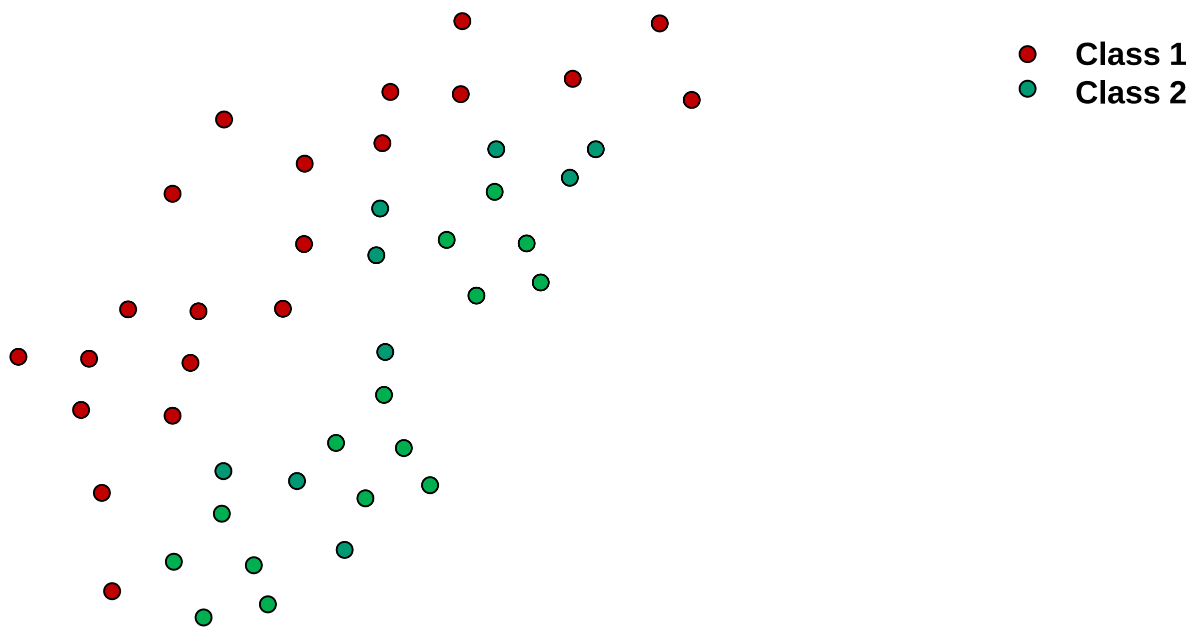
APPLIED MACHINE LEARNING

Evaluating the performance of classifiers

Estimating from sampling the datapoints

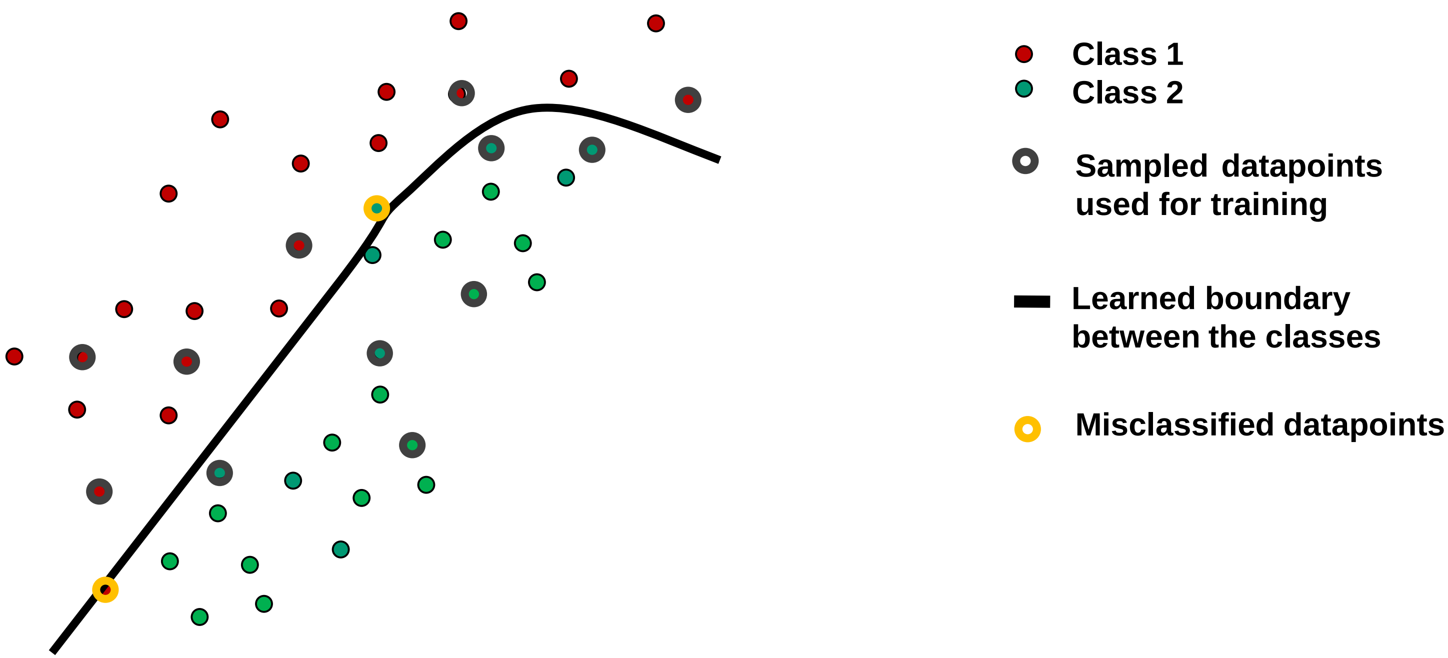


If **one trains the algorithm with all datapoints**, one cannot test if the algorithm can predict well.

To test the ability of the model to predict correctly the class labels:

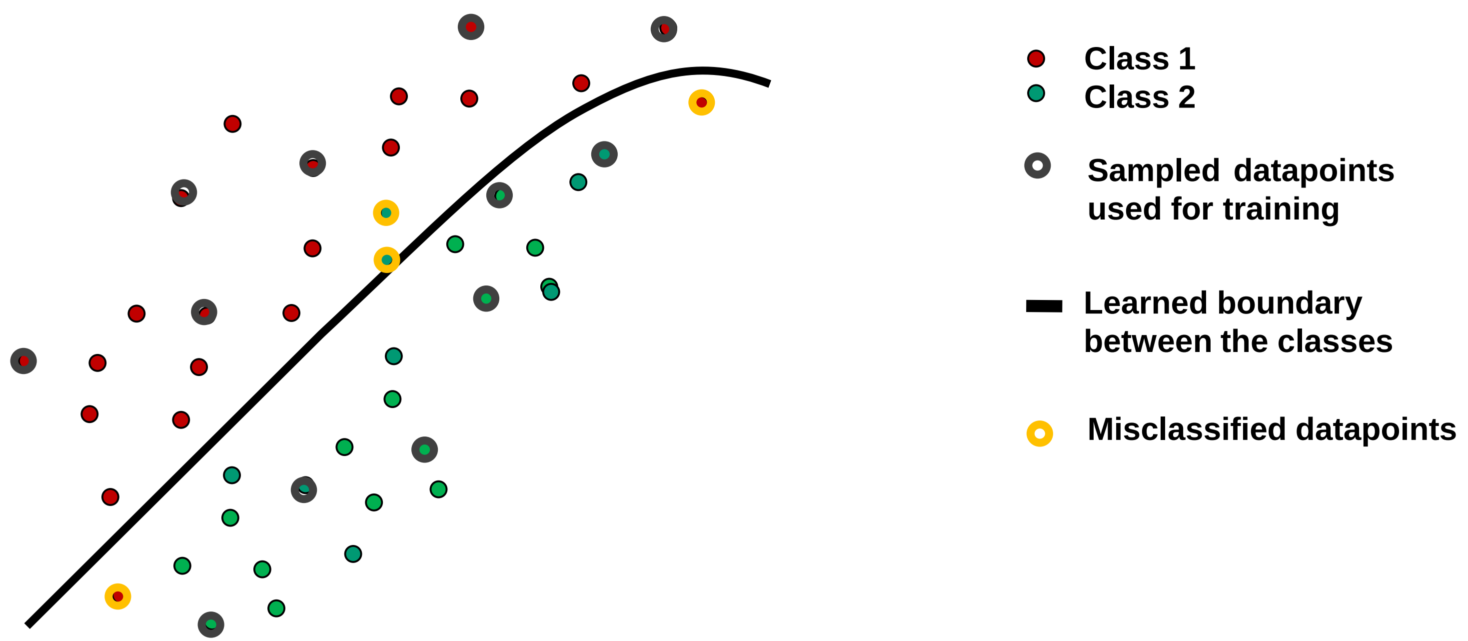
1. Train the model using only a subset of datapoints sampled randomly.
2. Test the prediction of the model on the datapoints not used during training.

Estimating from sampling the datapoints



1) Sample the datapoints
2) Train the algorithm on the sampled points
3) Test the prediction of the learned model on the rest of the points

Estimating from sampling the datapoints



1) Pick another sample of datapoints
2) Train the algorithm on the new sampled points
3) Test the prediction of the learned model on the rest of the points

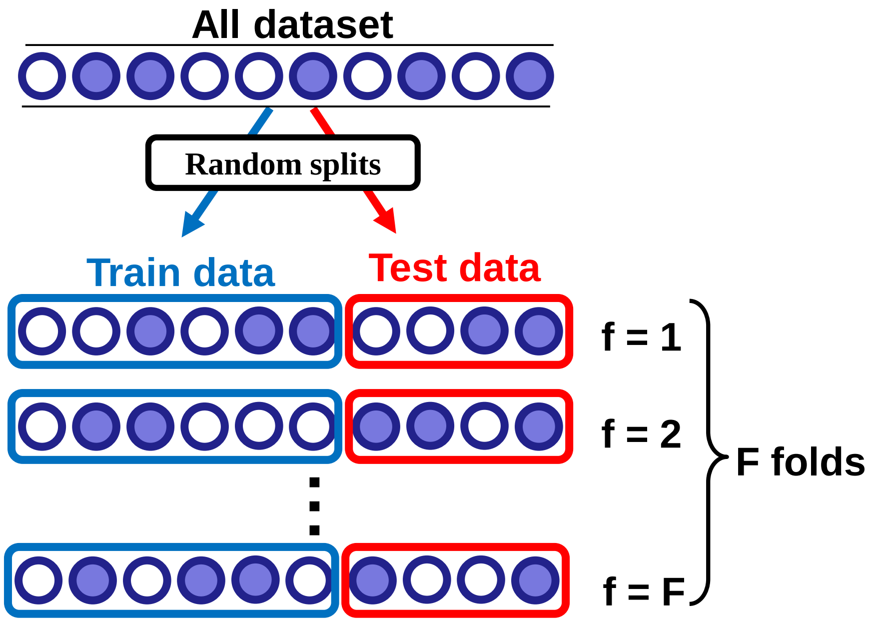
Crossvalidation: repeat training/testing procedure several times and compute average performance.

Crossvalidation

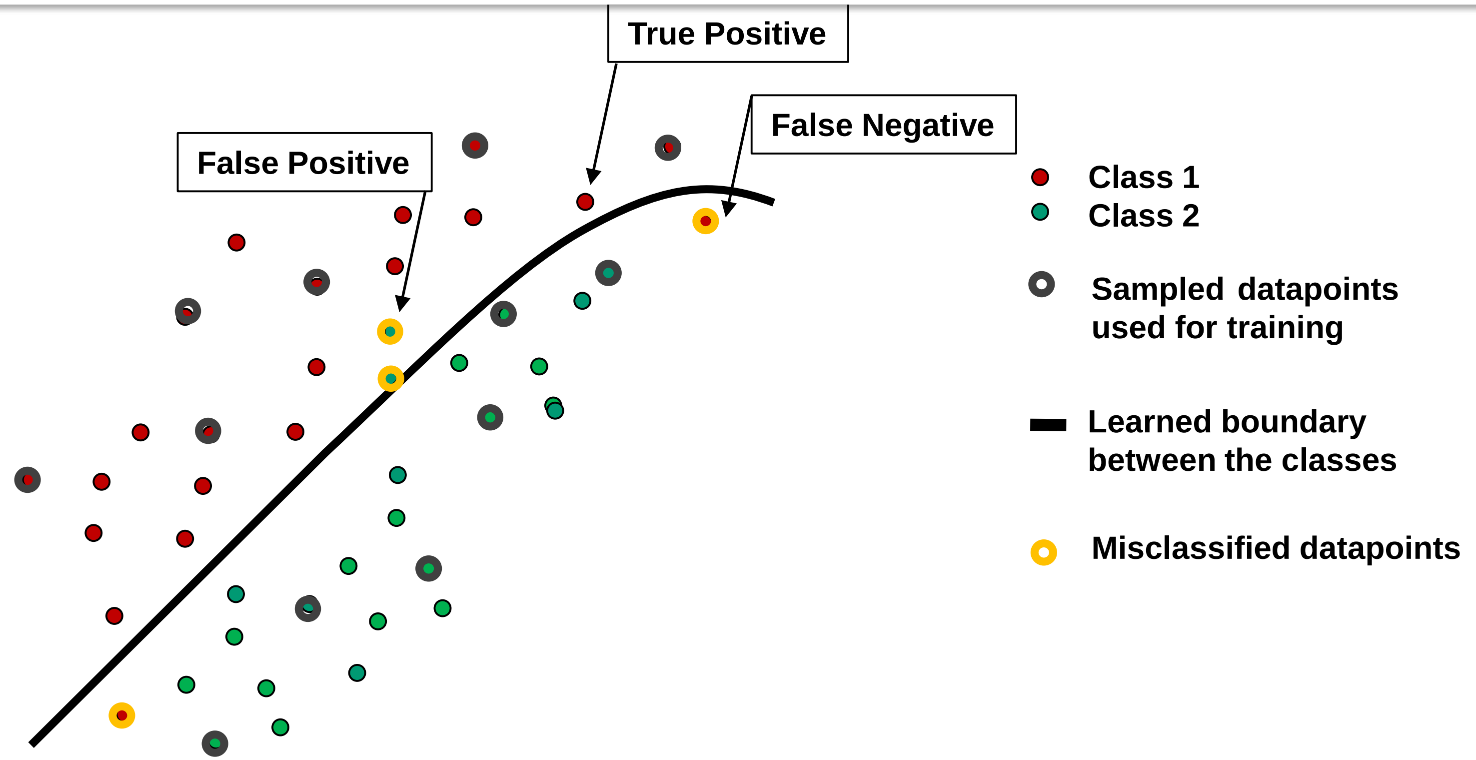
Definition: "Cross validation is the practice of confirming an experimental finding by repeating the experiment using an independent assay technique"

f-fold cross validation

- Constant Train/Test ratio
- At each iteration:
  1) Random split of the data between Train and Test
  2) Repetition of classification
- Averaging of the result across folds



Quantifying Performance



True Positives(TP): nm of datapoints of class 1 that are correctly classified

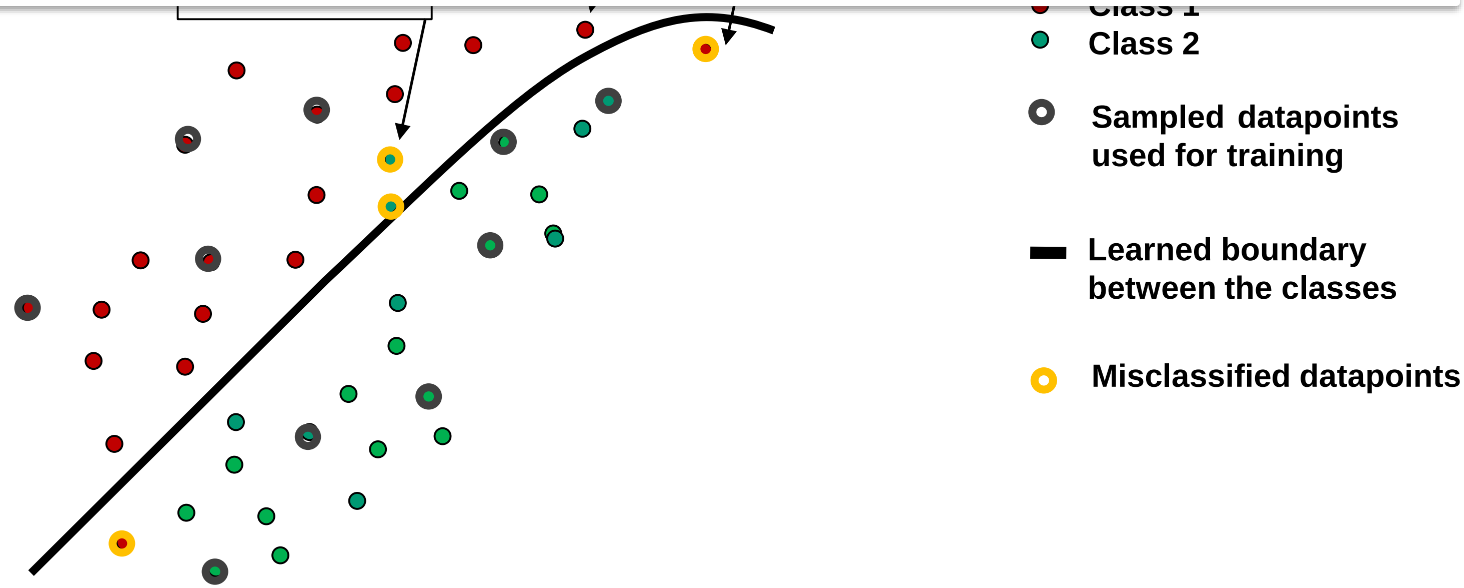
False Negatives (FN): nm of datapoints of class 1 that are incorrectly classified

False Positives(FP): nm of datapoints of class 2 that are incorrectly classified

The F-measure

F-measure finds a tradeoff between classifying correctly all datapoints of the same class and making sure that each class contains points of only one class.

The classification F-measure is similar but is not the same F-measure as the F-measure we saw for clustering!



Recall: $\dfrac{TP}{TP+FN}$

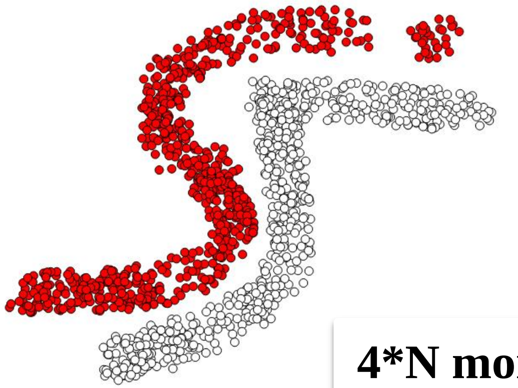
Precision: $\dfrac{TP}{TP+FP}$

$$F = \frac{2\text{*Precision*Recall}}{\text{Precision+Recall}}$$

Recall: Proportion of datapoints from class 1 correctly classified.

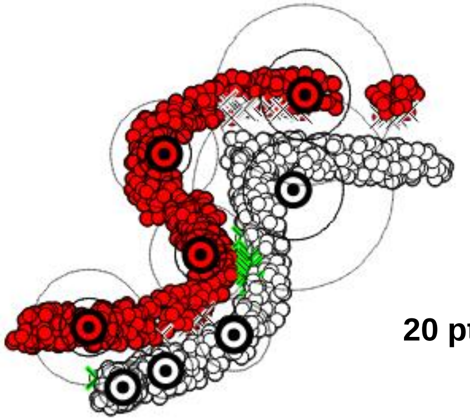
Precision: proportion of datapoints of Class 1 correctly classified over all datapoints classified as Class 1.

How to determine the optimal model?



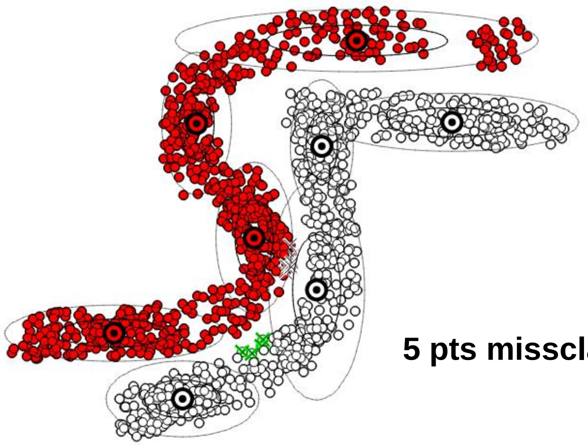
4*N more parameters for small improvement

Spherical covariance matrix



20 pts missclassified

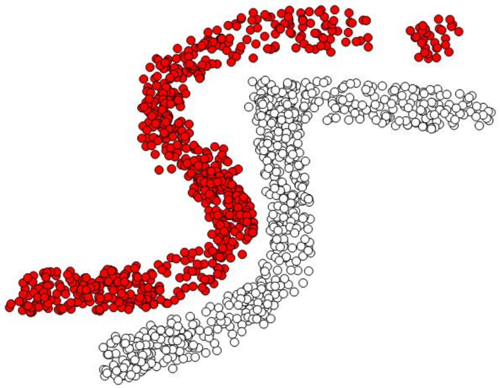
Diagonal covariance matrix

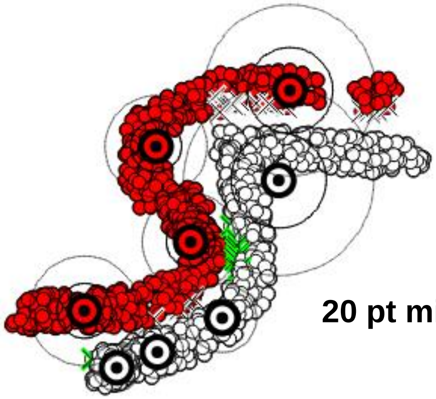


5 pts missclassified

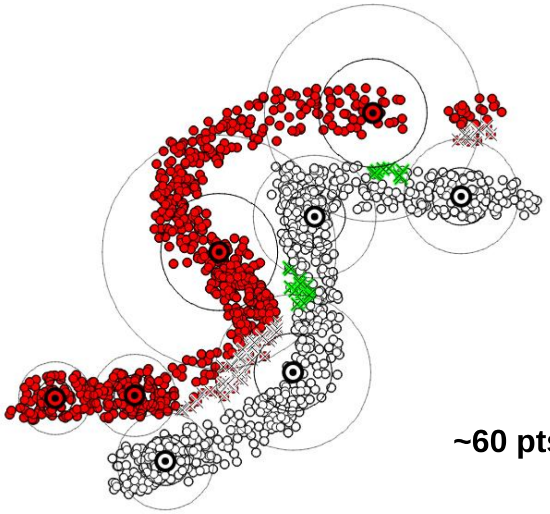
Crossvalidation allows to determine sensitivity to choice of data for training

Crossvalidation

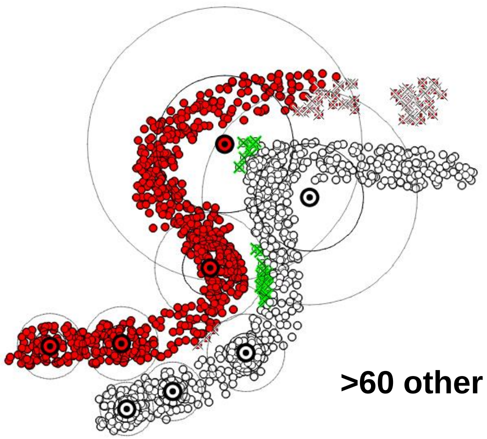




20 pt missclassified



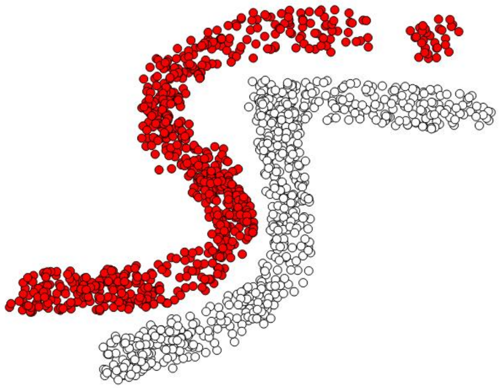
~60 pts missclassified



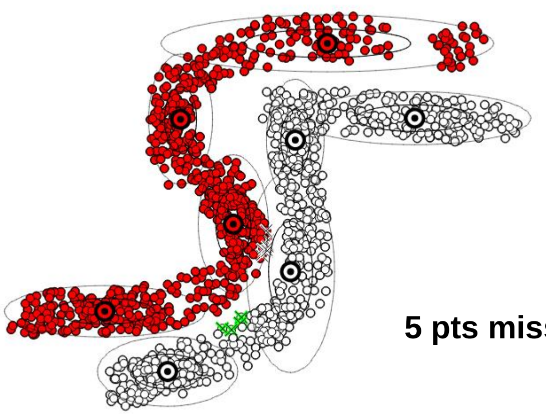
>60 other pts missclassified

Performance vary across each fold

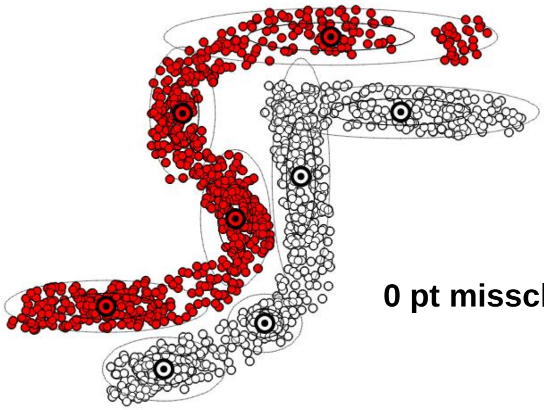
Crossvalidation



1 pt missclassified



5 pts missclassified

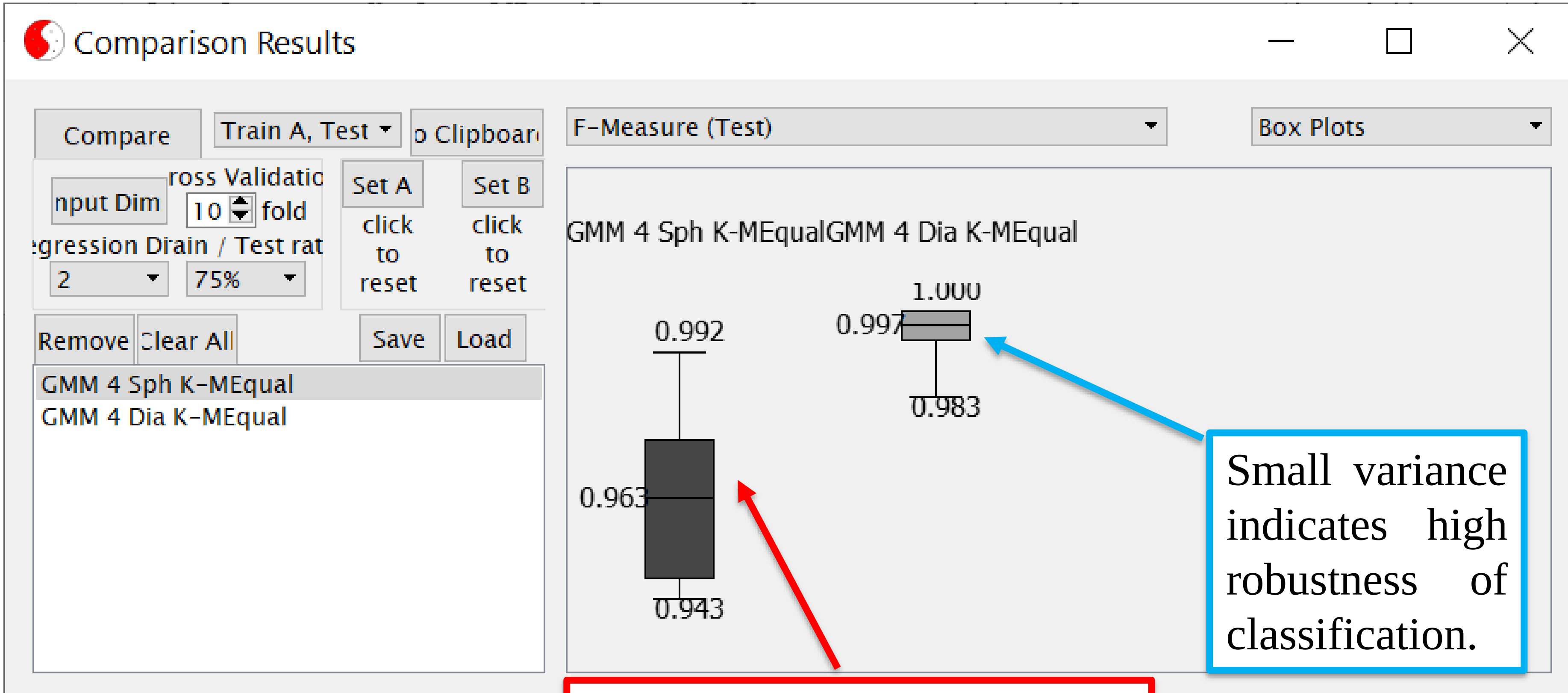


0 pt missclassified

Performance vary across each fold

Crossvalidation on F-measure

Variance of classification performance at testing across the different folds of crossvalidation measures the **sensitivity of the classifier** to the choice of training set and of hyperparameters.

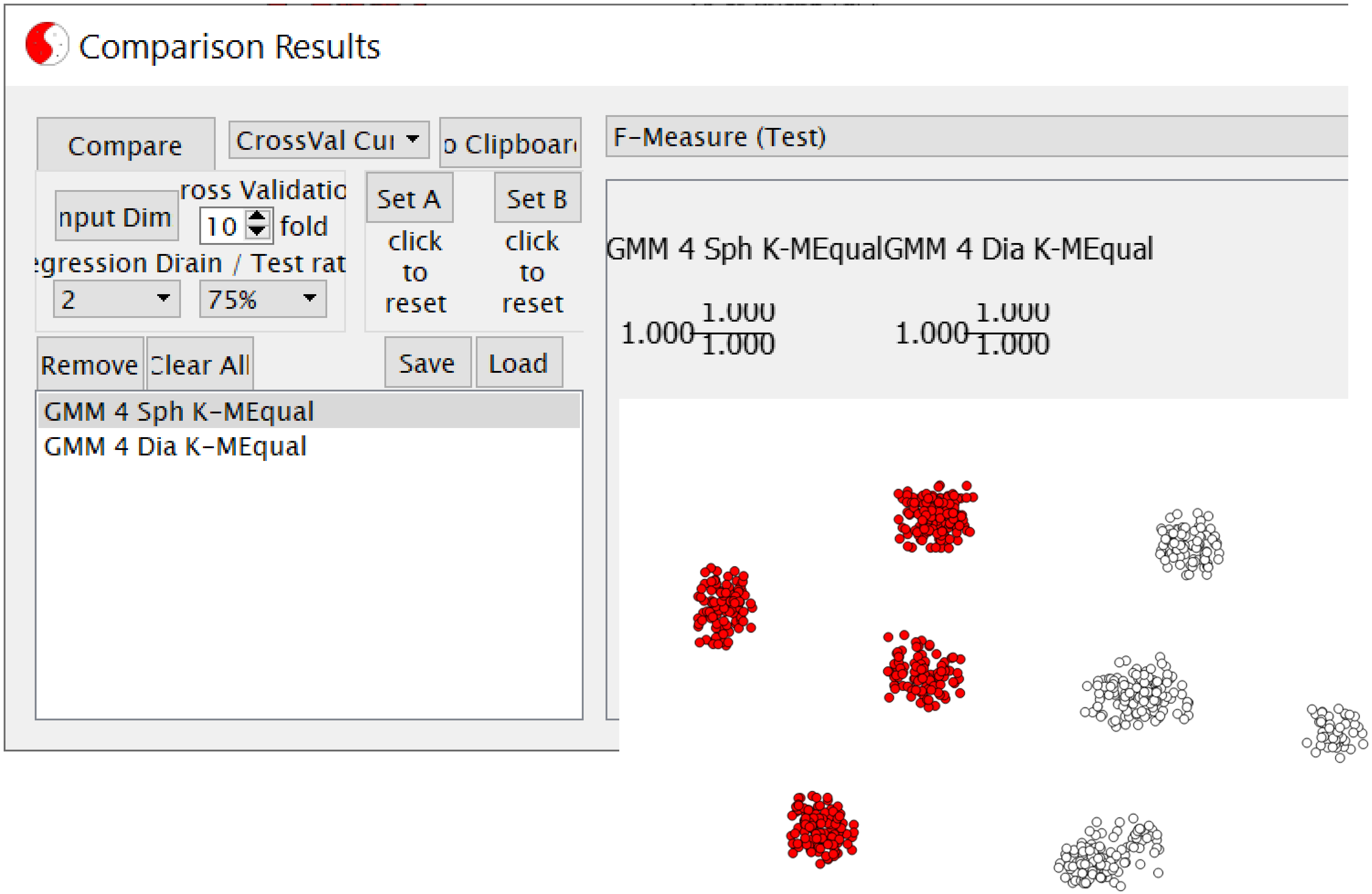


Small variance indicates high robustness of classification.

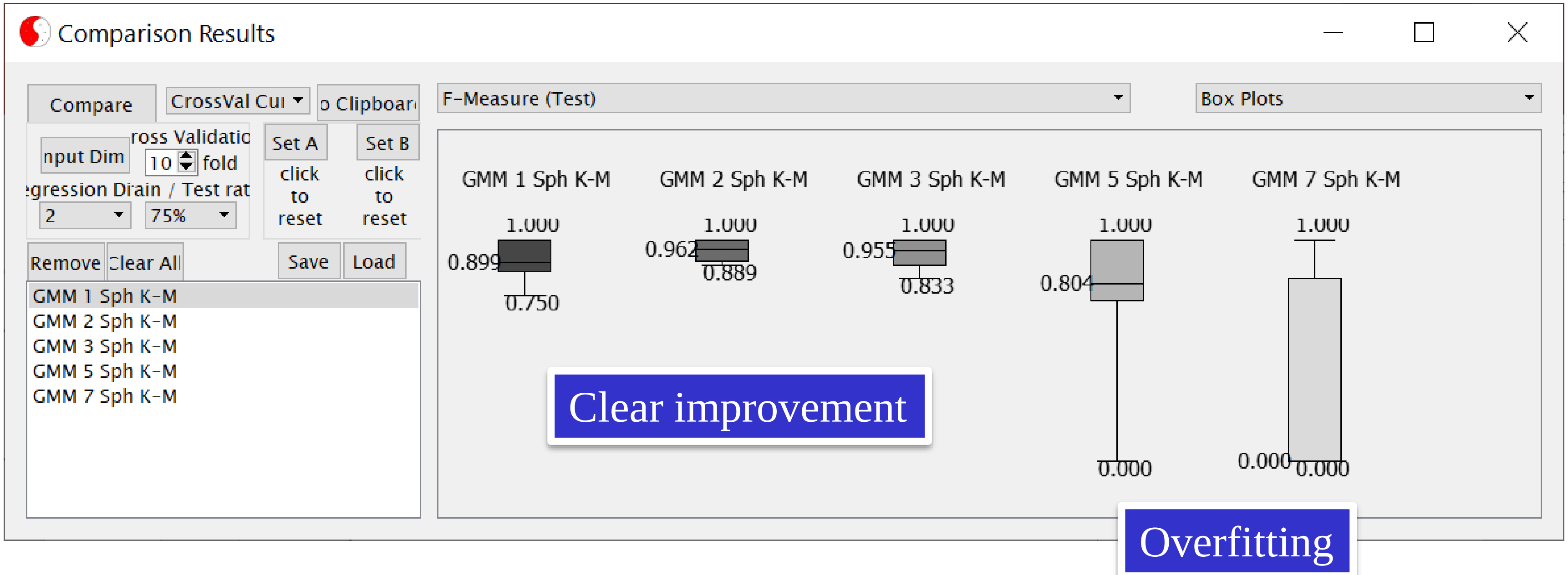
Large variance on testing set may indicate overfitting

Crossvalidation on F-measure

Perfect results across folds indicate ideal classification

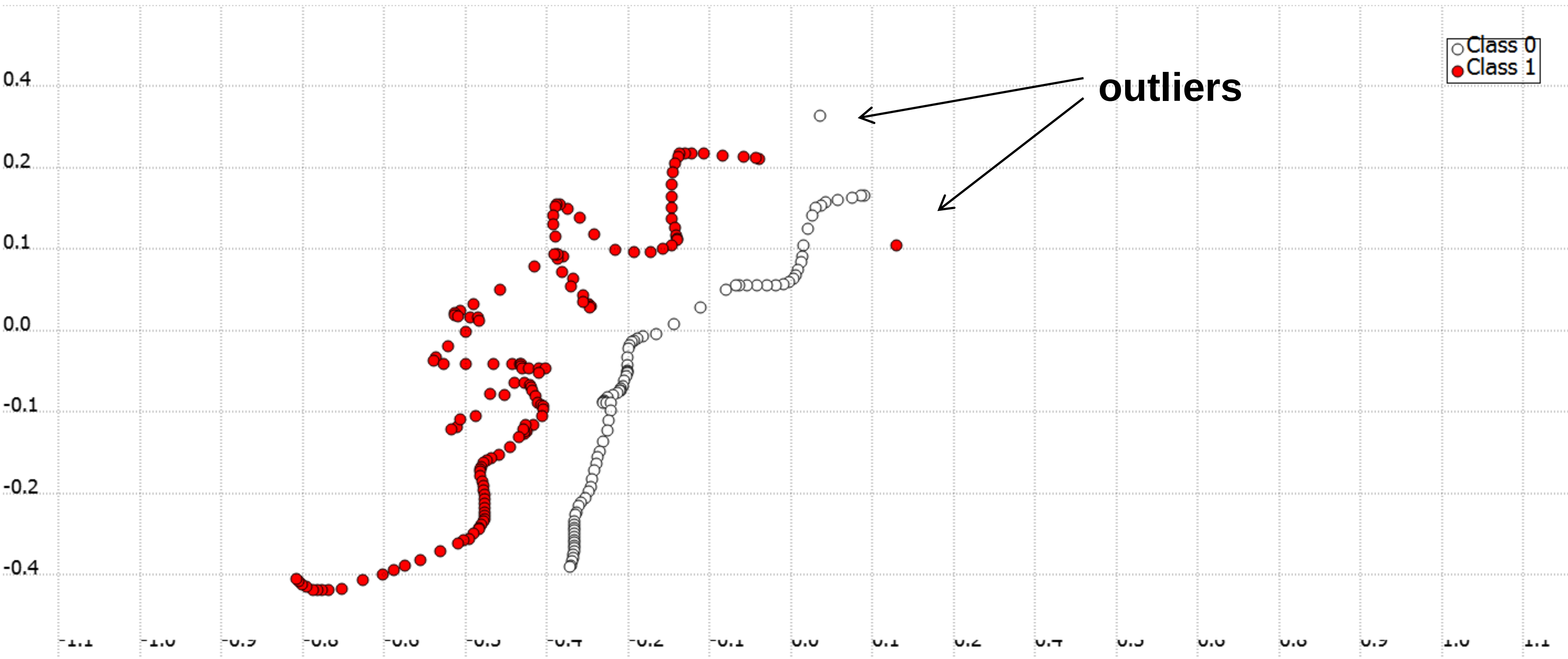


Over-fitting

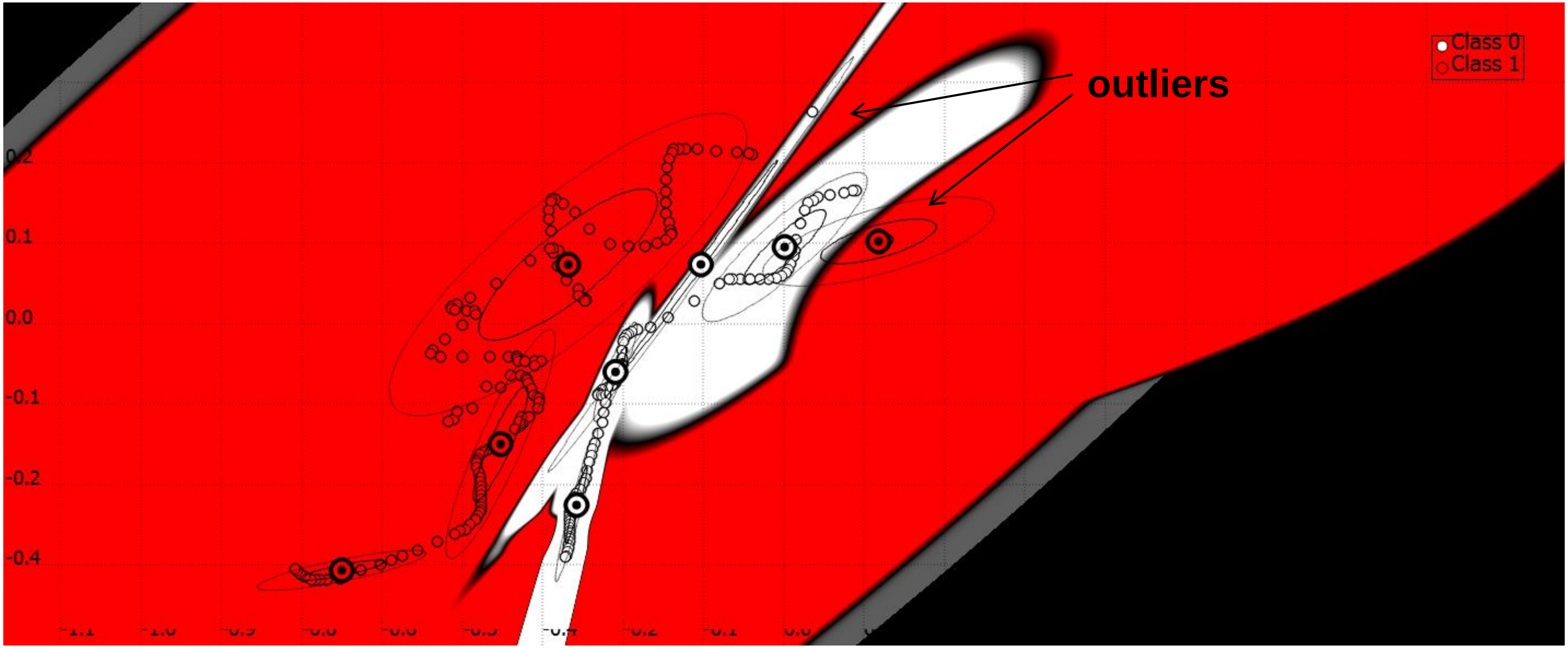


When performing crossvalidation, overfitting can be detected by looking at the variance of the error across crossvalidation rounds. Large variance may be a sign of overfitting (here too many models are used to fit the data, and one starts modeling noise).

When can overfitting arise: example

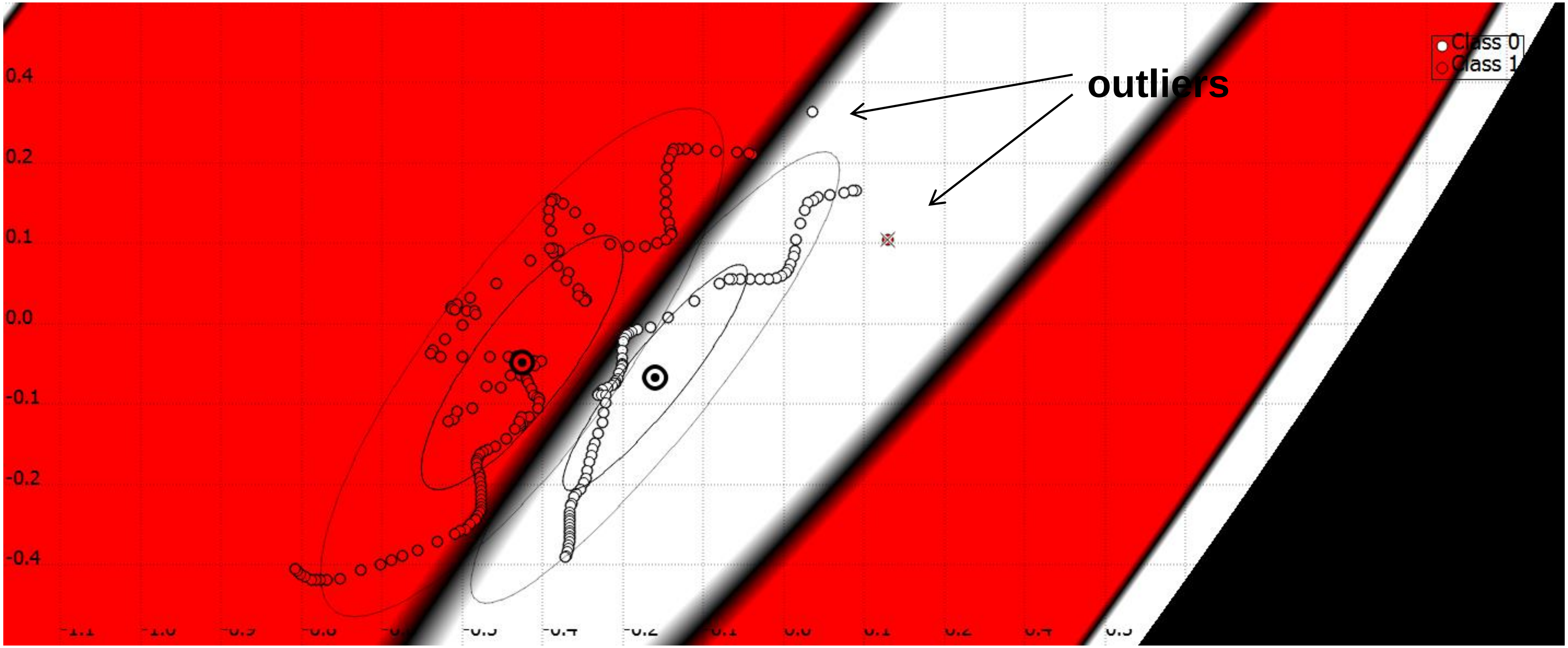


Over-Fitting



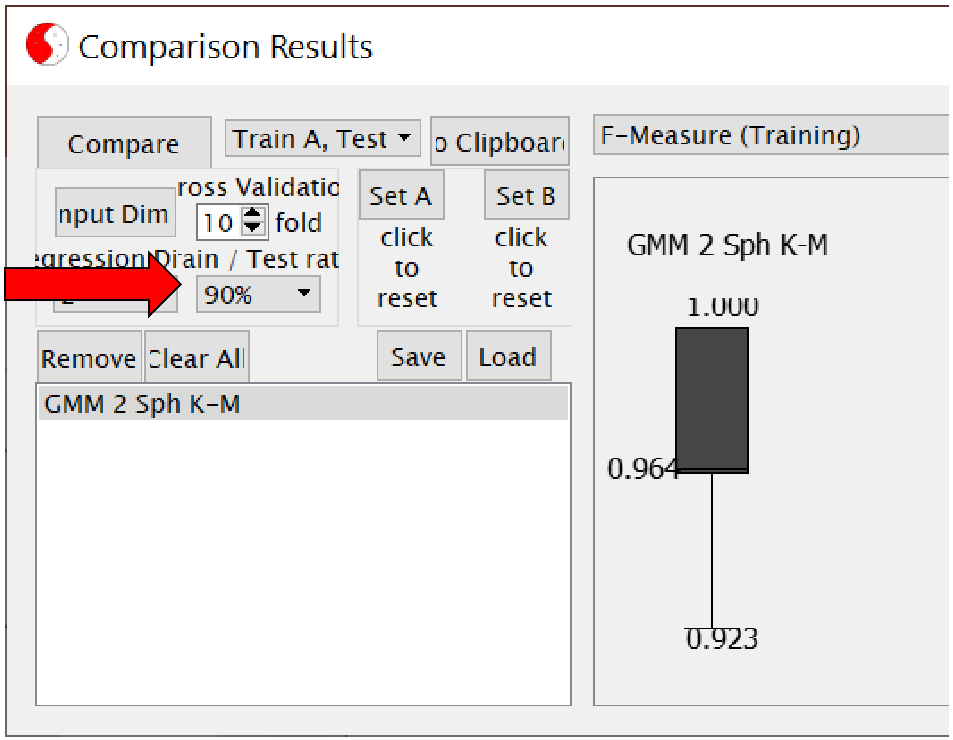
It classifies well all datapoints including outliers but requires 4 Gaussians for each model and overall shape of density not well encapsulated.

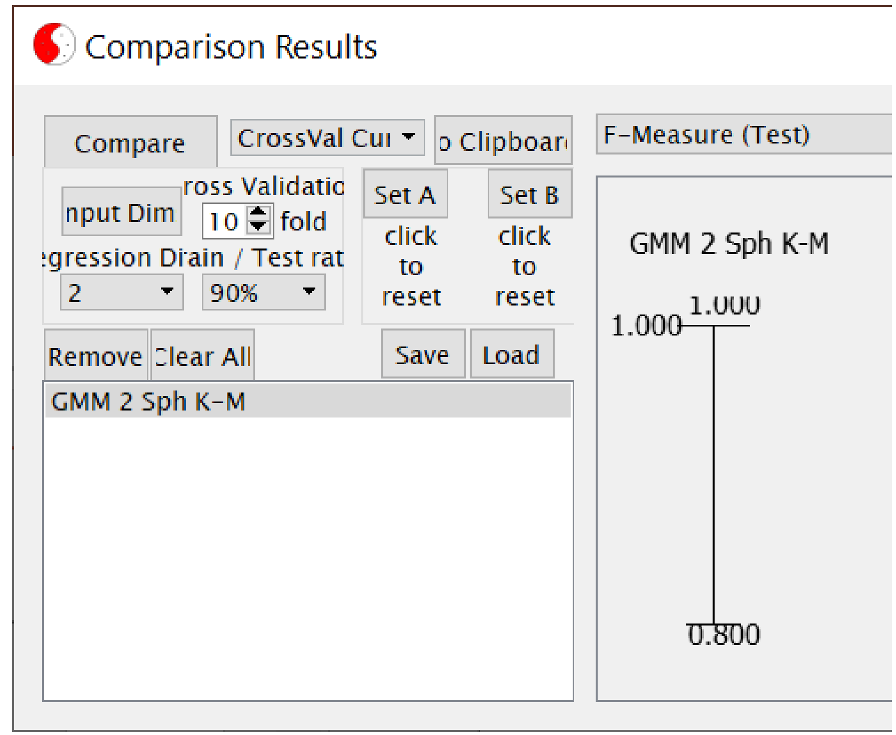
Imperfect classification but no overfitting



Does not classify well one of the outliers but generally a good fit

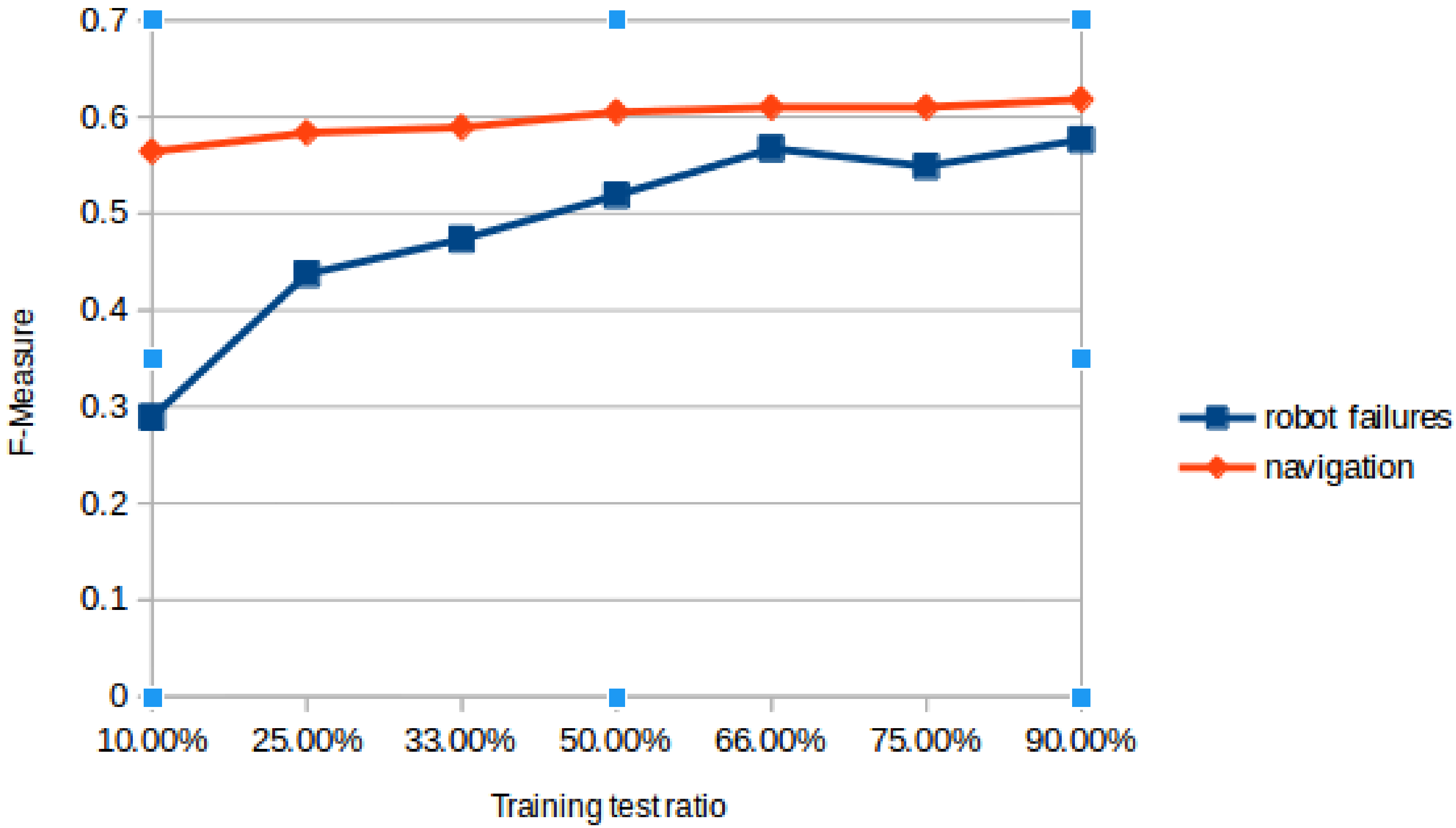
Training/testing ratio and overfitting





Using 90% training/testing ratio may lead to poor generalization, whereby you get excellent performance at training but poorer performance at testing.
To assess proper generalization, you expect similar performance (mean and std of F-measure) on both training and testing sets.

Choice of training/testing ratio

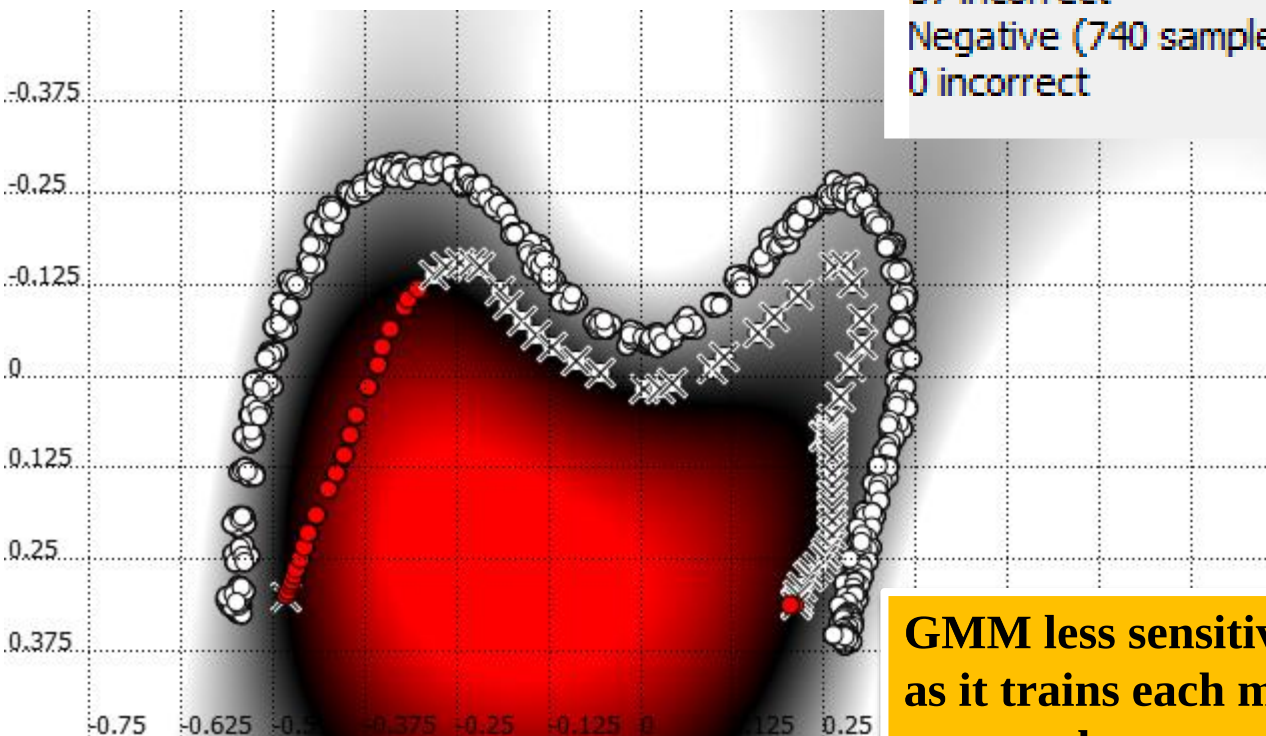


The same classifier can achieve its peak performance
for different ratios depending on the dataset.

Sensitivity of performance to distribution across classes

Classification may appear good overall but be very poor for one of the classes if instances of each class are **not well balanced.**

Training Set (821 samples):
Positive (81 samples): 24 correct (29%)
57 incorrect
Negative (740 samples): 740 correct (100%)
0 incorrect



Model learned with SVM (see next week)

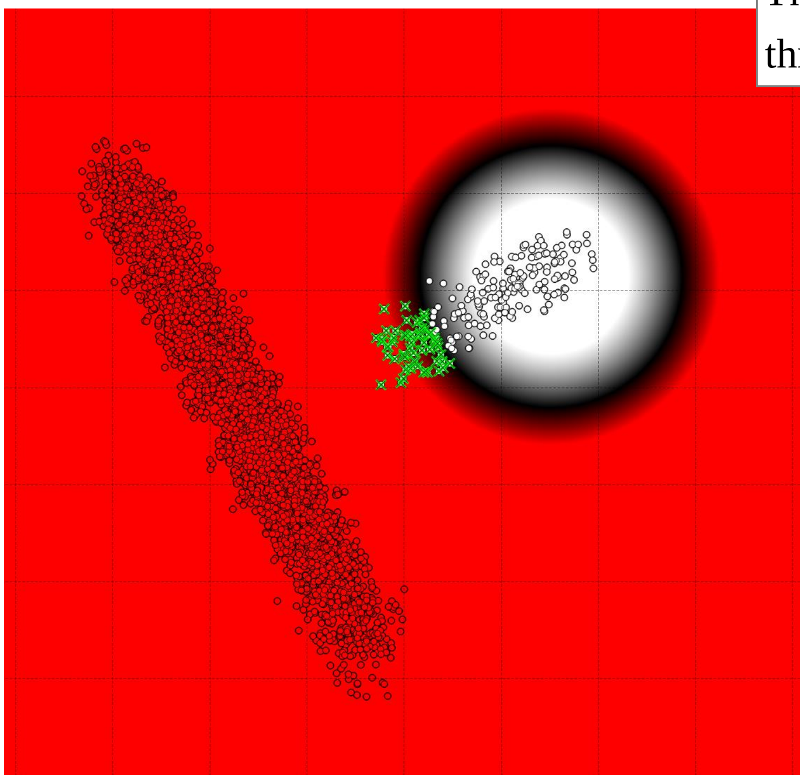
GMM less sensitive to unbalanced datasets as it trains each model on each class separately.

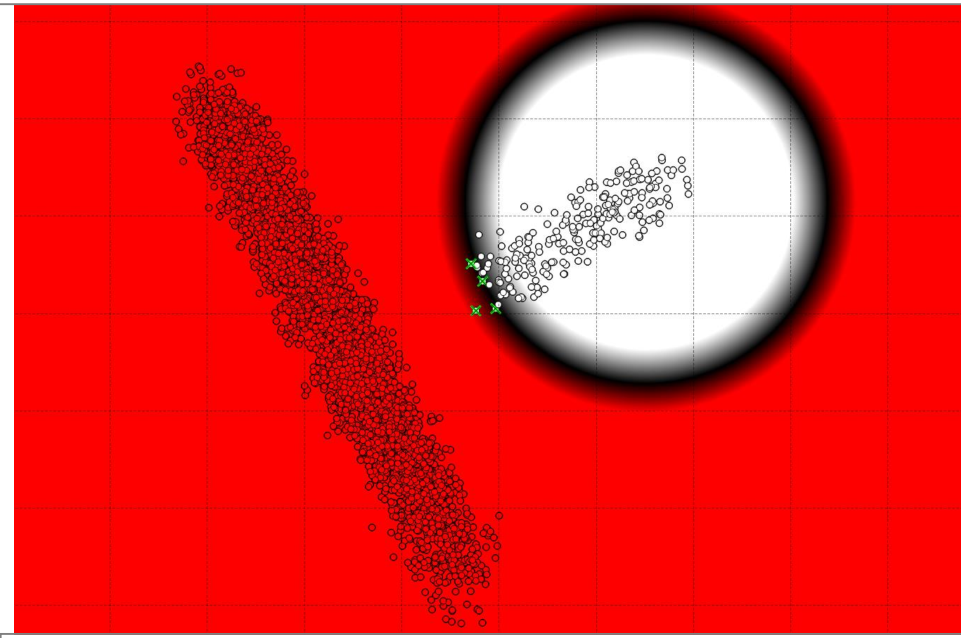
Classification with GMM-s

Unbalanced dataset:

3950 Samples
3720 Positives
230 Negatives

The importance of one class over another one can be modulated through the probability of observing one class.





We set $p(y=0)=16\bullet p(y=1)$ in the Bayes' decision rule

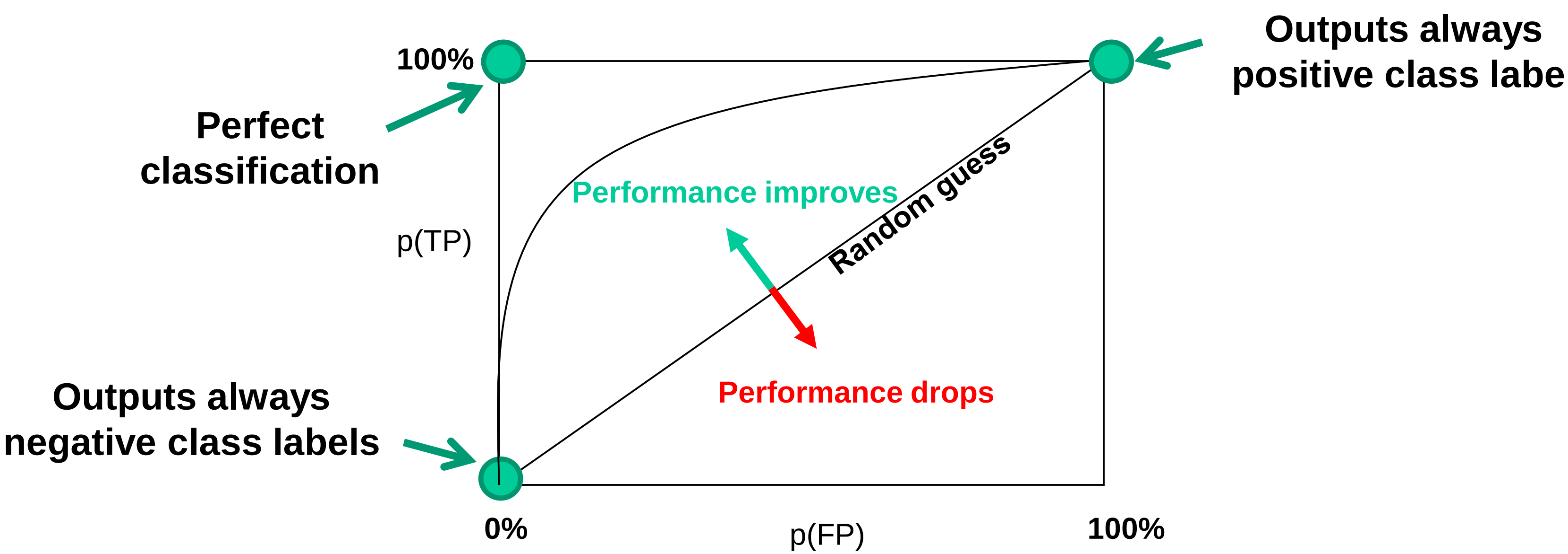
Unbalanced class distribution

Force equal class distribution

Determining the hyperparameters

The ROC (Receiver Operating Characteristic) curve plots the fraction of true positives and false positives over the total number of samples (for binary classification only).

Each point on the curve corresponds to a different value of the classifier's hyperparameter (e.g. a threshold on Bayes' classification).

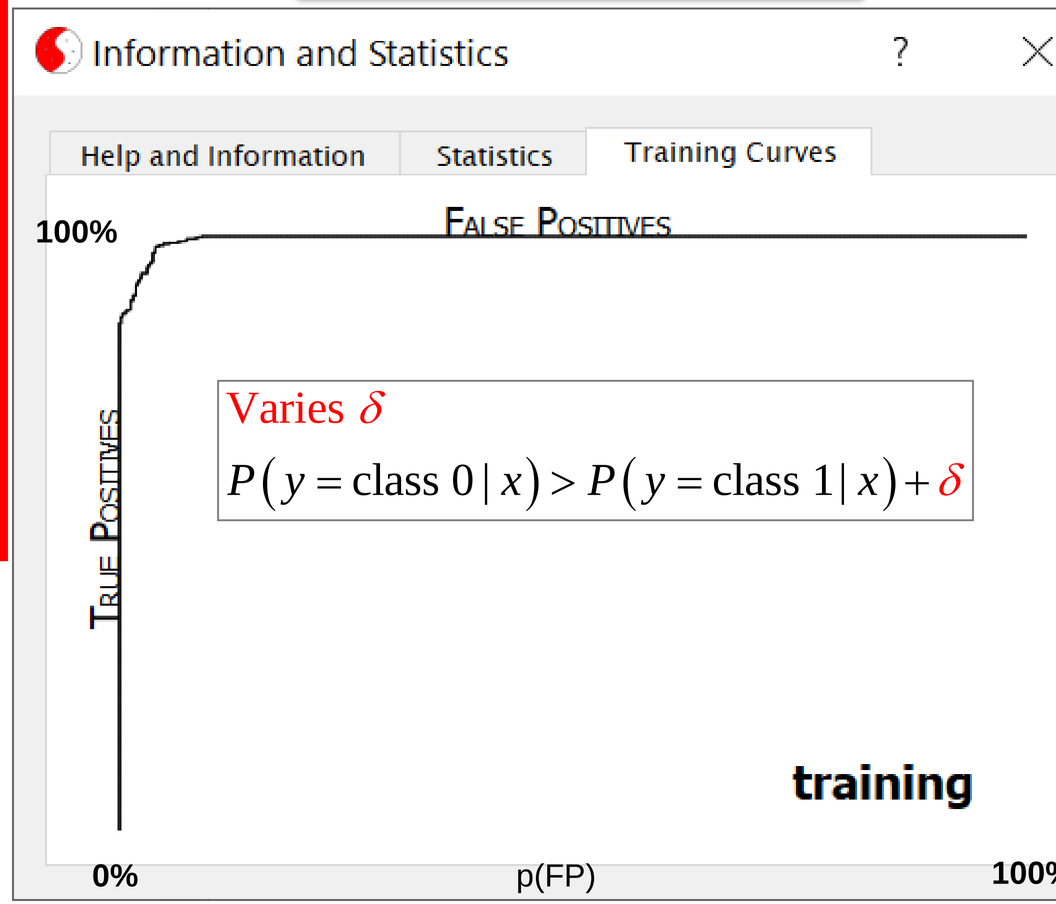


True Positives(TP): nm of datapoints of class 1 that are correctly classified

False Positives(FP): nm of datapoints of class 2 that are incorrectly classified

Determining the hyperparameters

Class 0 is positive class



Varies δ

$$P\left(y=\text{class } 0 \mid x\right) > P\left(y=\text{class } 1 \mid x\right)+\delta$$

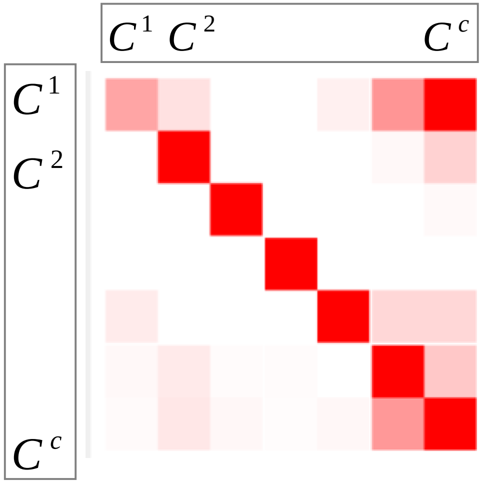
Sensitivity of performance for multi-classes

Confusion matrices provide information on which classes are merged (confused) with which by the classifier

Actual Class \ Predicted Class	C_1	C_2	$\cdots$	C_c
C_1	n_{11}	n_{12}	$n_{1\ldots}$	n_{1c}
C_2	n_{21}	n_{22}	$n_{2\ldots}$	n_{2c}
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
C_c	n_{c1}	n_{c2}	$\cdots$	n_{cc}

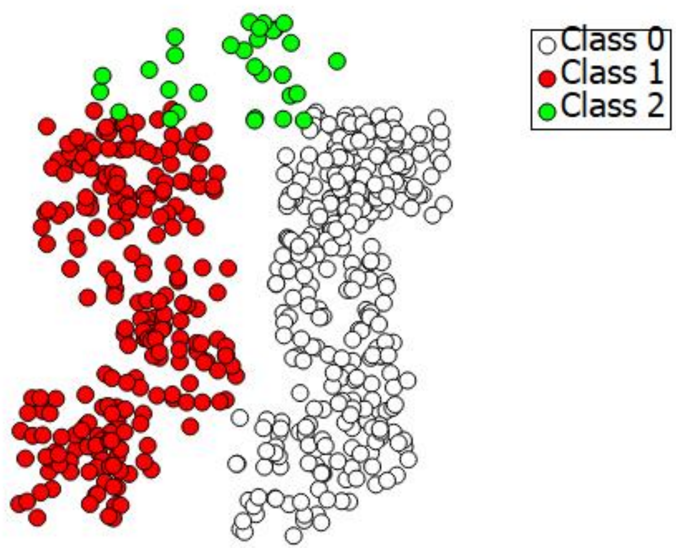
n_{ij} → Number of samples that belong to class i and classified at class j

Confusion matrices at MLDemos are color-coded



Confusion matrices

To detect effect of unbalanced classes in multi-class classification

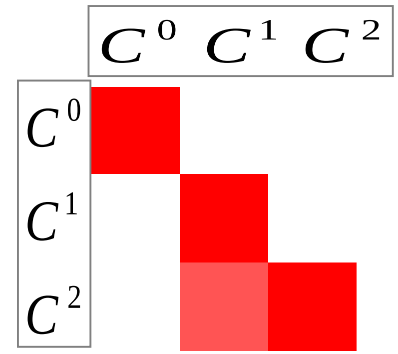


Class 2 has much less datapoints than class 0 and class 1

K-NN

Classification error: 9%

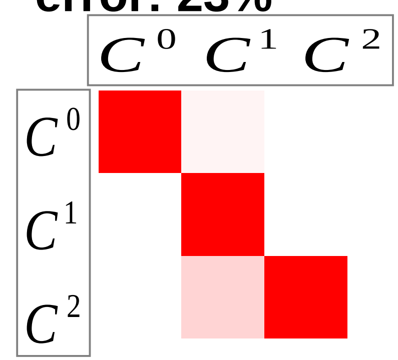
Confusion Matrix:



GMM

Classification error: 23%

Confusion Matrix:



Summary

To evaluate performance at classification, one should:

- Perform crossvalidation
- Use the F-measure on training and test sets
- Use the ROC curve to determine optimal choice of hyperparameters
- Use the confusion matrix to determine if some classes are poorly estimated

It is further important to check:

- Sensitivity of performance to choice of training/testing ratio
- Poor performance due to unbalanced classes
- Overfitting
- Tradeoff between computational costs and performance