Assignment report,

Machine Learning

MSc Applied Bioinformatics,

Marie Schmit

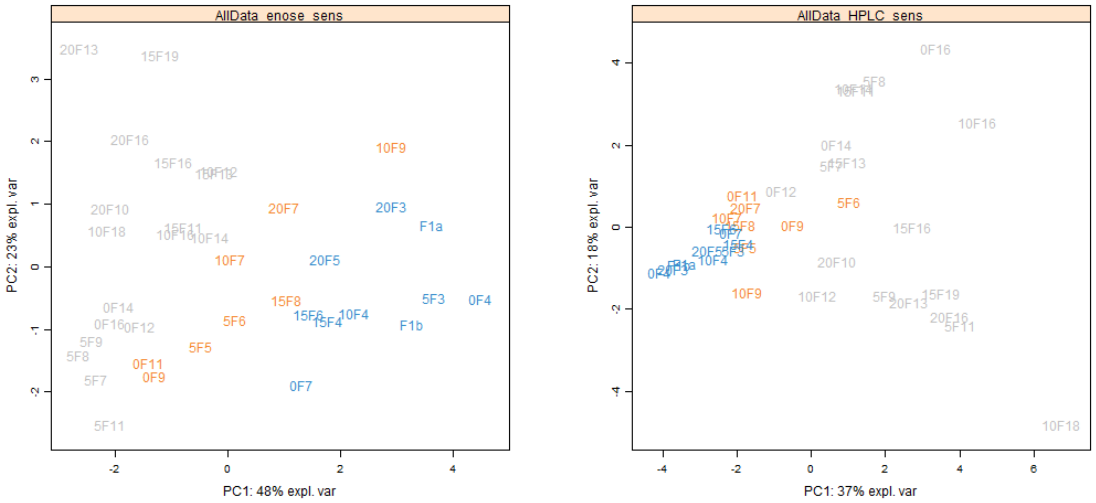
Abstract

Meat freshness determination is a stake in the agri-food and public health fields. This parameter can rapidly be determined via HPLC or enoses analysis methods, coupled with Machine Learning results processing. With testing data like bacteria counts or sensory testing of meat by a panel of experts, models can be trained to detect the freshness of the meat. In this study, classification models (SVM, RF, KNN) were trained with enose and HPLC data to predict the class of the meat (fresh, semi-fresh or rotten). Regression models (RF, KNN, PLS) were trained to detect the correlation between HPLC and enose data and the number of bacteria TVC and Pseudomonads in the meat, directly linked to its freshness. After tuning and comparing those models, it appears that random forest on HPLC data is a very good method for classification, and PLS with enose produced the best results for regression.

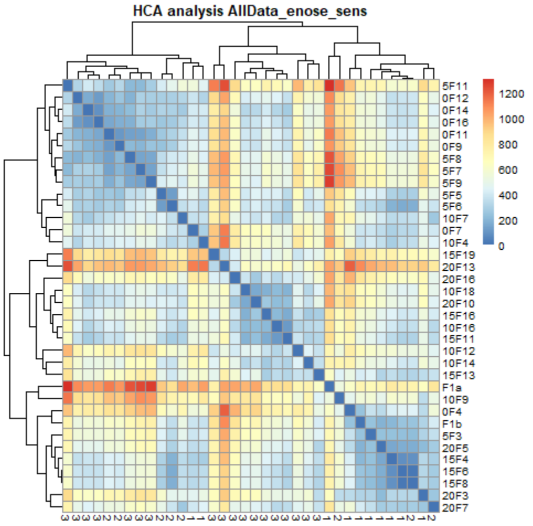
Data exploratory and analysis

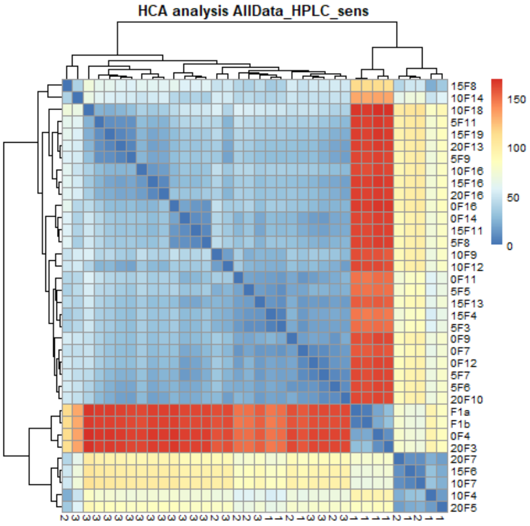
1) PCA was applied to both datasets with the function pca, package MixOmics. Enose samples clusters were in adequation with their sensory scores. Sample 10F9 could be considered as an outlier. HPLC samples were less clearly clustered by samples, but three close groups still emerged. There were some potential outliers like 0F12 or 5F6.

Table 1 PCA scatter plots of enose and HPLC grouped by sensors class



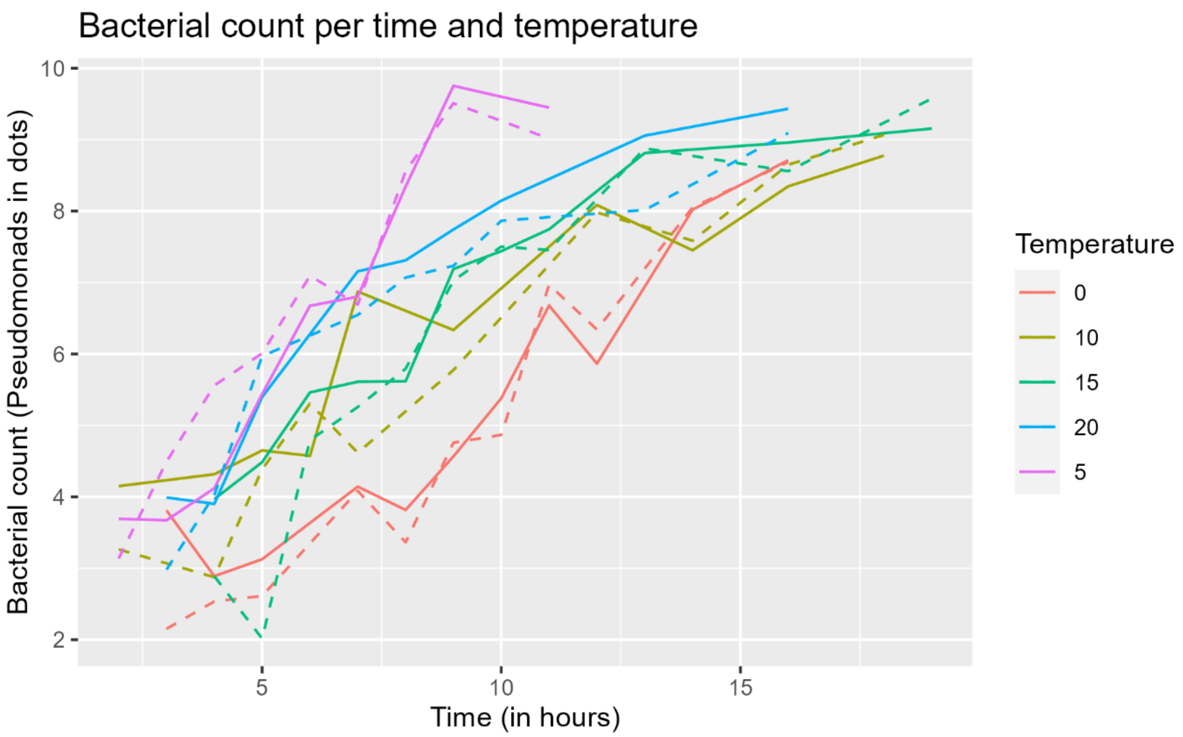
Analytical methods were tried to find better separation. First, PCA plots were displayd with other styles (in 2D, 3D, in a biplot). Then, HCA was used (pheatmap::pheatmap), which produced three cluster. However, they did not correspond to their sensory values. For instance, 10F9 was no longer clustered in class 2, but in class 1. The same goes for HPLC. The clustering was better with PCA analysis method.





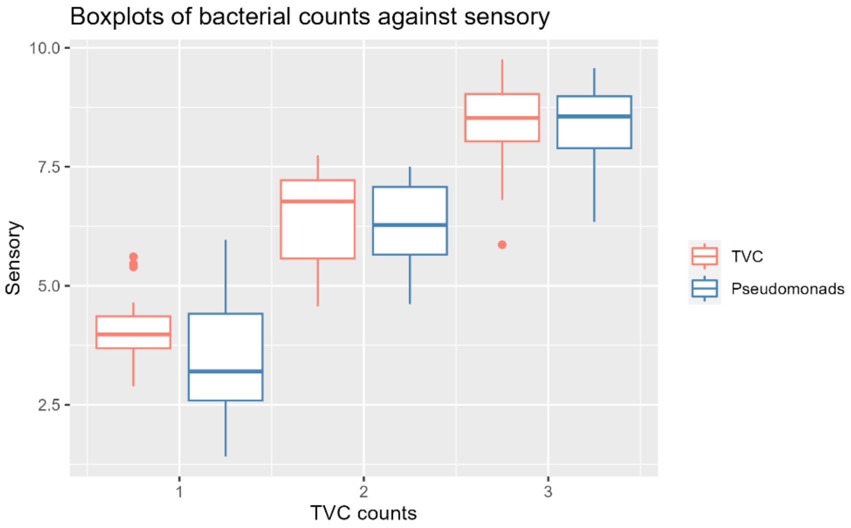
Since potential outliers did not particularly stand out, and considering the small number of data provided, outliers were not removed to avoid errors due to losses of information.

2) The number of bacteria of both types, was increasing with time. The counts were higher for higher temperatures, except for 5 degrees, temperature for which less values were provided. The number of bacteria seemed generally to grow with time and temperatures.



The temperatures did not reach a plateau, which is probably due to a too small number of points.

3) The number of bacteria TVC and Pseudomonias was higher when the sensory score indicated "rotten". Rotted meat had thus more bacteria than fresh one. TVC bacteria were slightly more numerous than pseudomonas. Also, TVC had three outliers.



Time and temperature have both an influence on bacteria growth: as time passed and especially for temperature around 20C, TVC and pseudomonads grow. Those bacteria cause the meat rottenness.

Classification

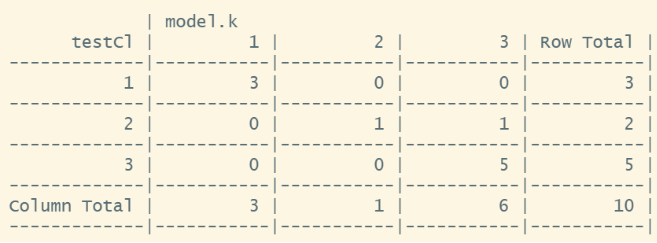
Three different classification methods (k nearest neighbours, random forest, support vector machine) were used to determine to which sensory class belonged enose and HPLC measures. Data was split into a training and a test set for cross validation. Each model was tunned to optimise its hyperparameters, and the results were compared, to established what model was the best for this classification. Data was partitionned using the function createDataPartition from the package caret, that ensured a balanced representation of the train and test sets. This function creates an index of each set. The partition was made reproducible, with the seed set at 8.

Knn classification

A model using k nearest neighbour methods was first trained. The aim was to optimise it by finding the best fit and k parameter. Since knn is a distance-based algorithm, different methods of scaling (center, auto scale and range scale) were tested, to give the features the same weight in distance calculations. Different values of k, the number of nearest neighbours' data taken in the same cluster, were tested, from 1 to 20. The model was trained using the function class::knn, that evaluates the Euclidian distance between nearest neighbours and decides the classification by majority vote (R documentation, knn {case}). The accuracy of the model was evaluated for each of those tested hyperparameter, using the function caret::confusionMatrix, that evaluates the cross-tabulation of predicted and observed data (R documentation, confusionMatrix {caret}). Only the hyperparameters leading to the best accuracy were kept.

For HPLC data with one iteration, the best accuracy without scaling was 0.9 for k =7 (different k give the same accuracy, k=7 was chosen).

Table 2 Cross table for HPLC data, 1 iteration, k = 7, no scaling



```
             | model.k 
      testCl |         1 |         2 |         3 | Row Total | 
-------------|-----------|-----------|-----------|-----------|
           1 |         3 |         0 |         0 |         3 | 
-------------|-----------|-----------|-----------|-----------|
           2 |         0 |         1 |         1 |         2 | 
-------------|-----------|-----------|-----------|-----------|
           3 |         0 |         0 |         5 |         5 | 
-------------|-----------|-----------|-----------|-----------|
Column Total |         3 |         1 |         6 |        10 | 
-------------|-----------|-----------|-----------|-----------|
```

This model was trained on 100 iterations. For each iteration, the accuracy value was saved in a list that was later used to calculate the cumulative mean accuracy.

The same process was repeated for enose data. The best accuracy was 1 for non-scale data, k=3. This could indicate an overfitting: a model too much trained on specific values to be performant on new datasets. However, all the accuracies values were not equal to 1. Reducing the number of train with a 5:5 (50% of training data, 50% of test data) partition instead of a 7:3 reduced the accuracy to 0.83, which decreased the overfitting. The first partition of 7:3 was used to calculate the cumulative mean value for 100 iteration, k=3 and no scaling, to evaluate the overfitting issue with a larger number of trains. The mean accuracy for 100 iterations was 0.791. Even if the best accuracy value was still 1 (which could be due to the small number of trained values), this cumulative mean accuracy was not indicating a problematic overfitting, so the model was kept.

Svm classification

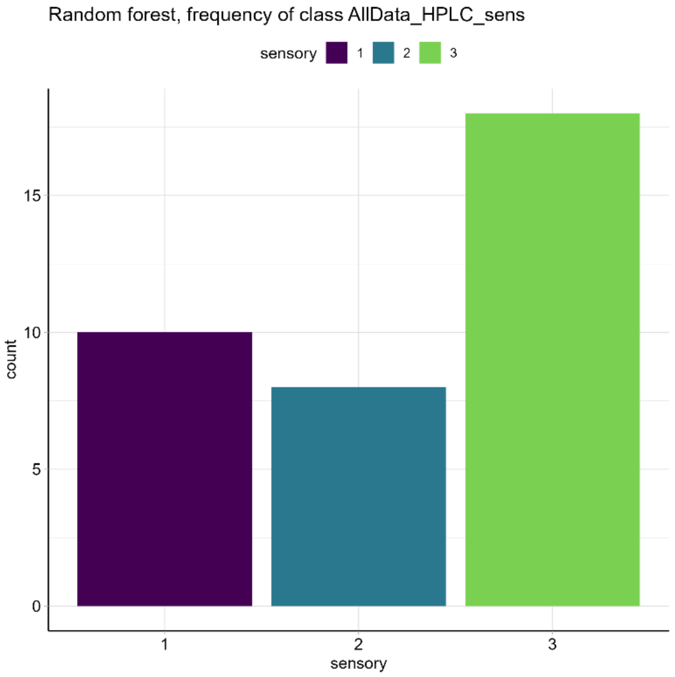
The svm-rd classification method was applied (kernlab::ksvm) using the same partition and predictor sensory. To tun the model, different kernels were tested: 'vanilladot' the linear kernel, 'polydot' the polynomial kernel, 'rbfdot' the Gaussian kernel , 'tanhdot' the hyperbolic tangent kernel (R documentation, dots {kernlab}). The best model was like for knn the one leading to the best accuracy.

For HPLC data, the best accuracy was 0.7 for the kernel “rbfdot” (the Gaussian kernel). For this model, the cumulative mean accuracy over 100 iterations was 0.789. With the same best kernel for enose, the best accuracy was 0.9. The cumulative accuracy for 100 iterations was 0.789.

Random forest

For random forest classification, a task was defined, with the predictor sensory, allowing to check the number of elements in each sensory class. Most of the data belonged to class 3 (rotten meat) for HPLC.

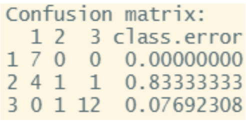
Table 3 Frequency plot, data HPLC, sensory classification



A learner was set for random forest (mlr3::lrn). The hyperparameters ntree, the number of trees in the forest; mtry, the number of features to sample at each node; maxnodes, the maximal number of leaves; nodesize, the minimum number of cases authorised in a terminal node were tuned with grid search (mlr3tuning::tnr). The number of features randomly sampled at each nodes had to be lesser than the total number of features, but still large enough to analyse trends of the data. The nodesize had to be small enough to avoid a too big tree but large enough to avoid underfitting. The methods of evaluation were resampling strategy cross validation (mlr3::rsmp) and performance measurement of classifier accuracy (mlr3::msr). Since evaluated each hyperparameter is a too heavy process, the tuning was stopped after 20 evaluations (bbotk::term).

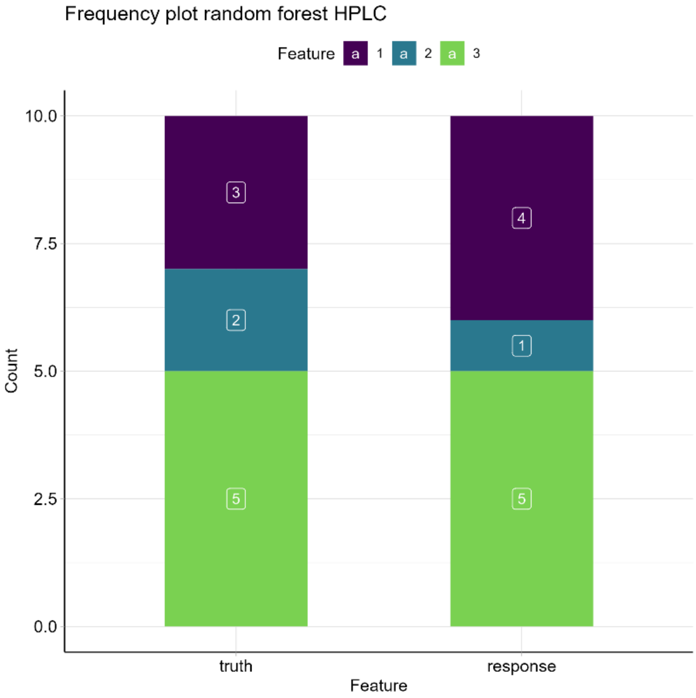
For HPLC, the best random forest parameters were: ntree=200, mtry=2, maxnodes=20, nodesize=2, with an accuracy of 0.84.

Table 4 Confusion matrix for HPLC tuned model



```
Confusion matrix:
  1 2  3 class.error
1 7 0  0  0.00000000
2 4 1  1  0.83333333
3 0 1 12  0.07692308
```

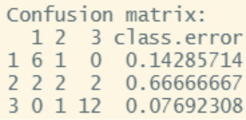
For one iteration, the accuracy of this model was 0.7: one sample of sensory 1 and one of sensory 2 were misclassified.



For 100 iterations, the mean accuracy of this model was 0.813.

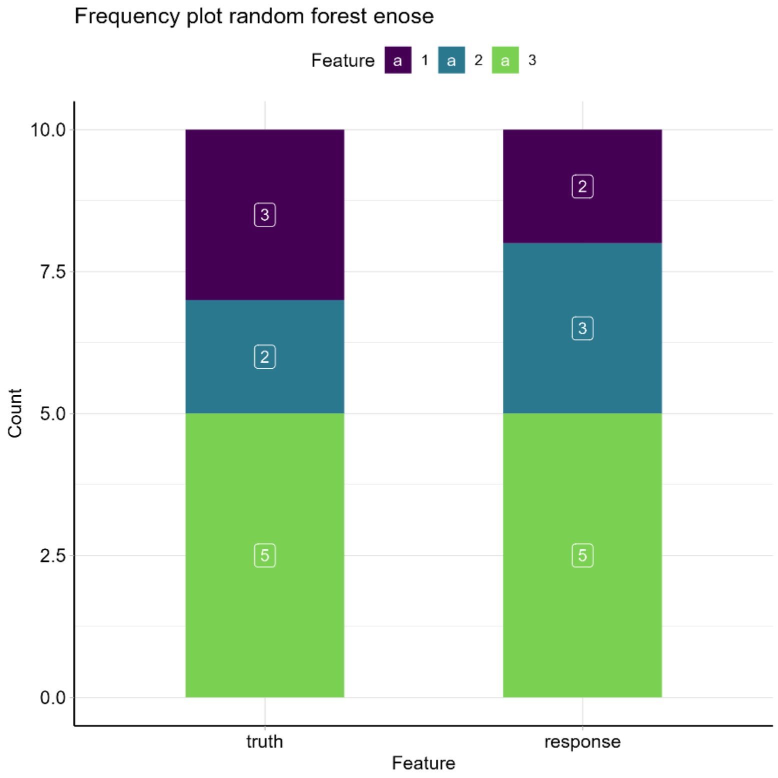
The same steps were repeated for enose data set. The maximal number of features to sample for each node was set to 8 since enose has 8 sensors. The best model had 200 trees, 2 features to sample per node, 20 leaves maximum, 2 cases allowed in terminal node. The accuracy for this model was 0.731 for one iteration.

Table 5 Confusion matrix for one iteration for tunned model, enose data



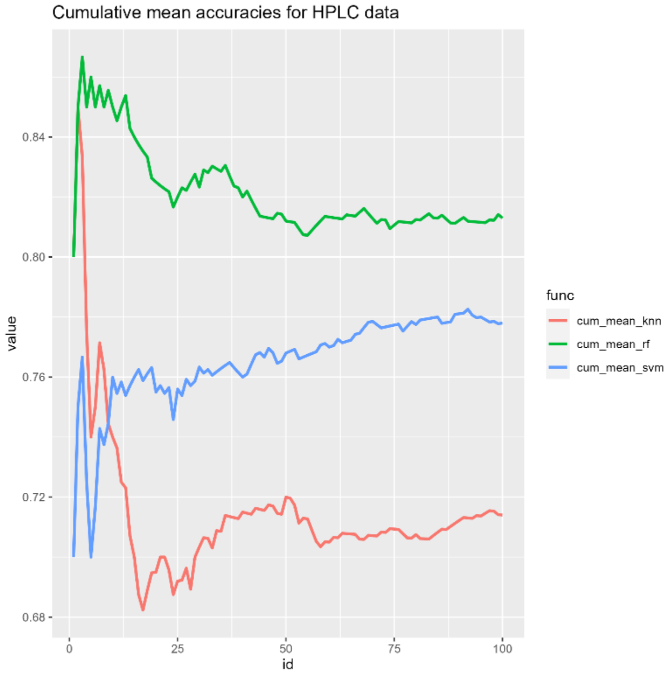
```
Confusion matrix:
  1 2  3 class.error
1 6 1  0  0.14285714
2 2 2  2  0.66666667
3 0 1 12  0.07692308
```

Two elements were misclassified: one for sample one, the other for sample two.



For 100 iterations, the cumulative mean was 0.745.

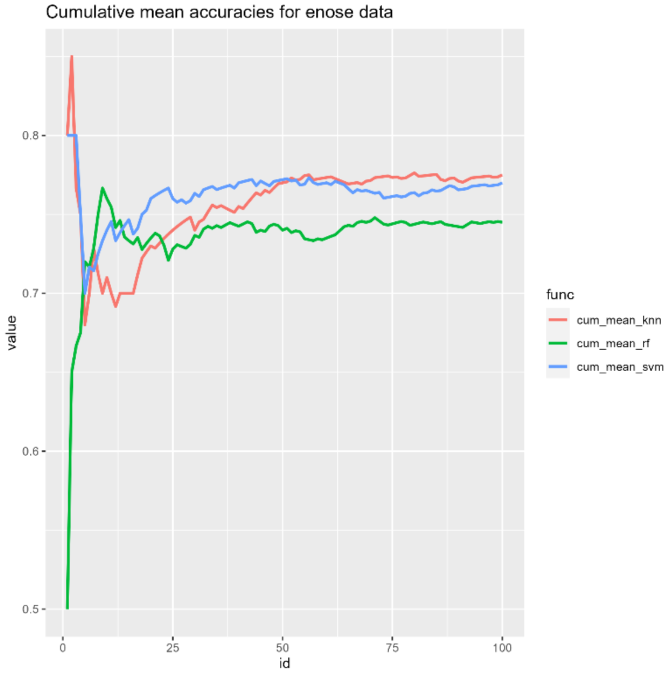
1. The cumulative mean accuracies were calculated for every dataset (enose and HPLC).



2.

For HPLC data, random forest had the higher cumulative mean.

Dataset HPLC	Method	Cumulative mean accuracy after 100 iterations
	Knn	0.714
	Svm	0.789
	Random forest	**0.813**

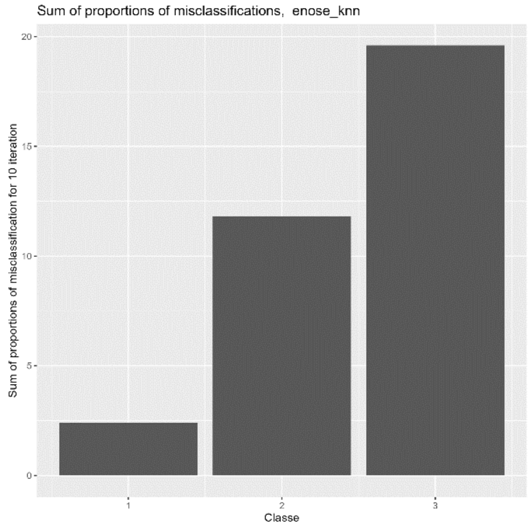


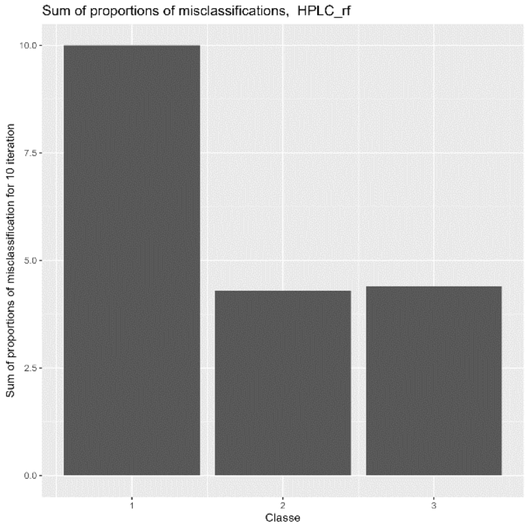
For enose data, knn had the higher cumulative mean accuracy.

Dataset enose	Method	Cumulative mean accuracy after 100 iterations
	Knn	0.791
	Svm	**0.789**
	Random forest	0.745

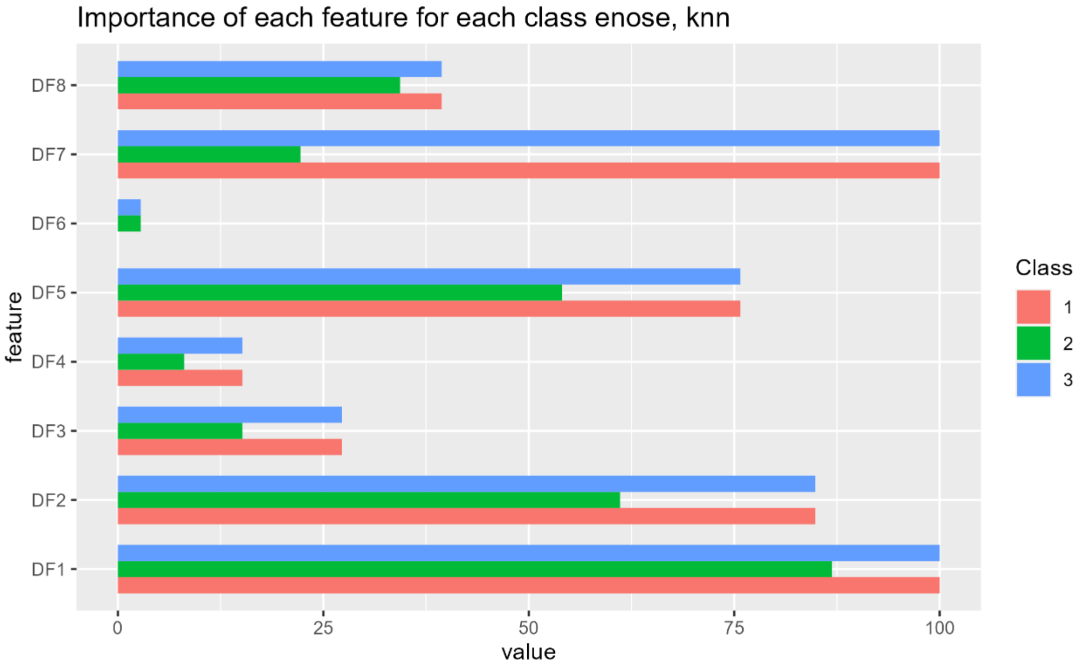
3. For each algorithm, we the number of misclassified (false positive or false negative number) samples was calculated for each iteration (gmodels::CrossTable), then summed up. The sum of the proportions was calculated for 100 iterations (see all the plots in folder “Plots”).

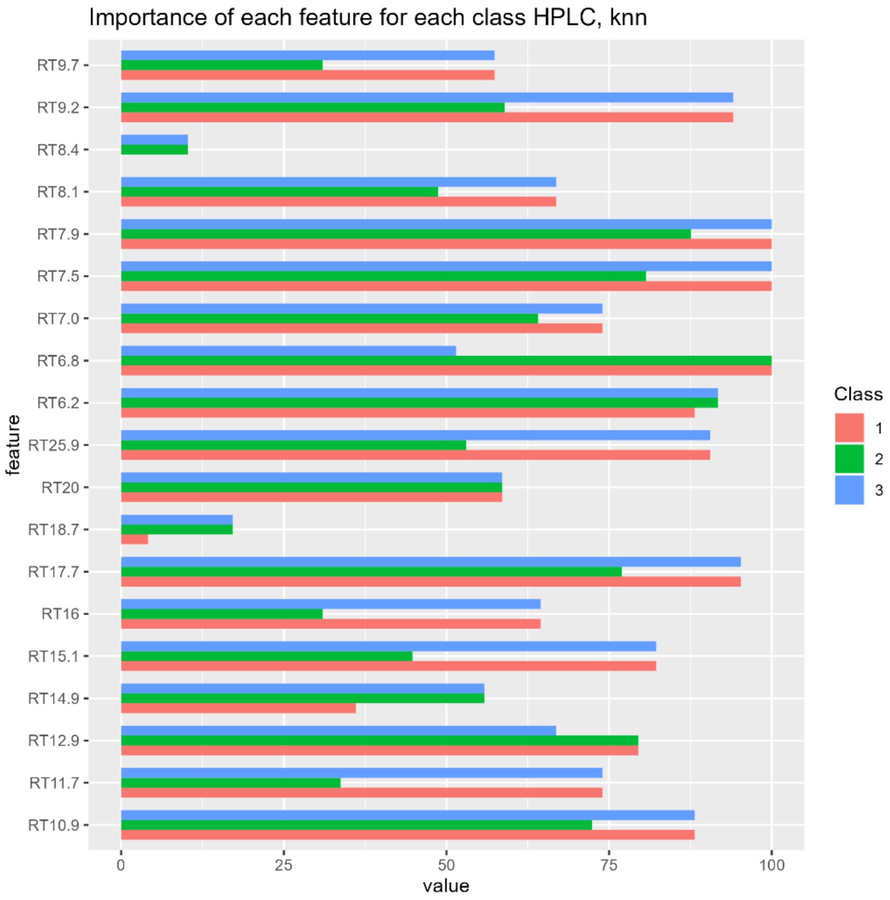
For enose, rf and knn had the smallest proportions of misclassification, with almost no misclassifications (2.5 in total proportion) for the class 1 for knn (the “true” class, corresponding to fresh meat). For HPLC, rf had the smallest numbers for all the classes (with maximum 10% of missclassifications for class 1), but svm had the smallest number of misclassifications sum of proportion (approximatively 2).



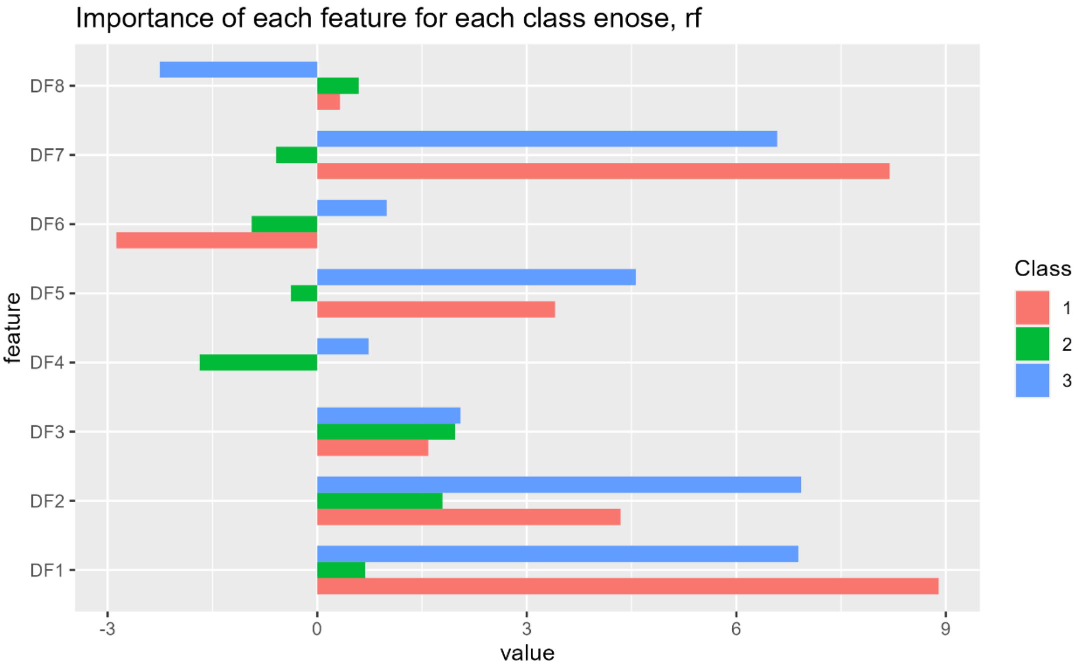


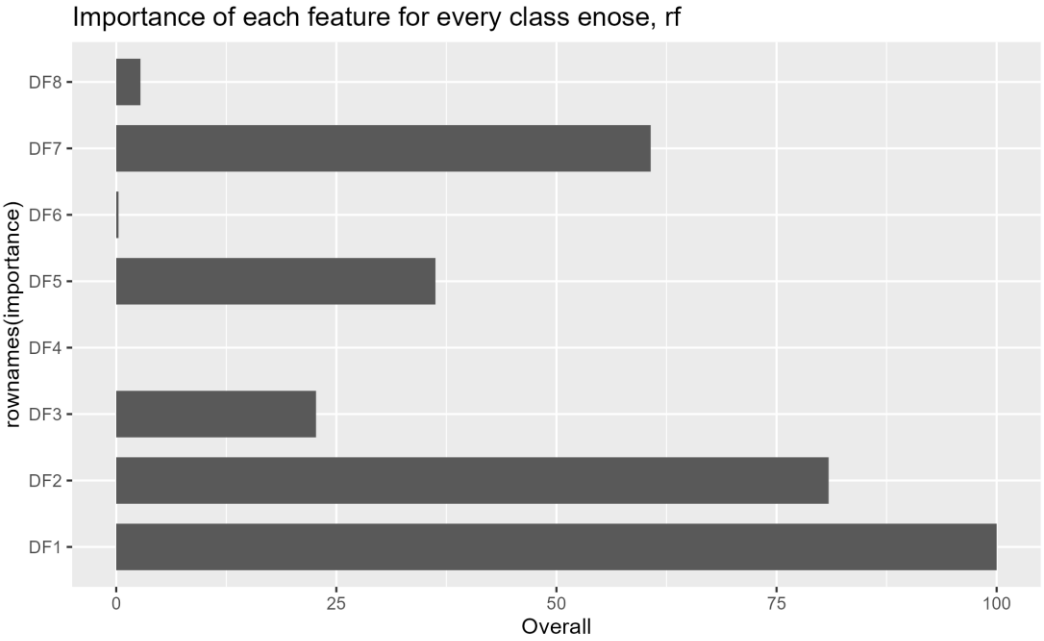
4. To find the importance of every variable for the classification, a model was trained for knn (carety::train). The model was tuned using a grid search of k from 1 to 20. For knn, the importance (caret::varImp) of all the variables was positive. The same goes for svm, with the kernel rbfdot. The results are the same as knn's because no importance scores are defined to differentiate the models.

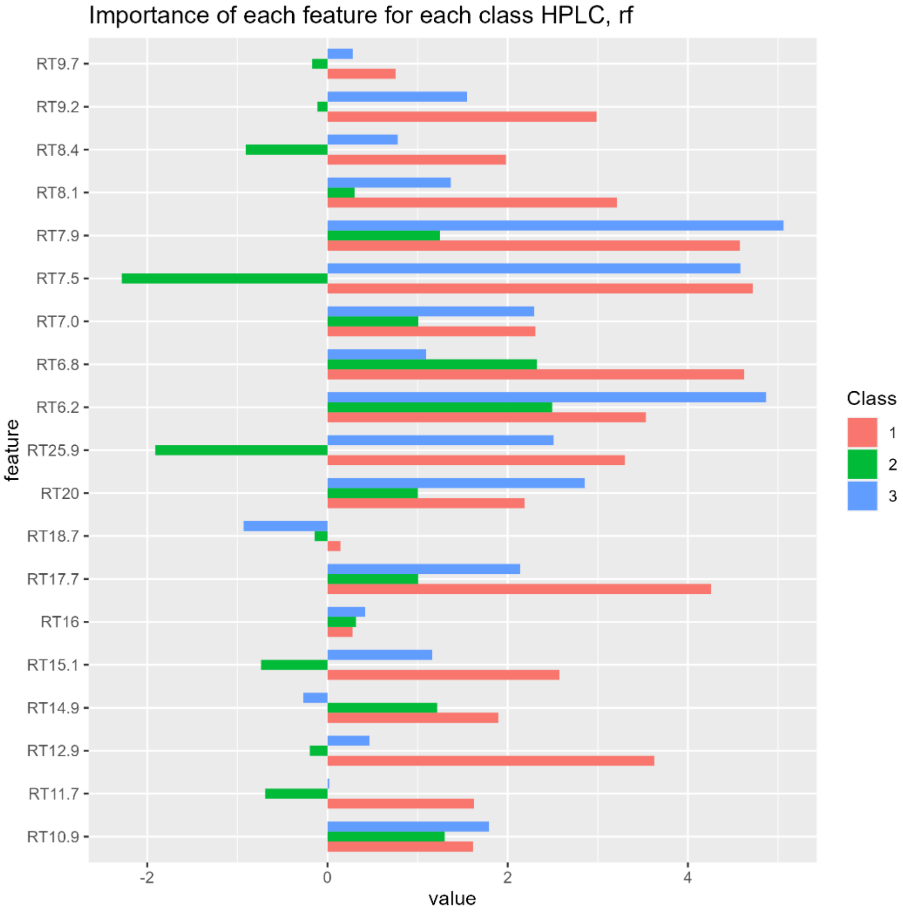


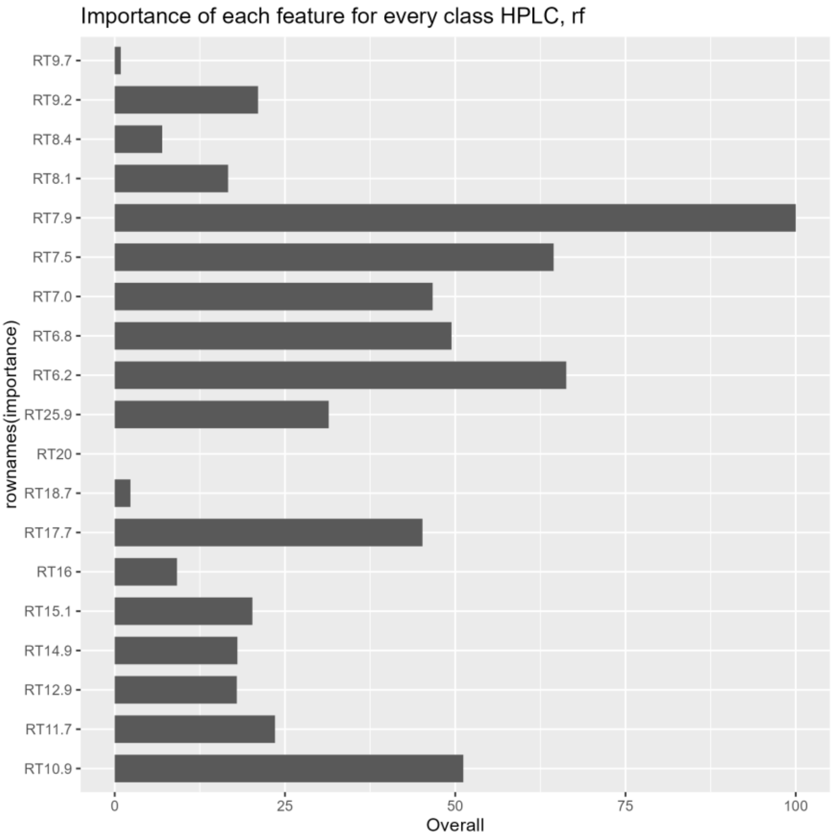


The model random forest was trained with randomForest::randomForest the importances for each class assessed with caret::varImp. Overall importances are calculated for all classes with caret::train.




Importance of each feature for every class enose, rf
rownames(importance)
DF8
DF7
DF6
DF5
DF4
DF3
DF2
DF1
0
25
50
75
100
Overall


Importance of each feature for each class HPLC, rf
feature
RT9.7
RT9.2
RT8.4
RT8.1
RT7.9
RT7.5
RT7.0
RT6.8
RT6.2
RT25.9
RT20
RT18.7
RT17.7
RT16
RT15.1
RT14.9
RT12.9
RT11.7
RT10.9
-2
0
2
4
value
Class
1
2
3



Enose data importances summary

Model	Minimum or maximum importances	Class 1	Class 2	Class 3
Knn	Max importance	DF7 (100/100) DF1 (100/100)	DF1 (82/100)	DF7 (100/100) DF1 (100/100)
	Min importance	DF6 (0/100)	DF6 (2/100)	DF6 (2/100)
rf	Max importance	DF1 (8.9/9)	DF2, DF3 (5.5/9)	DF1, DF2, DF7 (6.5/100)
	Min importance	DF6 (-2.9/9)	DF4 (-2.1/9)	DF8 (-2.5/9)
rf	Max importance	DF1 (100/100) DF2 (80/100)		
	Min importance	DF4 (0/100) DF6 (1 /100)		

For enose data, variable DF1, DF2 and DF7 play a major part in every model, while DF6 and DF4 have particularly poor importances for all the classes.

HPLC data importances summary

Model	Minimum or maximum importances	Class 1	Class 2	Class 3
Knn, svm	Max importances	RT7.9, RT7.5, RT6.8 (100/100)	RT6.8 (100/100)	RT7.9, RT7.5 (100/100)

	Min importances	RT18.7 (5/100)	RT8.4 (10/100)	RT8.4 (10/100)
rf	Max importances	RT7.5, RT6.8 (4.3/4.6)	RT6.2, RT6.8 (2.2/4.6)	RT7.9 (4.5/4.6), RT6.2 (4.4/4.6)
	Min importances	RT18.7 (0.1/4.6)	RT7.5 (-2.2/4.6)	RT18.7 (-0.9/4.6)
rf	Max importance	RT7.9 (100/100)		
	Min importances	RT20 (0/100)		

For HPLC, RT7.9 has a strongly positive influence on many classes and models, while RT18.7 and RT20 has a poor or negative influence.

3. The mean accuracy for all classifications methods were higher for the analytical platform enose (0.775) than for HPLC (0.772). However, the best method for HPLC classification, rf, had a higher accuracy (0.813) than knn, the best method for classification with enose data (0.791). The goal was to minimise misclassifications of fresh meat, to ensure the perfect safety of the food. For enose, the smaller average misclassification mean was for svm (8.16) but knn had less misclassifications for class C1 (1). For HCLP, rf had the smallest number of misclassifications (6.33), but more misclassified data for class 1 (10) than svm (0.5). The best mean of classifications is rf for HCLP, which corresponded to the best parameters for accuracy: this model might be the best. The same variables (DF1, DF2, DF7) had the most importance for enose classification, while the importances were more diverse between rf and knn, svm for HPLC data. If rf for HPLC were chosen, RT7.9 would be the most important variable.

Regression

Three regression methods were studied (knn, RF, PLS-R) to establish a relationship between the predictor (the data from enose and HPLC) and a predicted number of bacteria, an indicator of freshness. The aim was to determine the best regression method.

For knn, the function caret::train was used for the model tuning, with the option tuneGrid that conducted a grid search to find optimal hyperparameters. Here, the hyperparameter of interest was k, tested from 1 to 20. The pre-processing centering and scaling methods were applied to the dataset. The RMSE (Root Mean Square Error) values with and without pre-processing of the data were compared. The lower the RMSE, the lower the average distance between the predicted and observed values, and the better the model.

Knn tuning parameters

Analytical platform	Bacterial type	Model	Preprocessing (Scaling centering)	Best hyperparameter
HPLC	TVC	knn	Yes	K=3
	Pseudomonads		Yes	K=5
enose	TVC	knn	Yes	K=5
	Pseudomonads		Yes	K=6

Random forest

The model was tuned the same way with the model rf (library random forest) and a grid search on parameter mtry (the number of features to sample at each node).

rf tuning parameters

Analytical platform	Bacterial type	Model	Preprocessing (scaling, centering)	Best hyperparameter
HPLC	TVC	rf	Yes	Mtry = 18
	Pseudomonads		Yes	Mtry = 16
enose	TVC	rf	Yes	Mtry = 8
	Pseudomonads		Yes	Mtry = 9

Partial Least Squares

The model was trained identically (caret::train) with method “pls” from pls package. The hyperparameter to tune was ncomp, the numbers of components of the model (6 Availables models, n.d.). The maximal number of components for tuning was set to the maximal number of variables.

PLS-R tuning parameters

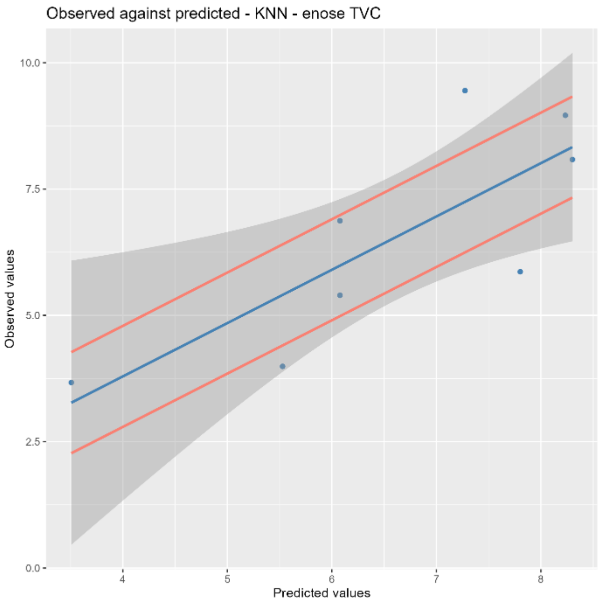
Analytical platform	Bacterial type	Model	Preprocessing (scaling, centering)	Best hyperparameter
HPLC	TVC	pls	No	Ncomp = 8
	Pseudomonads		No	Ncomp = 8
enose	TVC	pls	No	Ncomp = 9

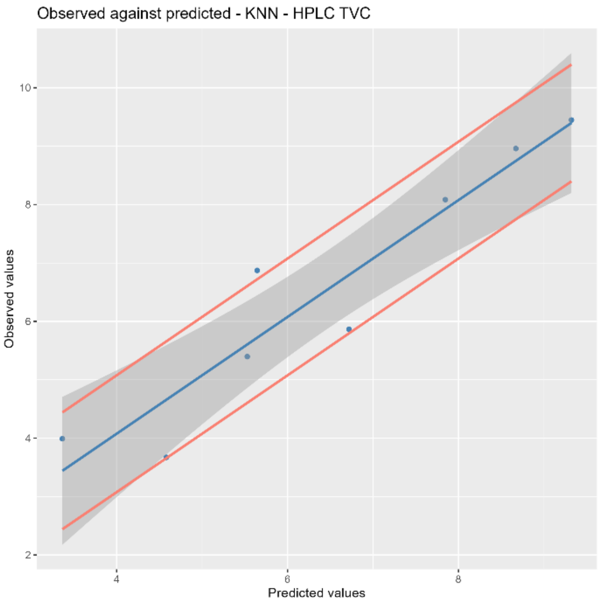
	Pseudomonads		No	Ncomp = 9

2. The previous tunned models were trained and evaluated for 100 iterations. 95% CI was calculated with Rmisc::CI.

Analytical Platform	Bacterial Type	Model	RMSE	STDV	95% CI
Enose	TVC	k-NN	1.0580	0.3036	[1.1183; 0.9978]
		RF	1.0668	0.2940	[1.1251; 1.0084]
		PLS-R	**0.8631**	**0.3521**	**[0.9330; 0.7933]**
	Pseudomonas	k-NN	1.3577	0.2465	[1.4066; 1.3088]
		RF	1.3102	0.2389	[1.3576; 1.2628]
		PLS-R	**1.1122**	**0.3963**	**[1.1908; 1.0336]**
HPLC	TVC	k-NN	0.8910	0.2519	[0.9410;0.8411]
		RF	0.8865	0.1996	[0.9261; 0.8469]
		PLS-R	**0.8474**	**0.2906**	**[0.9050; 0.7897]**
	Pseudomonas	k-NN	1.2266	0.2889	[1.2840; 1.1693]
		RF	**1.1793**	**0.2824**	**[1.2354; 1.1233]**
		PLS-R	1.2567	0.3637	[1.3288; 1.1845]

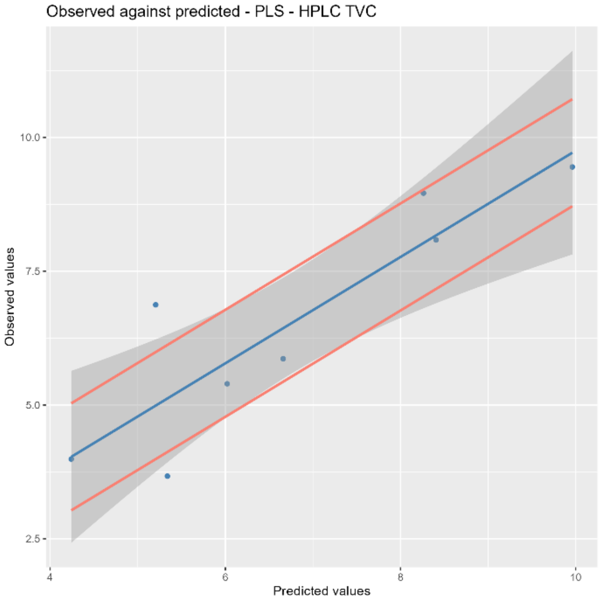
2. The observed and predicted values of each regression model, for each analytical platform and type of bacteria were saved for the last iteration. A LOESS smooth line was added to the plot (ggplot2:smooth_geom) with a linear model.

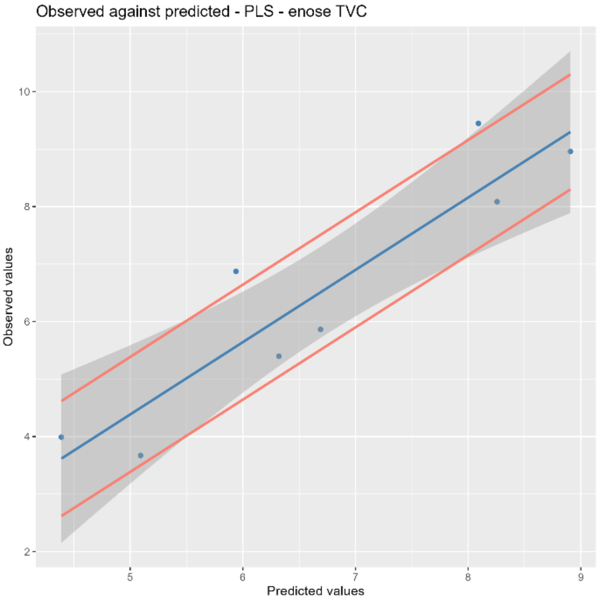




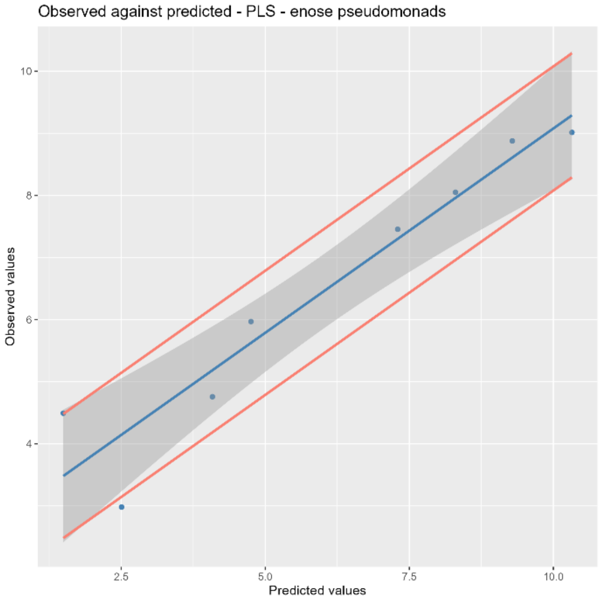
3. For TVC, the smallest standard deviation (data points close to the mean) for TVC was 0.1996, with HPLC, rf. The lowest RMSE values were PLS for enose (0.8631) and PLS for HPLC which was better (0.8474). They also had the smallest 95% confidence interval, respectively [0.9330;0.7933] and

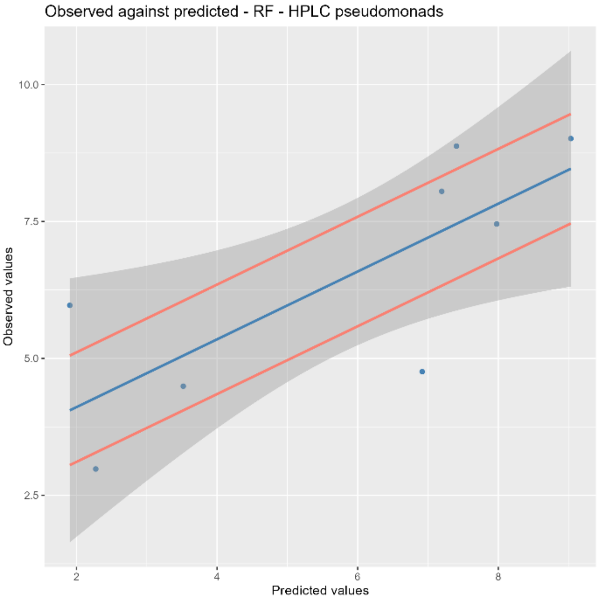
[0.9050;0.7897]. Between enose and HPLC PLS, the best linear model with data between -1 and +1log of values was enose. Since the differences of RMSE values between the two models were very small, enose PLS was a good model for TVC.


Observed against predicted - PLS - HPLC TVC


Observed against predicted - PLS - enose TVC

For pseudomonads, enose, PLS had the smallest RMSE mean and 95CI (1.1122, [1.1908; 1.0336]). The values were higher for HPLC, RF (1.1793, [1.2354;1.1233]), but its standard deviation was smallest: 0.2824. The linear model of enose, PLS was much better, within +1 and -1log of values. Since its RMSE mean and 95CI were also smaller than HPLC, RF, this model is better.


Observed against predicted - PLS - enose pseudomonads


Observed against predicted - RF - HPLC pseudomonads

Bibliography

6 Availables models. (n.d.). Retrieved from Github, topepo, caret: https://topepo.github.io/caret/available-models.html

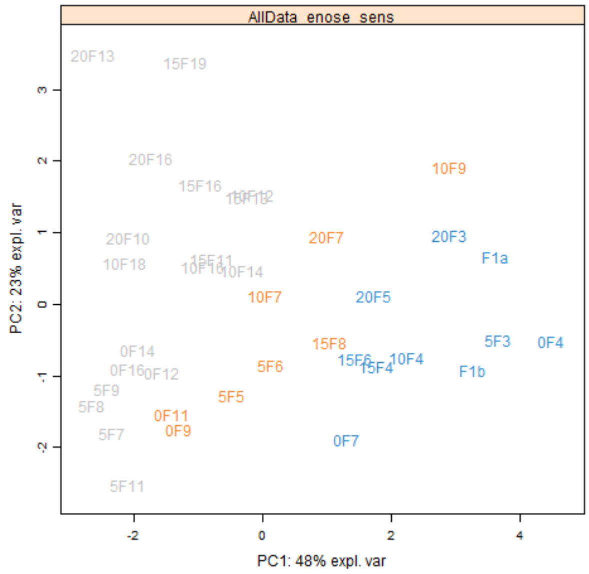
Hadley Wickham, W. C. (n.d.). *Smoothed conditional means*. Retrieved from ggplot2: https://ggplot2.tidyverse.org/reference/geom_smooth.html

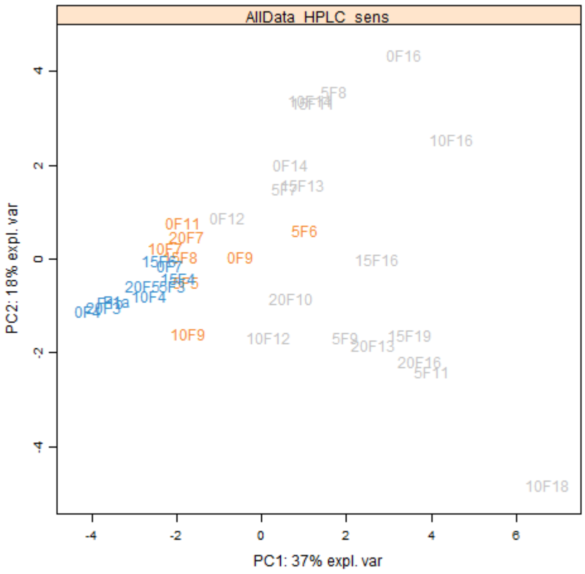
R documentation, confusionMatrix {caret}. (n.d.).

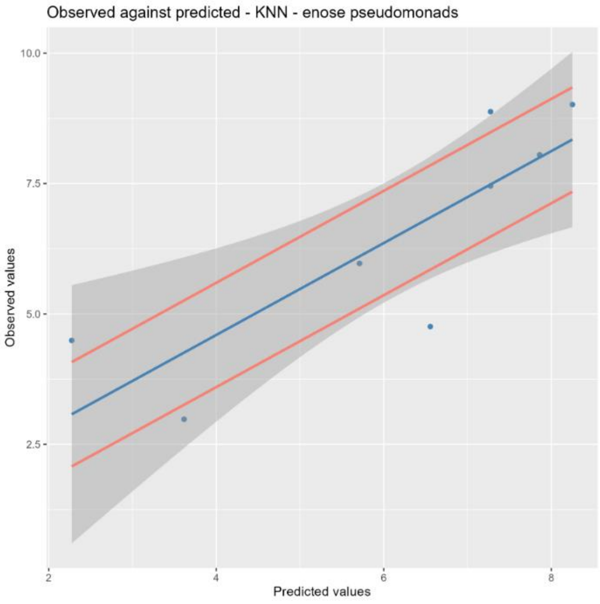
R documentation, dots {kernlab}. (n.d.). *Kernel functions*.

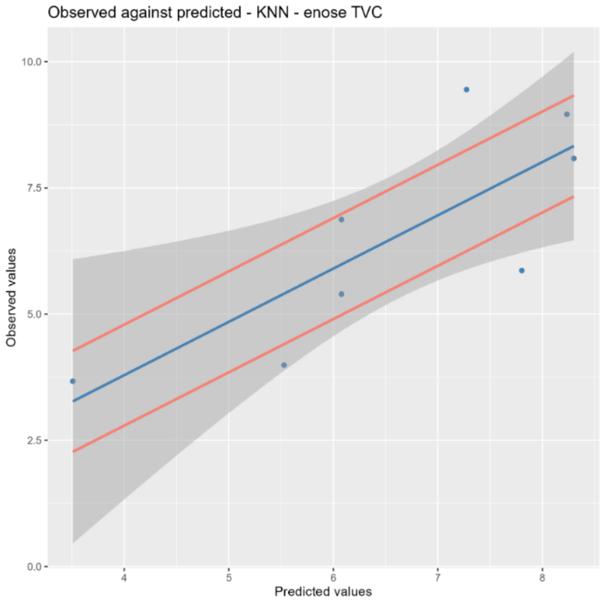
R documentation, knn {case}. (n.d.).

Appendix

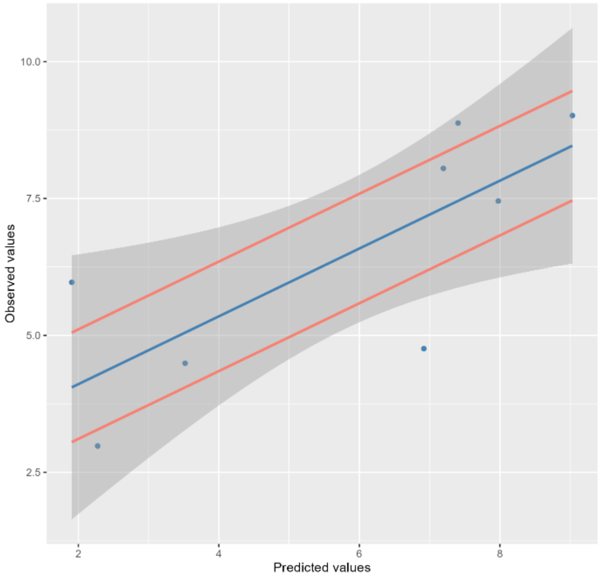

AllData_enose_sens
PC2: 23% expl. var
PC1: 48% expl. var
3
2
1
0
-1
-2
-2
0
2
4
20F13
15F19
20F16
10F9
15F16
10F12
20F10
20F7
20F3
F1a
10F18
15F11
10F16
10F14
10F7
20F5
0F14
15F8
5F3
0F4
0F16
0F12
5F6
15F6
15F4
10F4
F1b
5F9
5F8
5F5
0F11
0F9
5F7
0F7
5F11


AllData_HPLC_sens
PC2: 18% expl. var
PC1: 37% expl. var
4
2
0
-2
-4
-4
-2
0
2
4
6
0F16
5F8
10F11
10F16
0F14
5F7
15F13
0F11
0F12
20F7
5F6
10F7
0F7
0F9
15F16
15F4
20F5
10F4
0F4
F1a
20F10
10F9
10F12
5F9
20F13
15F19
20F16
5F11
10F18

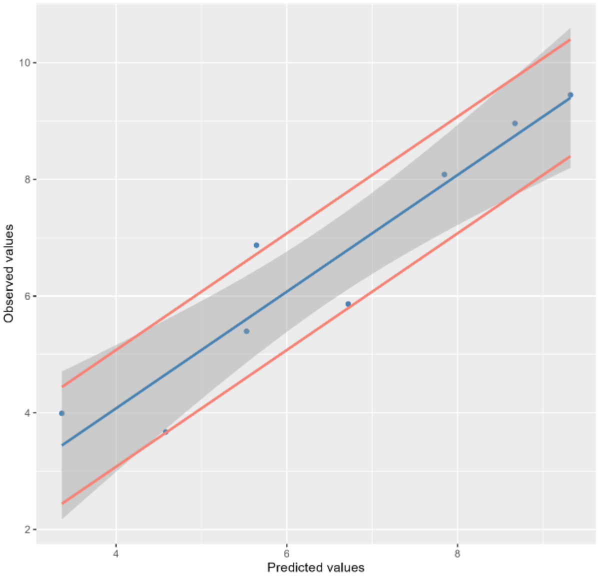

Observed against predicted - KNN - enose pseudomonads
Observed values
Predicted values
10.0
7.5
5.0
2.5
2
4
6
8


Observed against predicted - KNN - enose TVC
Observed values
Predicted values
10.0
7.5
5.0
2.5
0.0
4
5
6
7
8

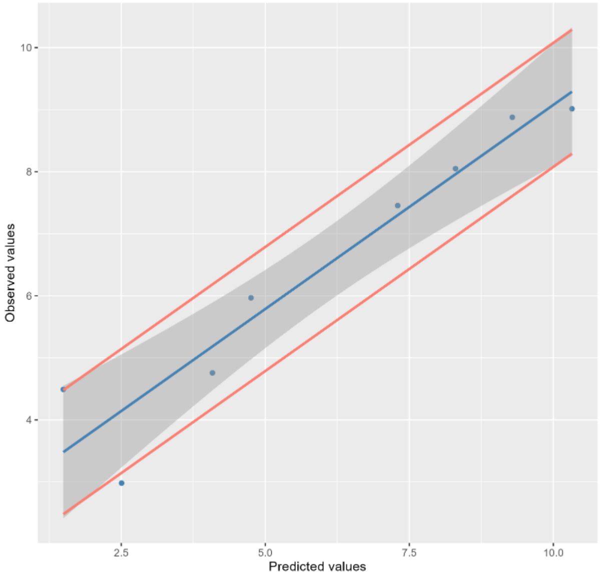
Observed against predicted - KNN - HPLC pseudomonads



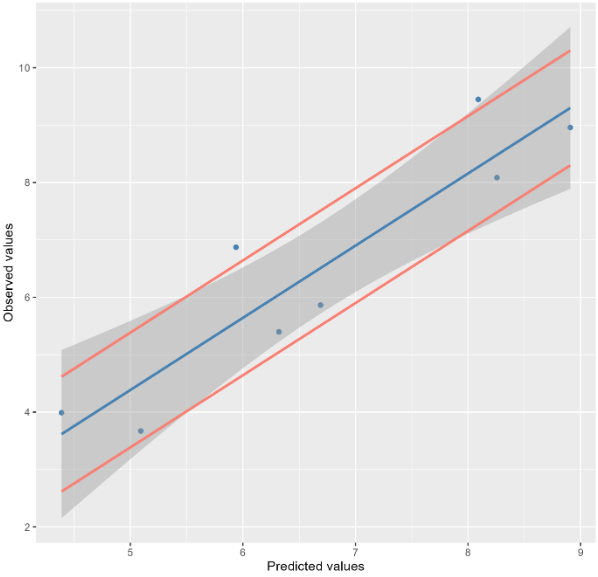
Observed against predicted - KNN - HPLC TVC



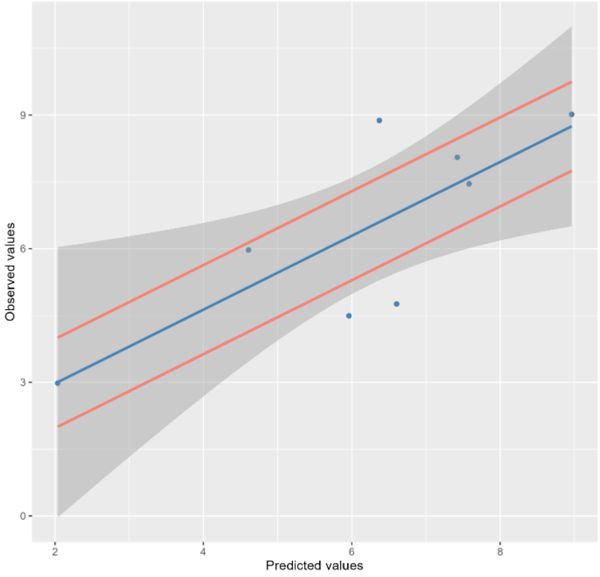
Observed against predicted - PLS - enose pseudomonads



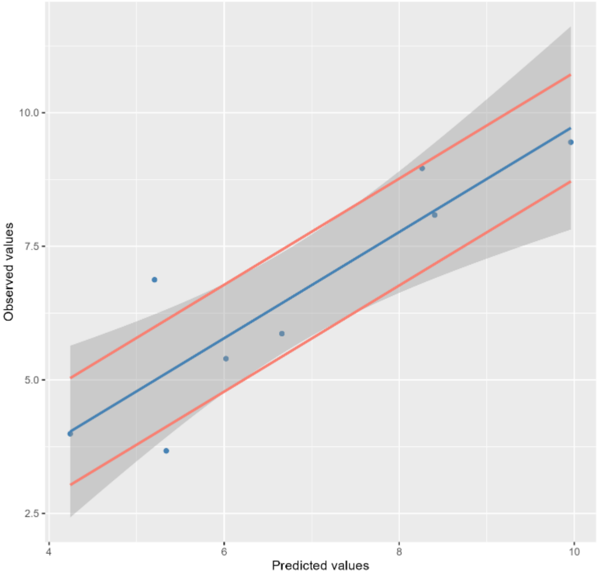
Observed against predicted - PLS - enose TVC

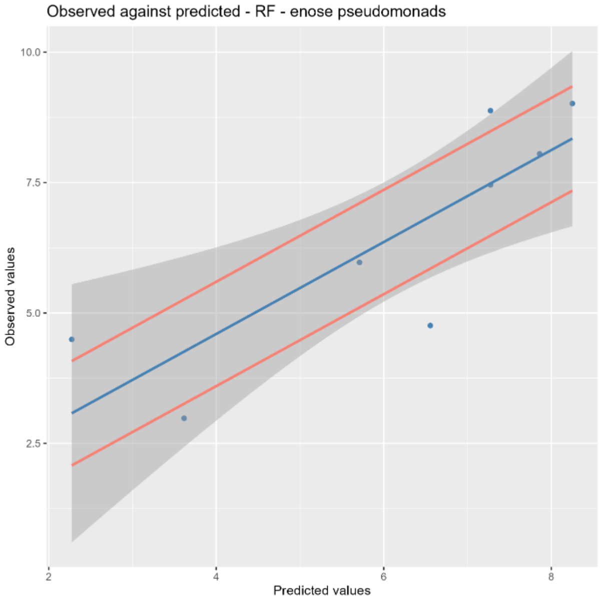


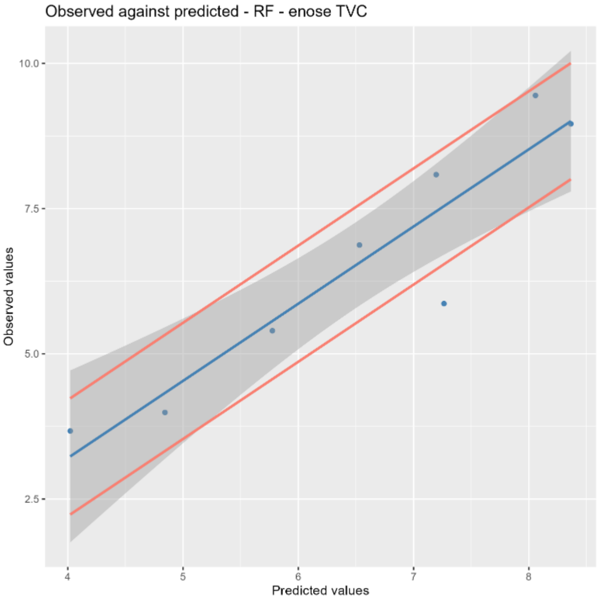
Observed against predicted - PLS - HPLC pseudomonads

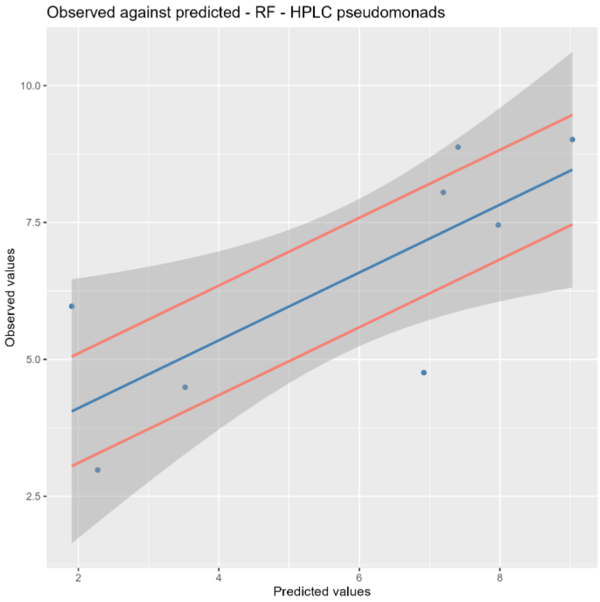


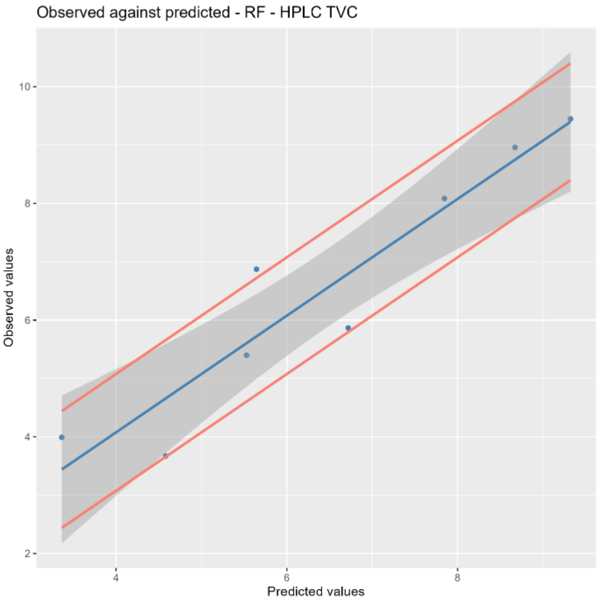
Observed against predicted - PLS - HPLC TVC

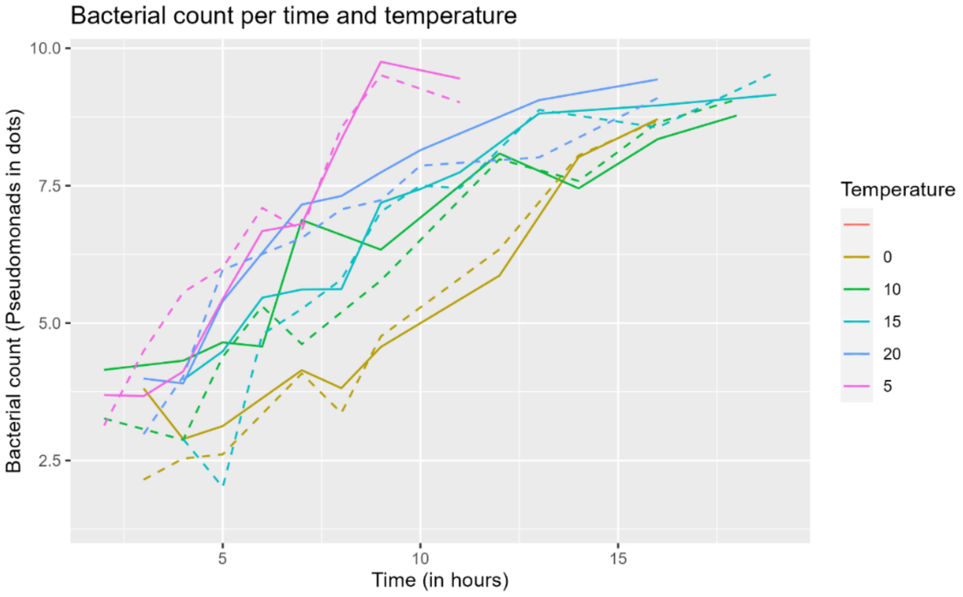



Observed against predicted - RF - enose pseudomonads
Observed values
10.0
7.5
5.0
2.5
Predicted values
2
4
6
8

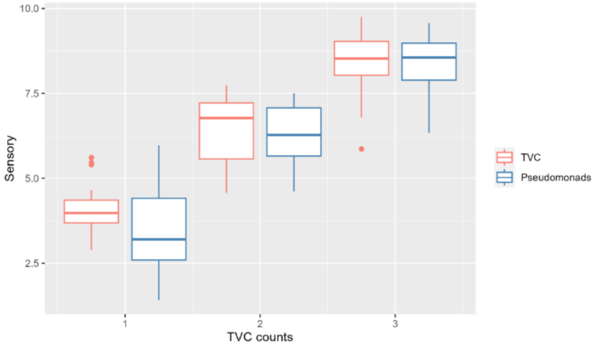

Observed against predicted - RF - enose TVC
Observed values
10.0
7.5
5.0
2.5
Predicted values
4
5
6
7
8

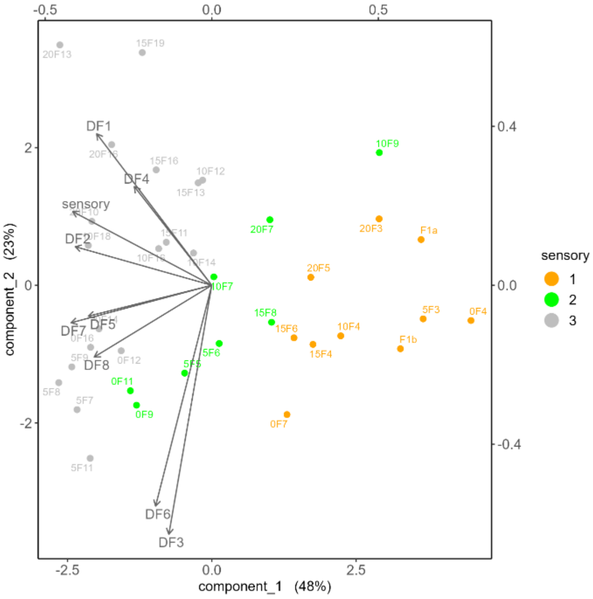

Observed against predicted - RF - HPLC pseudomonads
Observed values
10.0
7.5
5.0
2.5
Predicted values
2
4
6
8

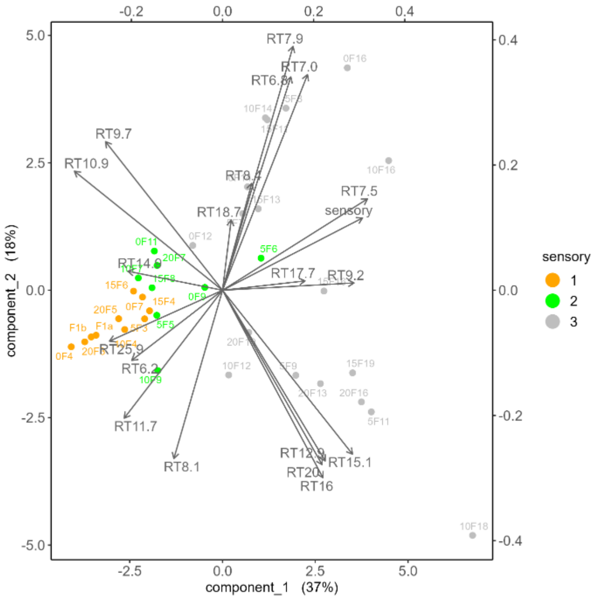

Observed against predicted - RF - HPLC TVC
Observed values
10
8
6
4
2
Predicted values
4
6
8


Bacterial count per time and temperature
Bacterial count (Pseudomonads in dots)
10.0
7.5
5.0
2.5
Time (in hours)
5
10
15
Temperature
0
10
15
20
5

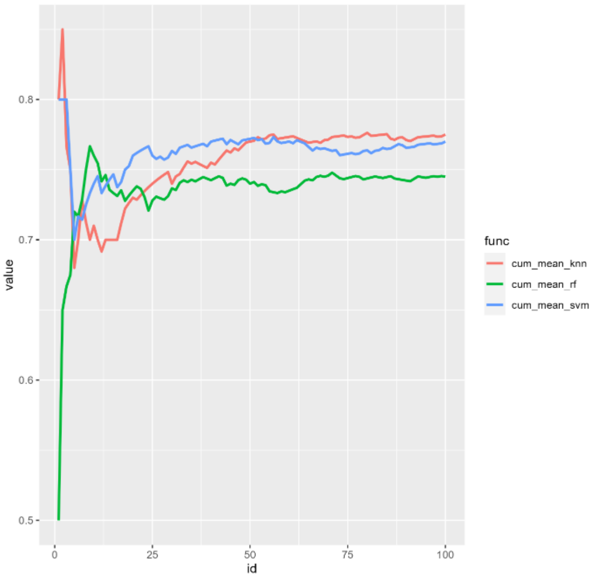
Boxplots of bacterial counts against sensory



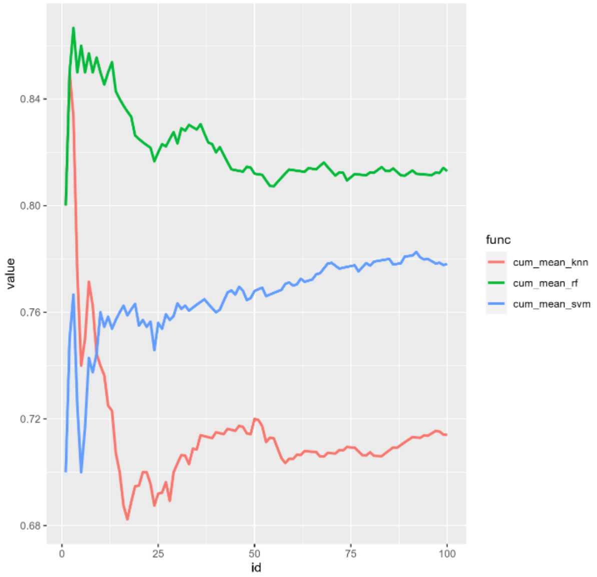




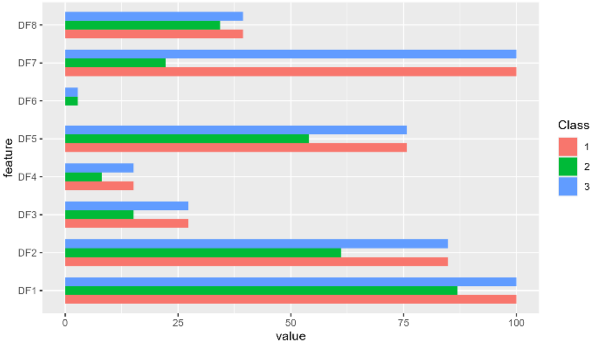
Cumulative mean accuracies for enose data



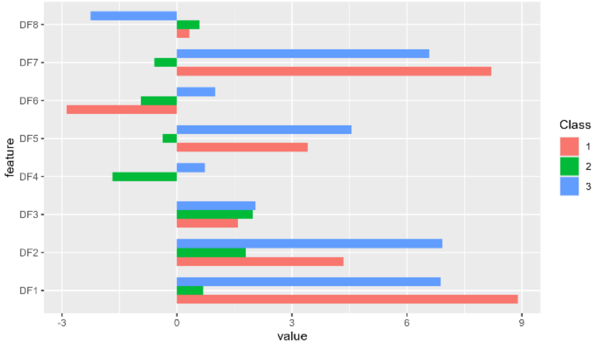
Cumulative mean accuracies for HPLC data



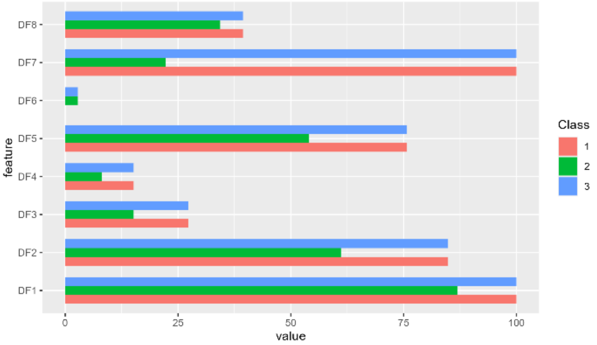
Importance of each feature for each class enose, knn



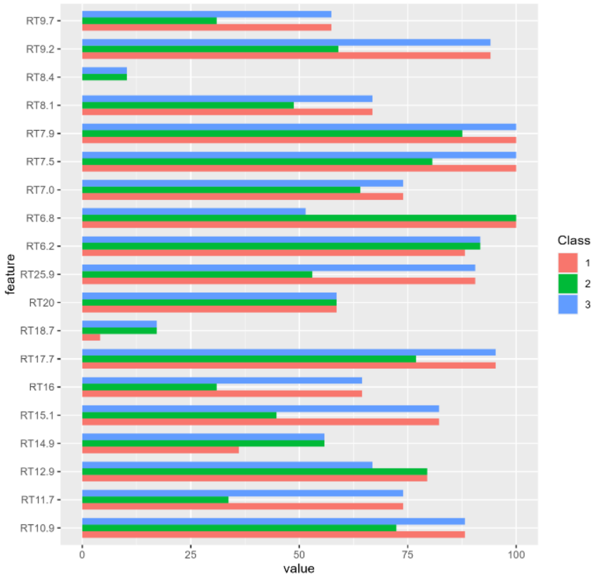
Importance of each feature for each class enose, rf



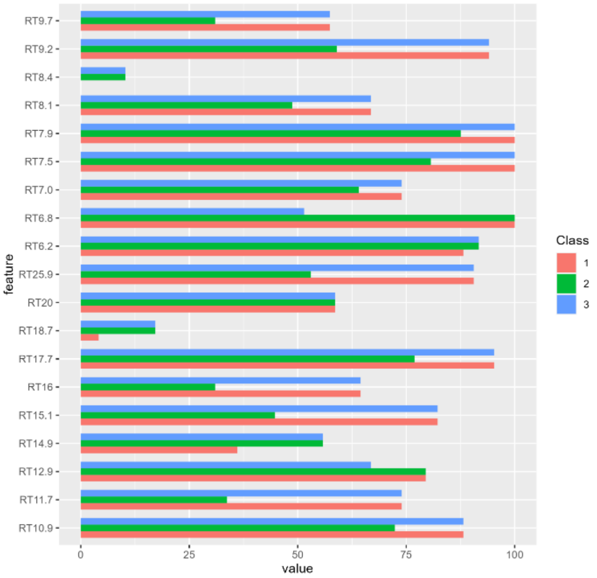
Importance of each feature for each class enose, svm

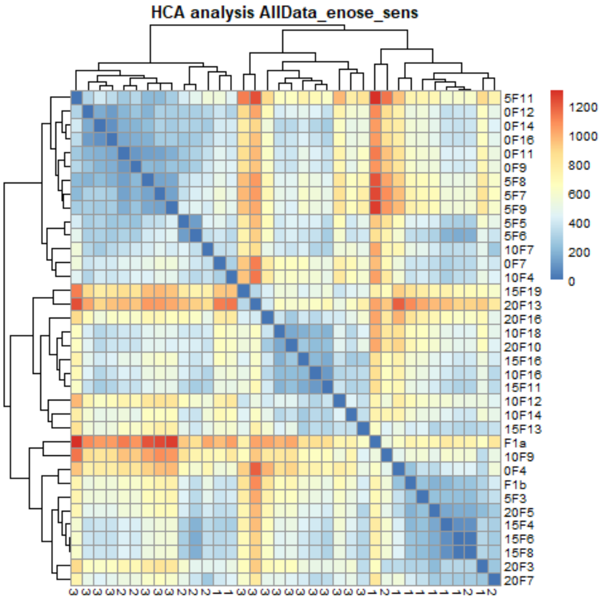


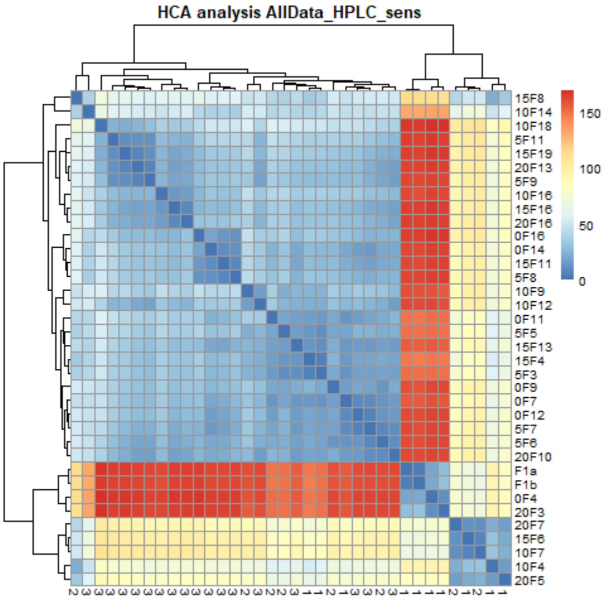
Importance of each feature for each class HPLC, knn

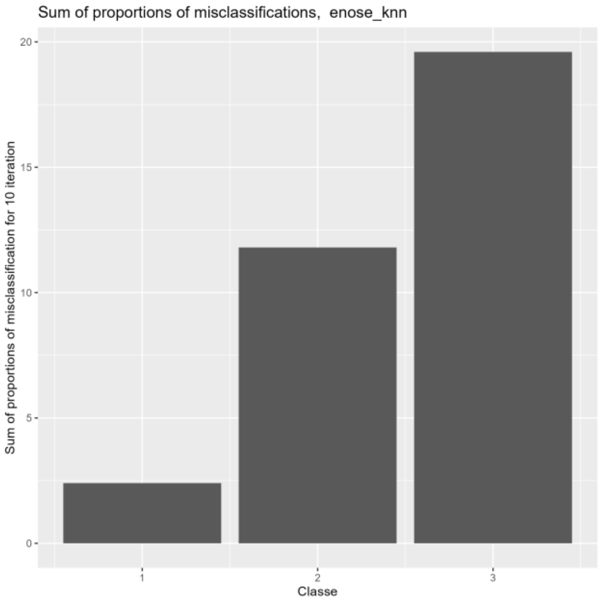


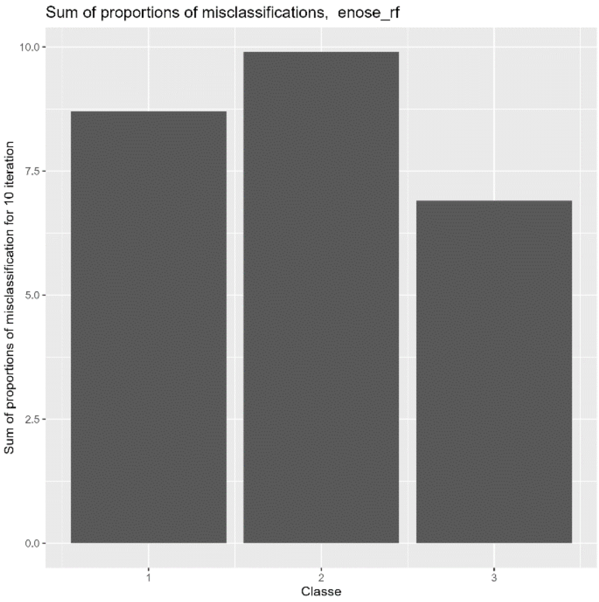
Importance of each feature for each class HPLC, svm

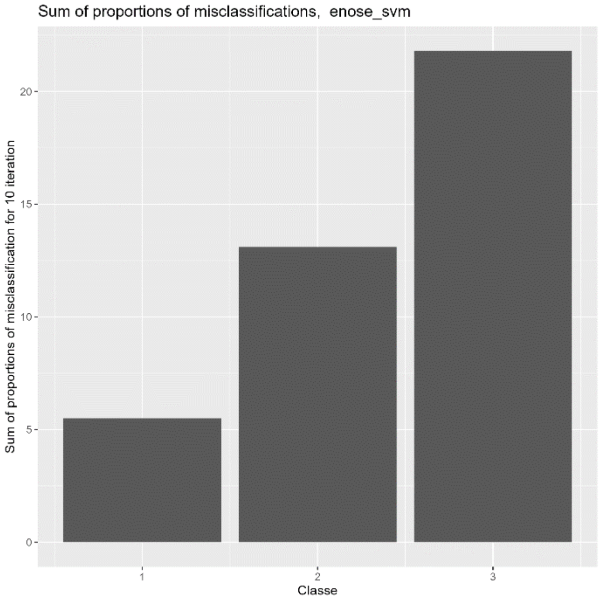


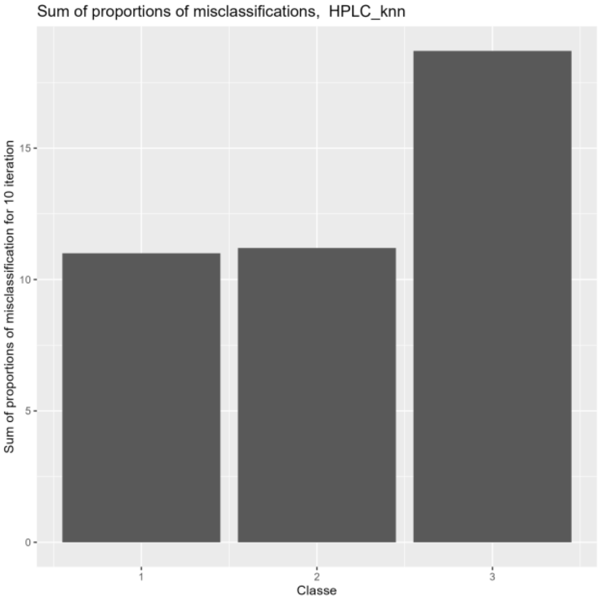

HCA analysis AllData_enose_sens


HCA analysis AllData_HPLC_sens

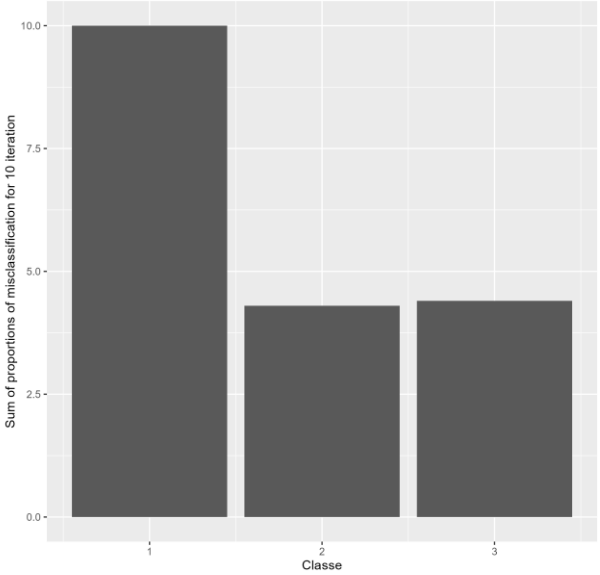

Sum of proportions of misclassifications, enose_knn


Sum of proportions of misclassifications, enose_rf

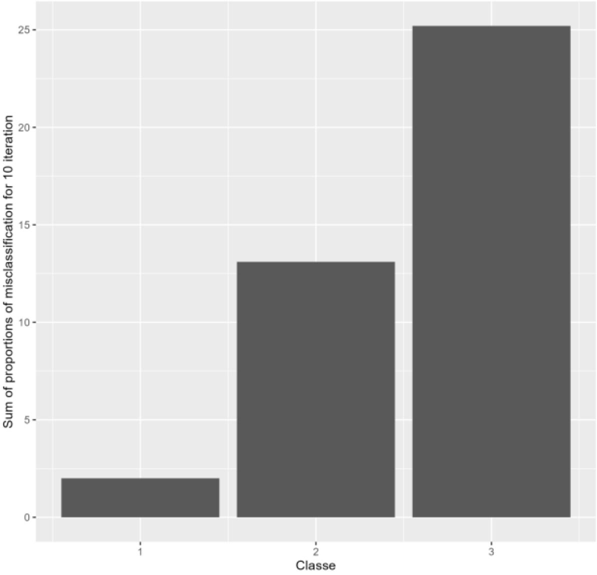

Sum of proportions of misclassifications, enose_svm


Sum of proportions of misclassifications, HPLC_knn

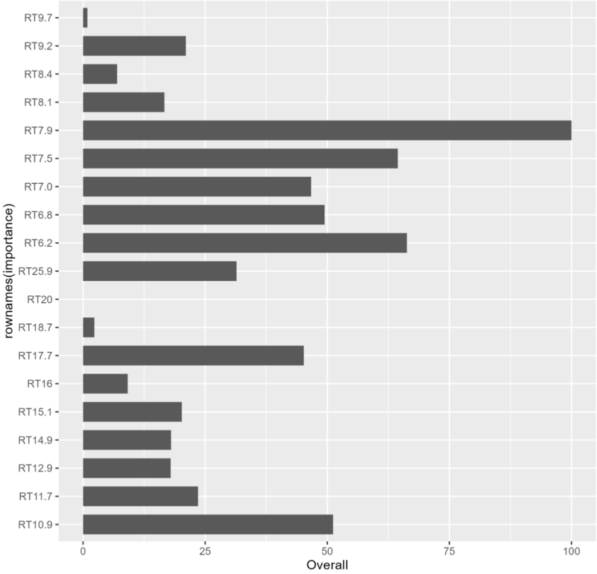
Sum of proportions of misclassifications, HPLC_rf



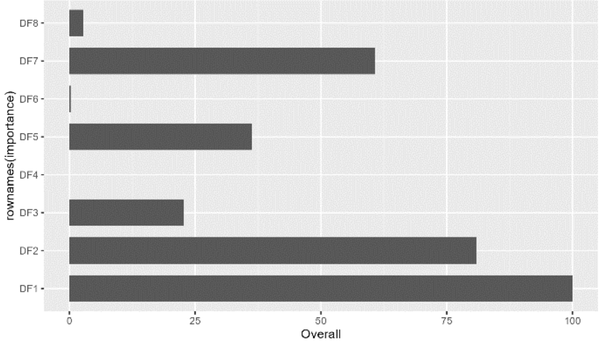
Sum of proportions of misclassifications, HPLC_svm

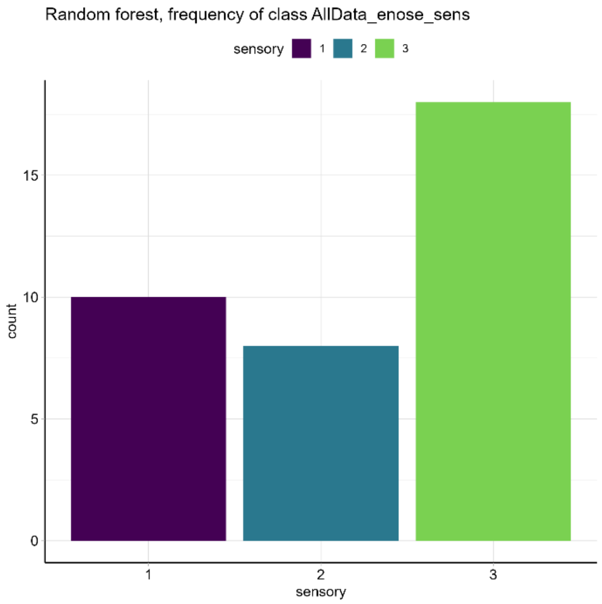


Importance of each feature for every class HPLC, rf

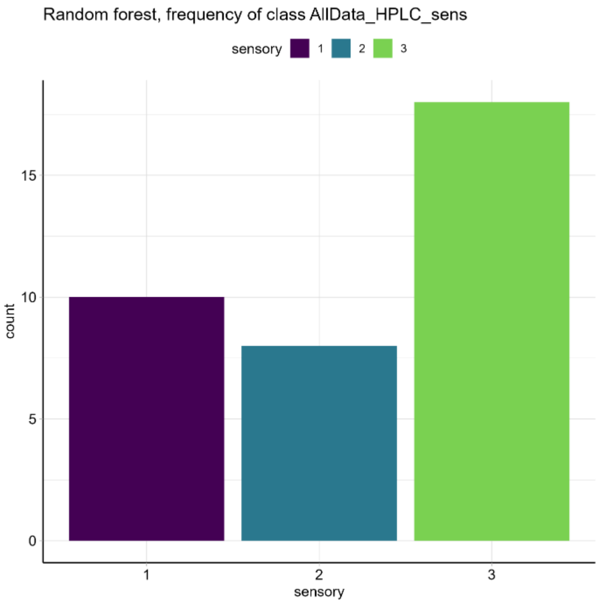


Importance of each feature for every class enose, rf

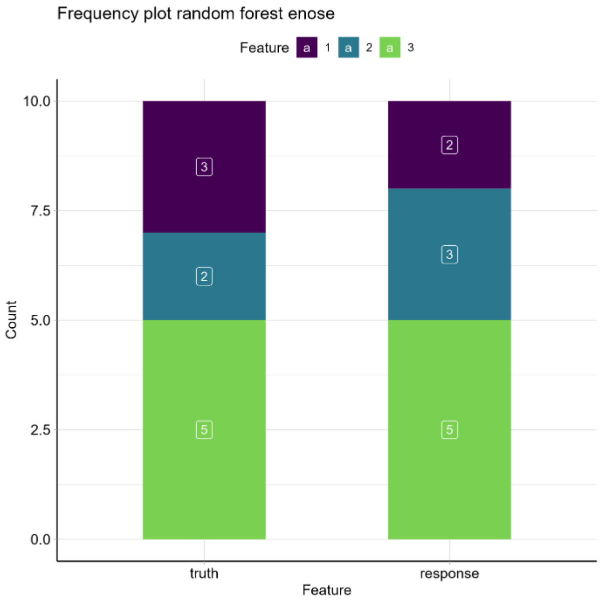




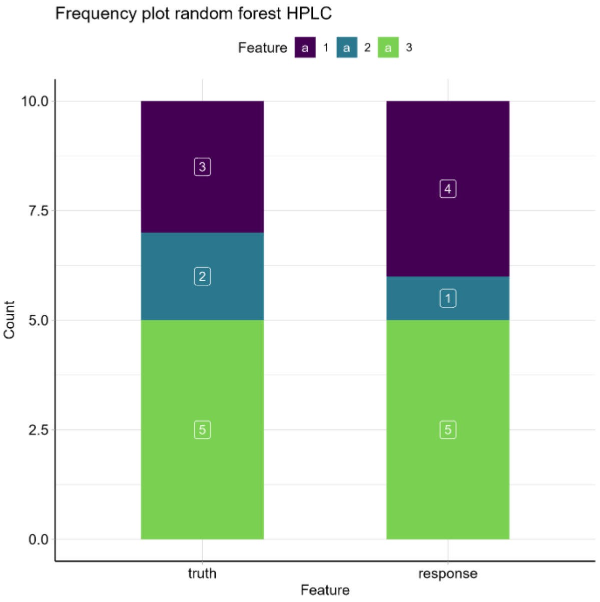
Random forest, frequency of class AllData_enose_sens



Random forest, frequency of class AllData_HPLC_sens



Frequency plot random forest enose



Frequency plot random forest HPLC