introduction to statistical machine learning
- Using R code to build classification models with RNA-seq or microarray data and basic performance assessment: 90 minutes.
- Afternoon tea: 30 minutes.
- Part 2: Performance assessment with cross-validation
- Understanding the ClassifyR package and using cross-validation to assess an existing classifier: 80 minutes.
- Final wrap up - overview of the latest methods on biologically guided machine learning approaches: 10 minutes.

Performance Assessment

- Any *classification rule* needs to be *evaluated* for its performance on the future samples. It is almost never the case in microarray studies that a large independent population-based collection of samples is available at the time of initial classifier-building phase.

- One needs to estimate future performance based on what is available: often the same set that is used to build the classifier.

- Assessing performance of the classifier based on

    - Cross-validation.
    - Test set.
    - Independent testing on future dataset.
    - Independent testing on existing dataset (integrative analysis).

Diagram of performance assessment

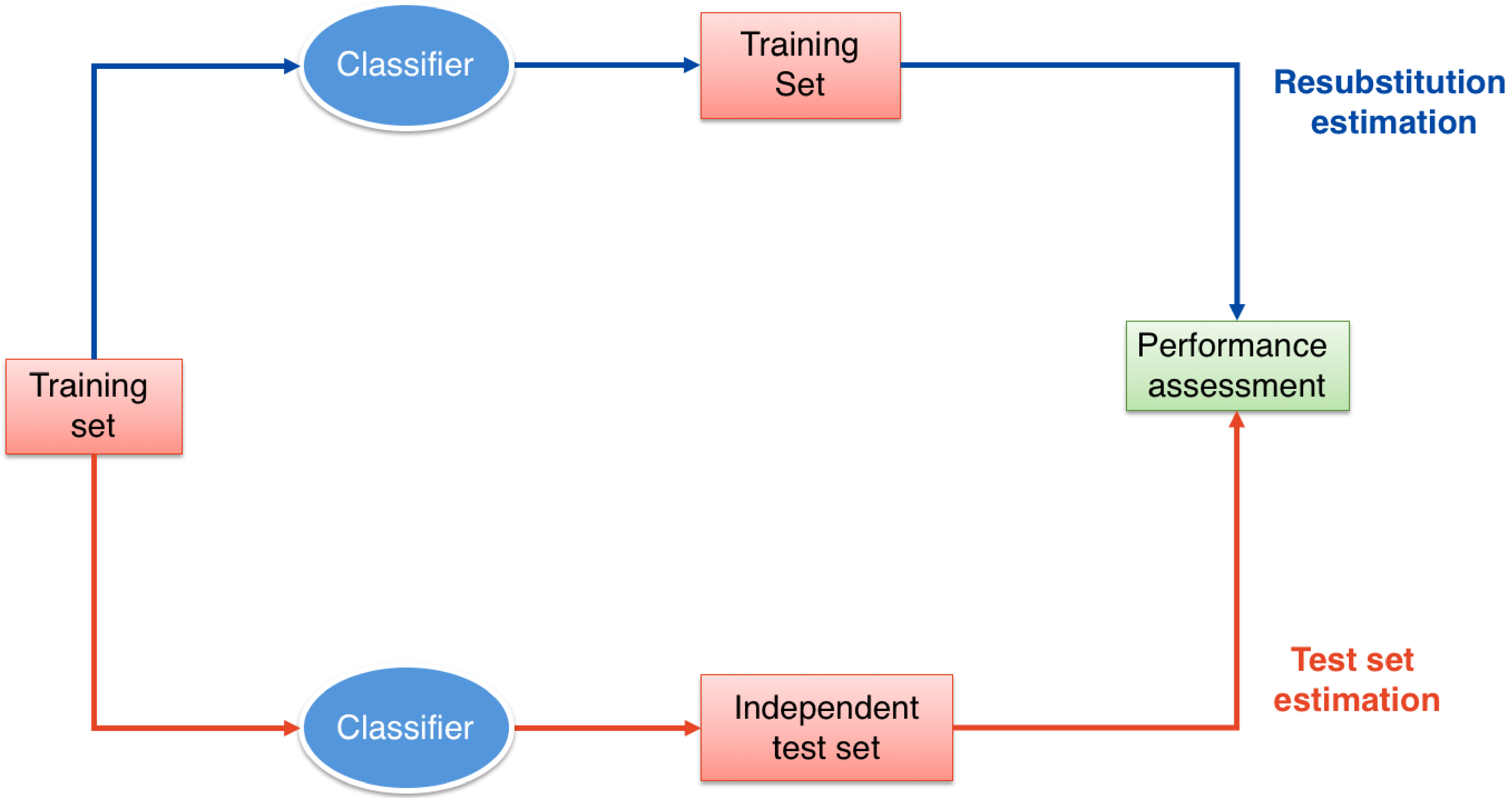
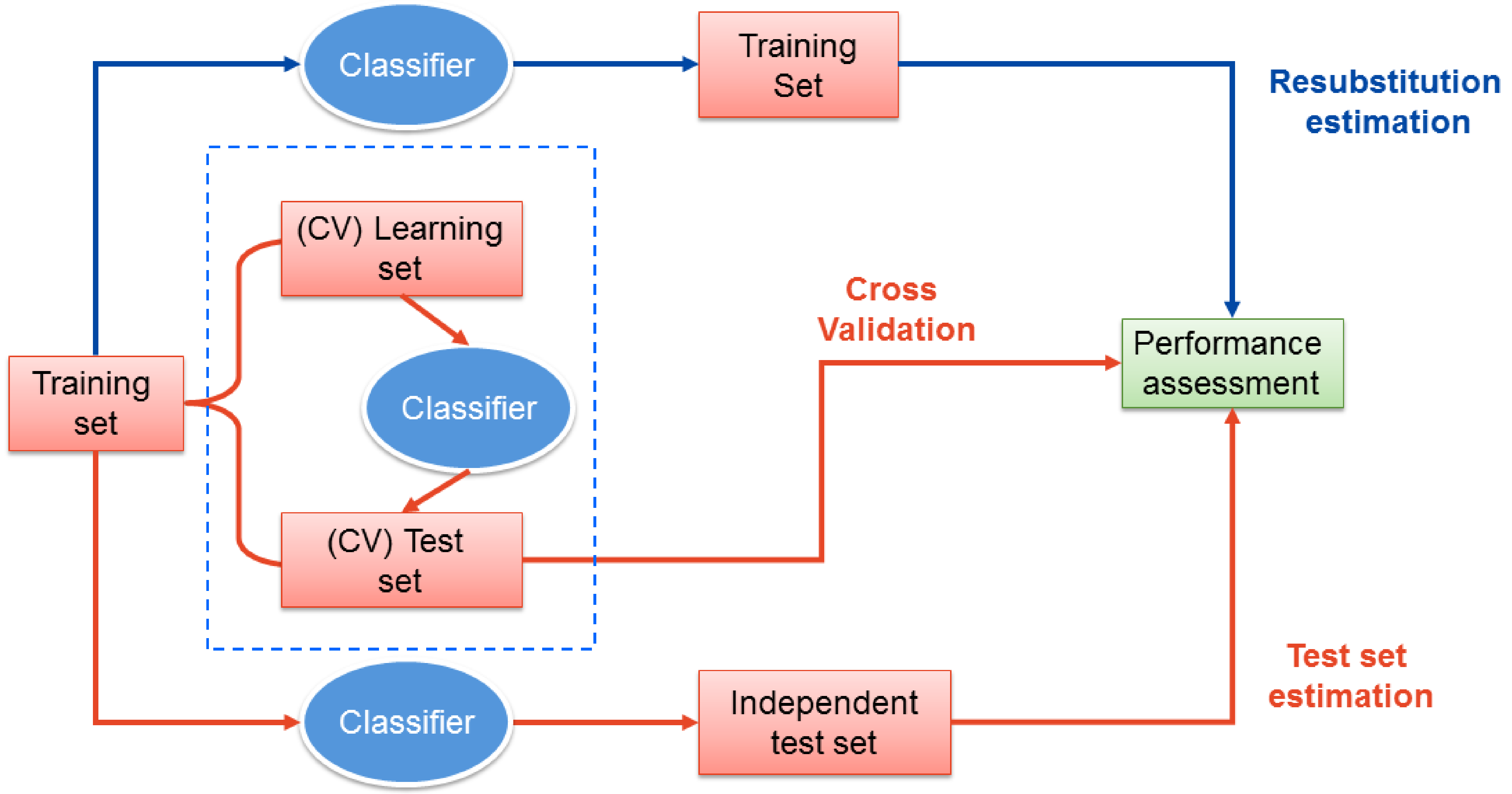
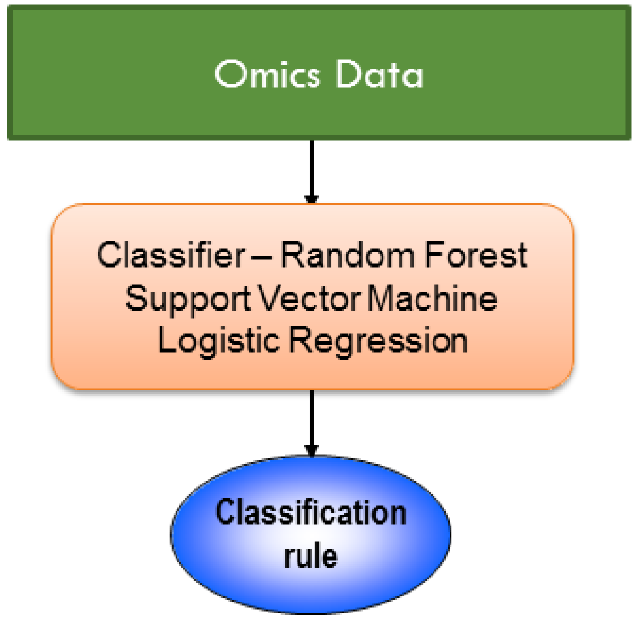
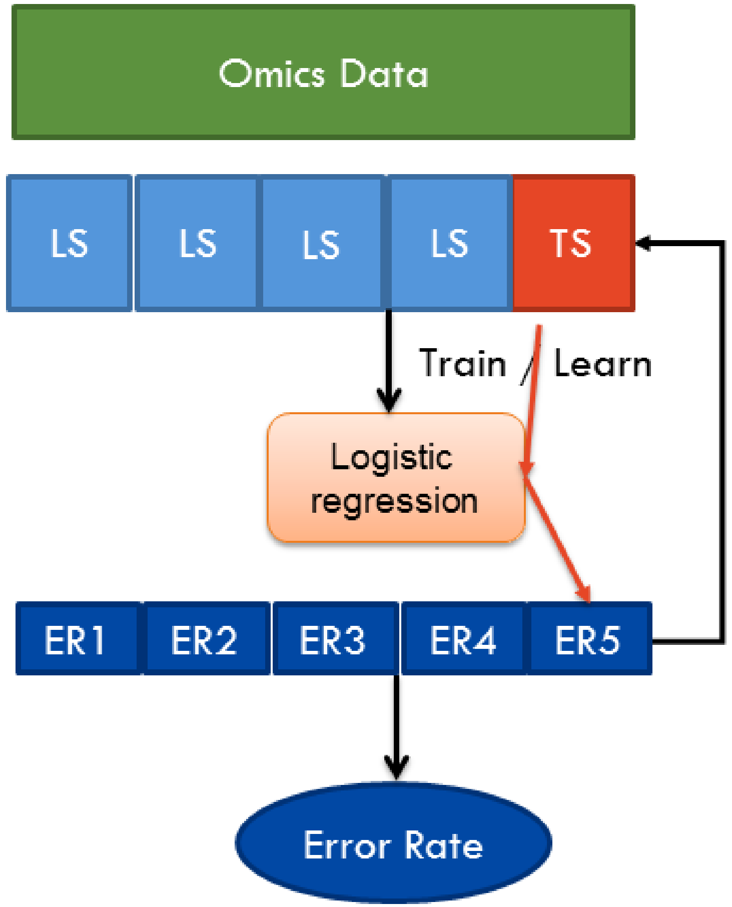


Diagram of performance assessment



5-fold CV


Omics Data
Classifier – Random Forest
Support Vector Machine
Logistic Regression
Classification rule

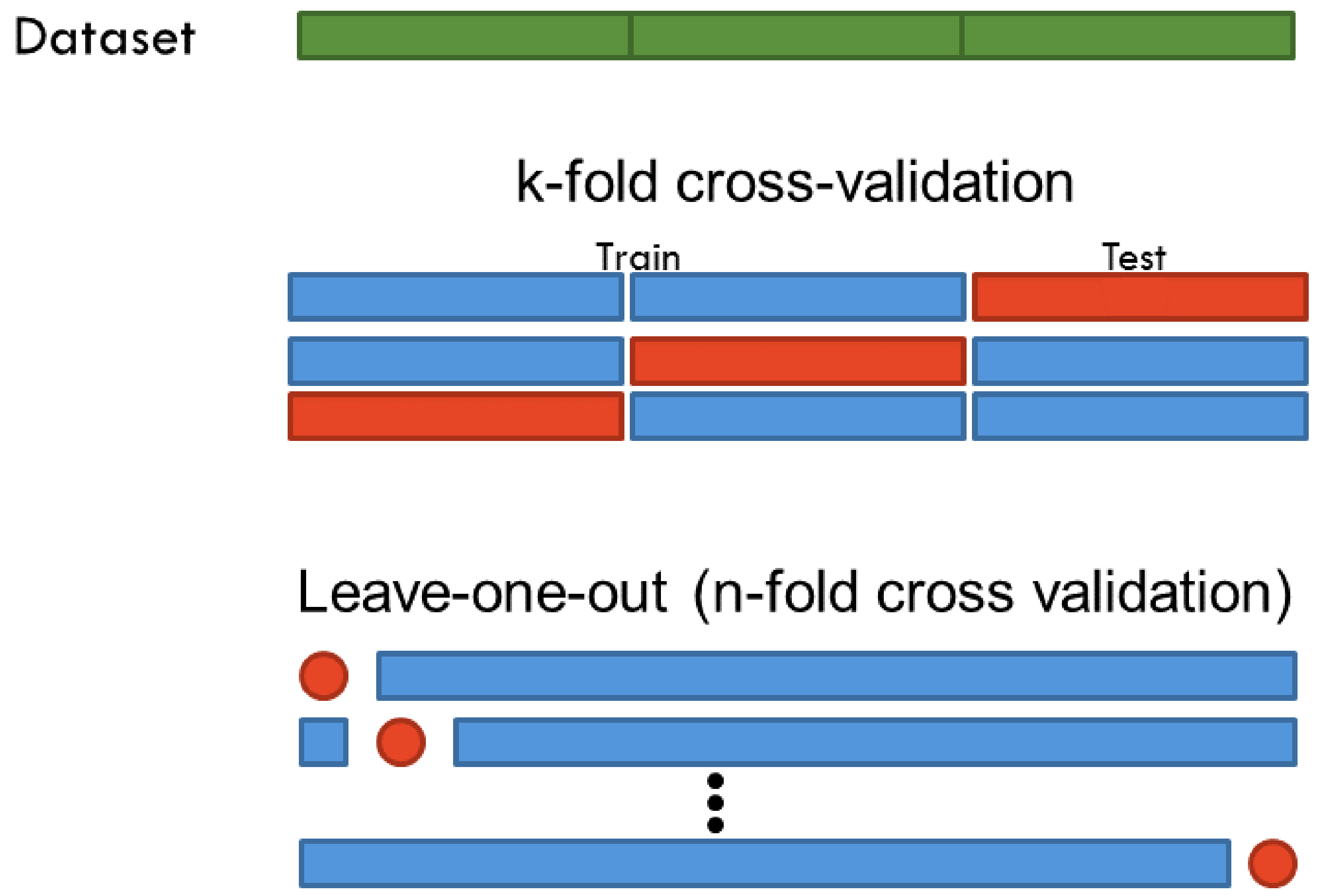

Omics Data
LS
LS
LS
LS
TS
Train / Learn
Logistic regression
ER1
ER2
ER3
ER4
ER5
Error Rate

Cross-validation

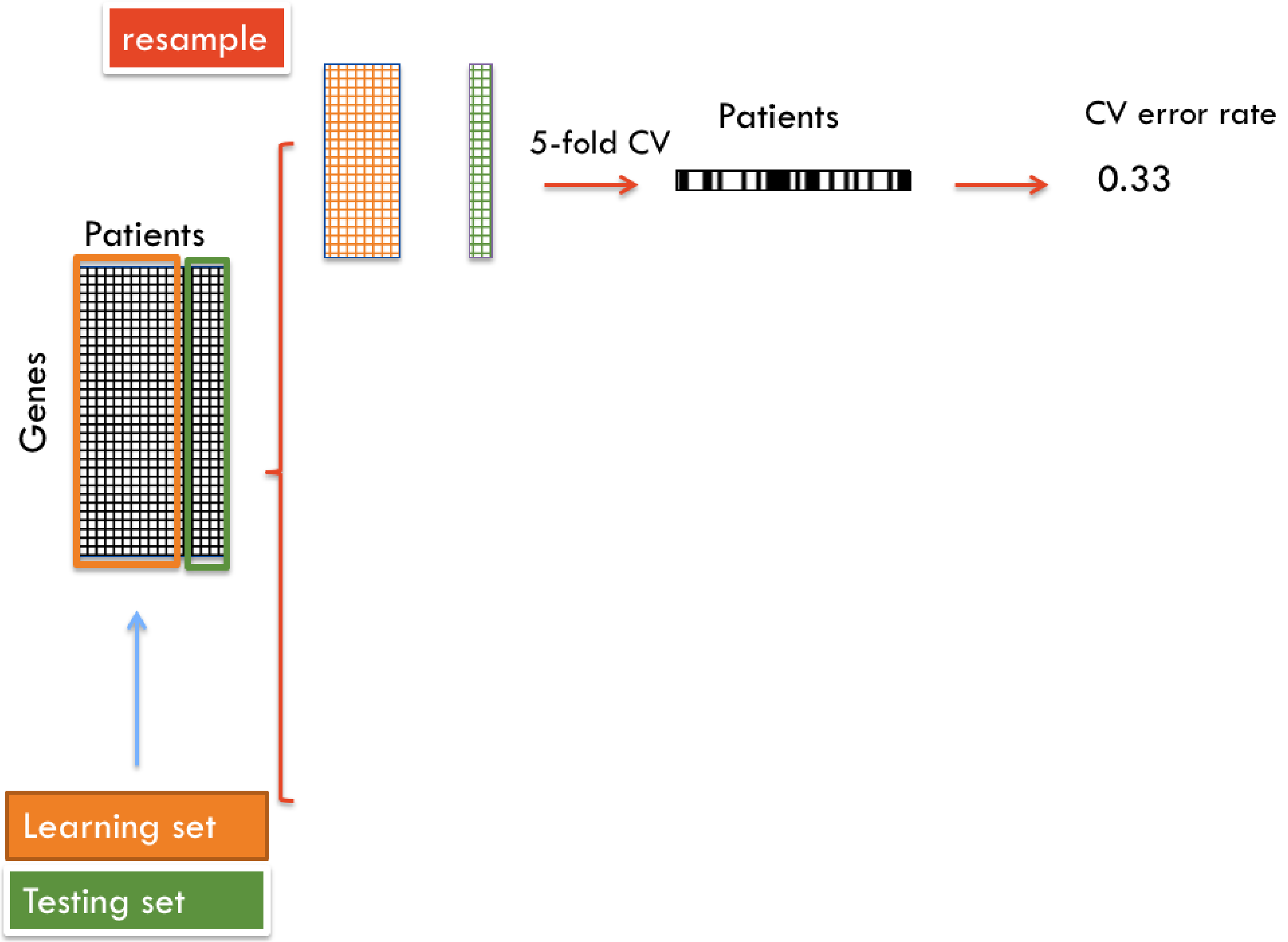
- Cross-validation is the procedure of selecting features and training a classifier on a set of samples and making predictions on a distinct set of samples.

- There are many cross-validation schemes commonly used in practice.
    - *k*-fold cross-validation
    - Leave-one-out cross-validation
    - Repeated *k*-fold cross-validation

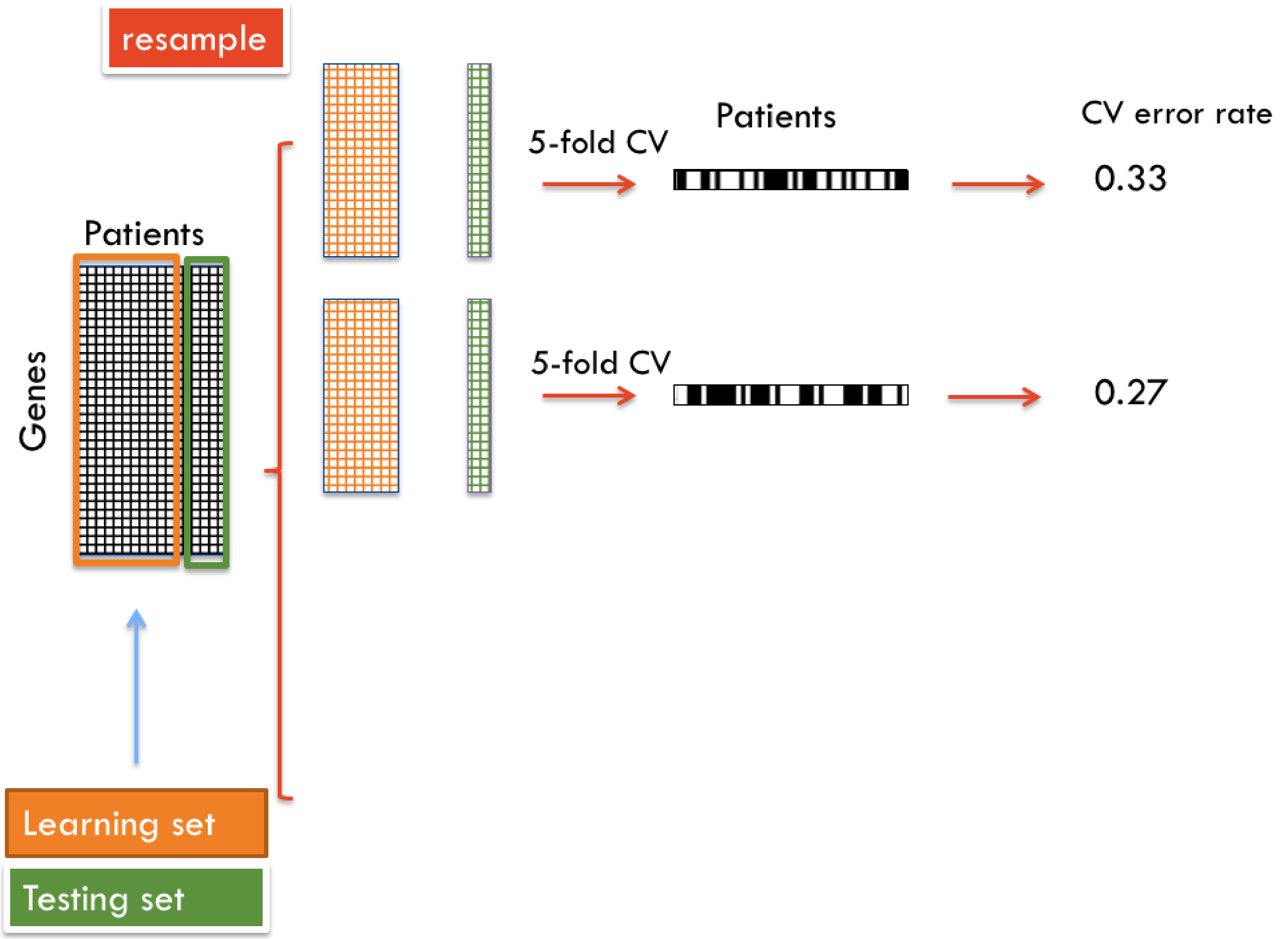
Common Splitting Strategies



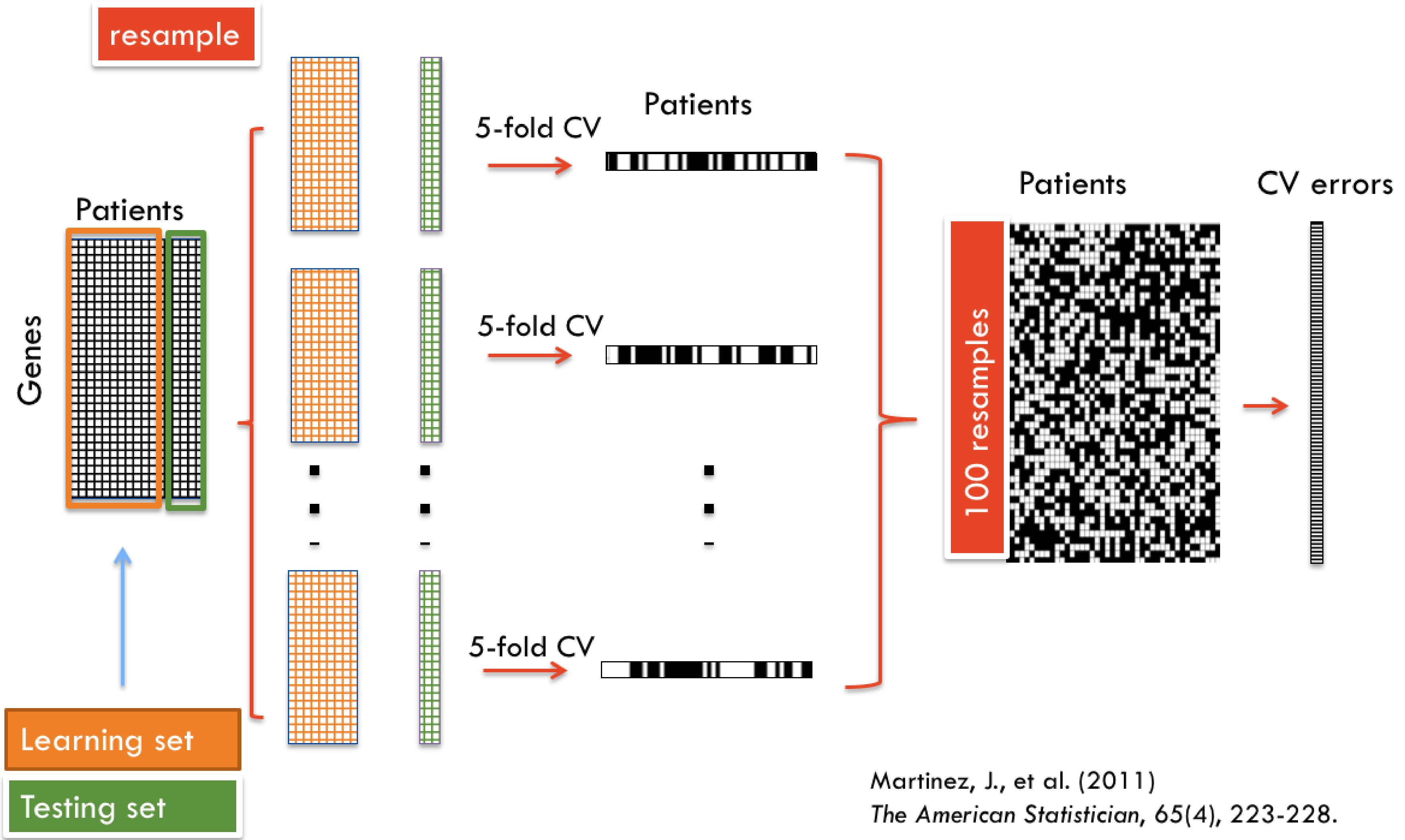
Average 5-fold cross validation



Average 5-fold cross validation



Average 5-fold cross validation



Martinez, J., et al. (2011)
The American Statistician, 65(4), 223-228.

Rpackage: ClassifyR

Reproducible Cross-validation

- A standardised form of cross-validation is not provided by a standard R installation. Often, researchers code their own cross-validation loop for each project, allowing opportunities for implementation inconsistencies to occur.
- A few frameworks have been developed (e.g. MCRestimate, MLInterfaces, caret) but their focus is on classification, so evaluation of the features and predictions is not comprehensive.
- Input formats of existing frameworks don't seamlessly handle new data containers for omics data sets, such as `MultiAssayExperiment`.
- `ClassifyR` provides a standardised cross-validation framework with a focus on performance evaluation and seamlessly integrates with `MultiAssayExperiment`.

ClassifyR Framework

- A *framework* for feature selection, cross-validated classification and its performance evaluation.
- Some popular feature selection methods and classifiers implemented in the package.
- Runs cross-validation in parallel on Windows, MacOS, Linux operating systems.
- Supports numeric-only (`matrix`) data, mixed numeric-categorical (`DataFrame`) data and multi-omics data (`MultiAssayExperiment`).
- Continually maintained and supported (first released in 2014).

Key concepts

- Each stage of classification is defined by a parameter object.
- The three key objects you should be aware of when using ClassifyR
    - `SelectParams` Feature selection for choosing which genes go into the model.
    - `TrainParams` This object is where you define your classifier eg. DLDA
    - `ClassifyResult` The object which will store the results from your CV performed by `runTests`.

Running cross-validation with ClassifyR

- The default feature selection method of `SelectParams` is a moderated t-test based ranking and selection of the top p genes that give the best resubstitution error (considering 10, 20, ..., 100 top-ranked features).
- The default training and prediction methods for `TrainParams` are for Diagonal Linear Discriminant Analysis (DLDA).
- A 20 permutations and 5 folds cross-validation using default selection and classification methods is done using `runTests`.

```r
library(ClassifyR)
classifiedDLDA <- runTests(measurements = measurementsVS, classes = classes,
                           datasetName = "AML", classificationName = "DLDA",
                           permutations = 20, seed = 2018)
```

Accuracy

- The overall proportion of predictions which were correct.

Confusion matrix

Actual \ Predicted	Negative	Positive
Negative	True Negative (TN)	False Negative (FN)
Positive	False Positive (FP)	True Positive (TP)

- Accuracy = (TP + TN) / (TP + TN + FP + FN)

```
classifiedDLDA <- calcCVperformance(classifiedDLDA, "accuracy")
performance(classifiedDLDA)["Accuracy"]
```

```
## $Accuracy
##  [1] 0.6553191 0.6680851 0.6468085 0.6765957 0.6723404 0.6893617 0.6638298
##  [8] 0.6808511 0.6978723 0.6978723 0.6680851 0.6893617 0.7148936 0.6936170
## [15] 0.6851064 0.7276596 0.7021277 0.6765957 0.6978723 0.6680851
```

Error Rate

- The proportion of samples which were assigned to the incorrect class by the classifier.

Confusion matrix

Actual \ Predicted	Negative	Positive
Negative	True Negative (TN)	False Negative (FN)
Positive	False Positive (FP)	True Positive (TP)

- Error rate = (FP + FN) / (TP + TN + FP + FN) = 1 - Accuracy

```
classifiedDLDA <- calcCVperformance(classifiedDLDA, "error")
performance(classifiedDLDA)["Error Rate"]
```

```
## $`Error Rate`
##  [1] 0.3446809 0.3319149 0.3531915 0.3234043 0.3276596 0.3106383 0.3361702
##  [8] 0.3191489 0.3021277 0.3021277 0.3319149 0.3106383 0.2851064 0.3063830
## [15] 0.3148936 0.2723404 0.2978723 0.3234043 0.3021277 0.3319149
```

Other metrics

- Balanced Error Rate
    - Simply the average error rate of each class.
    - Provides a fair evaluation for imbalanced data sets (each class contributes equally).
    - Same as ordinary error rate for balanced data sets.
- Precision
    - The proportion of predictions of the Positive class which are truly Positive.
- Recall
    - Proportion of the Positives that are predicted correctly.

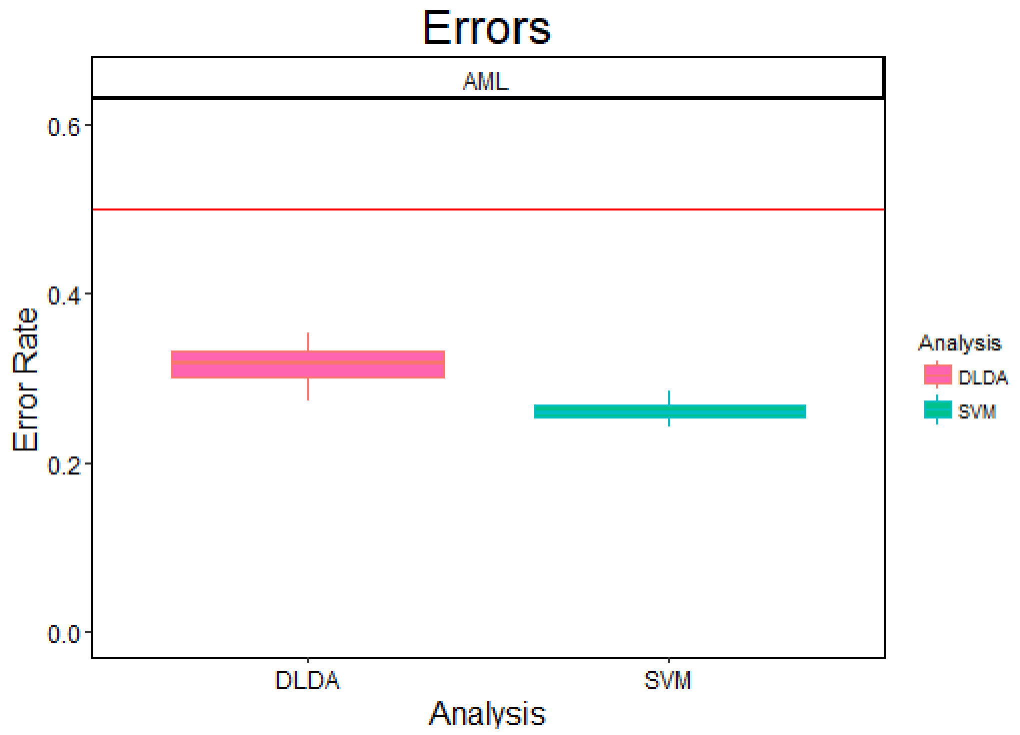
SVM

Perform 5-fold cross-validation on a Support Vector Machines classifier

```
trainParams <- TrainParams(SVMtrainInterface)
predictParams <- PredictParams(SVMpredictInterface,getClasses = function(result) result)
classifiedSVM <- runTests(measurementsVS, classes, "AML", "SVM",permutations = 20,
                          seed = 2018, params = list(trainParams, predictParams))
```

Performance comparison

```r
classifiedSVM <- calcCVperformance(classifiedSVM, "error")
performancePlot(list(classifiedDLDA, classifiedSVM),
                performanceName = "Error Rate", title = "Errors", yLimits = c(0,0.6),
                plot=FALSE) + geom_hline(yintercept = 0.5, colour = "red")
```



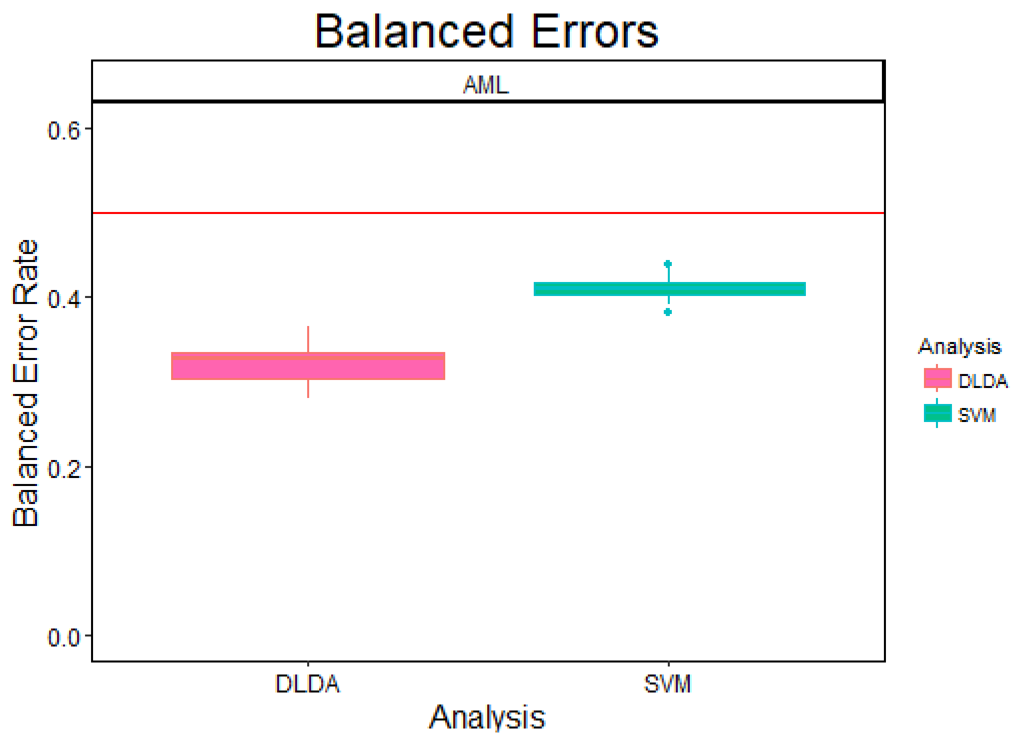
Sample-specific error rate

The function `calcCVperformance` can be used to calculate sample-specific error rates for each patient.

```r
classifiedDLDA <- calcCVperformance(classifiedDLDA, "sample error")
classifiedSVM <- calcCVperformance(classifiedSVM, "sample error")
errorPlot <- samplesMetricMap(list(classifiedDLDA, classifiedSVM), xAxisLabel = "Samples",
                              yAxisLabel = "Classifier", showXtickLabels = FALSE)
```

Performance comparison

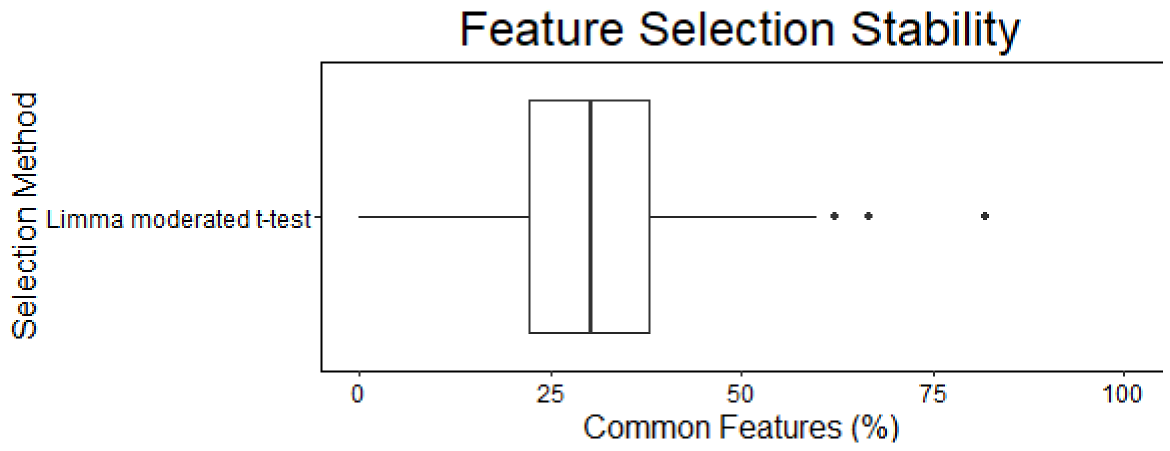
```
classifiedDLDA <- calcCVperformance(classifiedDLDA, "balanced error")
classifiedSVM <- calcCVperformance(classifiedSVM, "balanced error")
performancePlot(list(classifiedDLDA, classifiedSVM), performanceName = "Balanced Error Rat
                title = "Balanced Errors", yLimits = c(0,0.6), plot=FALSE) +
                geom_hline(yintercept = 0.5, colour = "red")
```



Model Stability

Plot the distribution of overlaps of selected features used in the DLDA classifier.

```r
withinChoices <- selectionPlot(list(classifiedDLDA),
                               xVariable = "selectionName", xLabel = "Selection Method",
                               columnVariable = "None",
                               boxFillColouring = "None", boxLineColouring = "None",
                               rotate90 = TRUE)
```



Performance assessment

- Cross-validation to evaluate classifier performance
- Evaluation of overall error, sample-specific error, precision, recall.
- Feature selection stability.

Now: Hands-on session