Machine Learning SVM and Random Forest

Introduction:

Crohns Disease is a chronic inflammatory bowel disease affecting people of all age groups. The aim of this work is to predict a sample type which is most viable to diagnose this disease. Therefore, GC-MS data was collected from patients that have been diagnosed with three different gut diseases (IBS, ulcerative colitis or Crohn's disease) and a group of healthy controls. For each patient samples from breath, blood, faeces and urine has been analysed. The GC-MS data contained TICs for each patient sample and the given retention time (RT)

Methods:

Two different approaches have been implemented to rank each sample type. First, *Sklearn-kit* used with Python will be explained. Second, *classyfiRe* package applied by R will be introduced.

Sklearn-kit: cross-validation

The GC-MS dataset consists of 4000+ variables for each sample type. To work with a machine learning algorithm, it is imperative to reduce the number of variables. A good approach to cope this problem is applying a principal component analysis and then perform machine learning on the PC scores.

Before the data has been processed the feature set has been normalized by using Pythons *sklearn.preprocessing* package (Pedregosa *et al, 2011)*. Mainly, learning algorithms benefit from this since outliers could influence the prediction accuracy and robust scalers can be more useful. The *preprocessing* module therefore provides a *Standardscaler* class to apply normalization of the data. In this case feature scaling was applied to minimize the gaps between values (i.e. a max value of 100 and min value of 10, produces a huge gap so features get scaled in a range from 0 to 1 to minimise the gap, and improve the models). Principal component analysis is a statistical technique to convert high dimensional data to low dimensional data by selection the most important features that capture maximum information about the dataset. These features are selected based on variance that they cause in the output. The feature that causes the highest variance is the first principal component. The feature responsible for the second highest variance is called second principal component, and so on (Jolliffe and Cadima, 2016).

After applying a PCA, ensembles of support vector machines have been trained to build an ensemble of individual classifiers. Therefore, the resampling method cross-validation (CV) was used and applied by *StratifiedShuffleSplit* which is a combination of *ShuffleSplit* and *StratifiedKFold. StratifiedShuffleSplit* returns stratified randomized folds which are made by keeping the percentage of samples for each class in train and test same on the same level. A major advantage in comparison to repeated stratified kfold is that it is possible to sample infinite times due to shuffling of the data every time prior to splitting into train-test sets, thus it is possible to fit more models. Furthermore, the fold data is overlapping whereas *StatifiedKfold* will always have different outcomes per fold. The splitting iterations for *StratifiedshuffleSplit* were set to 10-fold since literature showed a positive influence of 10-fold versus other values (Kohavi, 1995). 5-fold splits have been applied as well, turning to be more accurate and having a better test prediction accuracy for sample type faeces, but were quite unstable for the other sample types in case of showing higher variance but better bias, thus 10-fold has been used. Moreover, higher folds help the models being trained on larger training sets and tested on smaller test folds, thus resulting in a lower variance in average prediction accuracy as the models getting more of the available data as an input. In the model the test size for *StratifiedShuffleSplit* has been set to 1/3 of the samples so the model gets trained on 2/3.

To implement support vector machine classification, after the 10-fold shuffle-split, a bagging classifier was used to create a based estimator model. Sklearn-kits *BaggingClassifier* fits classifiers of random subsets from the original training set (*max_samples* were set to 1.0 to provide a bootstrap drawing of 100% of the training data) and calculates a final prediction by majority voting. This approach is useful to create an ensemble of classification models which is achieved by getting multiple bootstrap samples (with replacement) from the training dataset splits and fitting a classification model on it. To have more accurate and robust prediction the classification models then get combined to get an aggregated predictor consisting of bootstrap replicates of the training sets which operate as new training sets (Breiman, 1996). *BaggingClassifier* has been used with *LinearSVC* as base_estimator since the dataset is larger and *LinearSVC* tends to be faster.

For evaluation of the best parameters *gridSearchCV* has been used with *StratifiedShuffleSplit* as cross-validation method and *BaggingClassifier* as estimator. The scoring parameter was held as "accuracy" since the prediction accuracy of each sample type is the aim of this work.

For implementation of the Random Forest Algorithm all steps were used as written above with difference in the usage of *RandomForestClassifier* as the based estimator model. In random forests, as well as before in SVM with *baggingClassifier,* each tree in the ensemble is created from a sample drawn with replacement.

To evaluate the validation and functionality of the model applied, a permutation test has been covered on SVM using sklearn-kits *permutation_test_score* function. Therefore, the model classification accuracy was compared to data of random choice and the class vectors were shuffled from real class vectors multiple times to find the average classification accuracy. Here, 100 permutations have been set. For double-checking the data of each sample has also been tested with 1000 permutations and permutations equal to the optimum number of ensembles resulting in a similar outcome. Thus 100 permutations have been implemented.

<u>classyfiRe: bootstrap validation</u>

ClassyfiRe is a package that uses highly optimised SVM ensembles to provide a powerful classification workflow (Chatzimichali and Bessant, C, 2016).
Prior to evaluate classification models the data has been reduced into principal components inside R with *prcomp* by using scaling for better results. After getting lower dimensional data the *classyfiRe* function *cfBiuld*, which implements advanced ensembles of radial basis function (RBF) support vector machines, has been used to seperate the data into train- and testing subsets using a bootstrap validation approach. Setting the best parameters for the number of

ensembles (*ensNum*) and bootstrap iterations (*bootNum*) is important for getting the best classification model. Therefore, the average test accuracy has been tested using *ggEnsTrend* by adding models to the ensemble. With 1500 models added every sample type seemed to be classified with a stable test prediction accuracy. Bootstrap iterations were set at a minimum value of 100 iterations due to computational shortcomings since R-Studio Cloud was used and only 4 CPUs were available.

After having the best possible set of parameters, a permutation test with *cfPermute* as done in Python was planned to be applied to compare the model with data of random choice and evaluate the functionality of the model. However, due to computational shortcomings only models with 50 bootstraps and 50 iterations could have been evaluated using 100 permutations.

For both validation techniques a training and test size of 70% / 30% has been used.

Results:

Principal component analysis

With applying principal component analysis, the data for the sample type blood has been reduced into 32 principal components. The first principal component is responsible for 60.38% variance. Similarly, the second principal component causes 15.02% variance in the dataset and the third principal component 9.74% variance inside the dataset. Moreover, after the third principal component the change in variance almost diminishes. Collectively, (60.38 + 15.02 + 9.74) = 85.14% of the classification information contained in the feature set is captured by the first three principal components and they can be selected for the sample type blood (Fig.1).

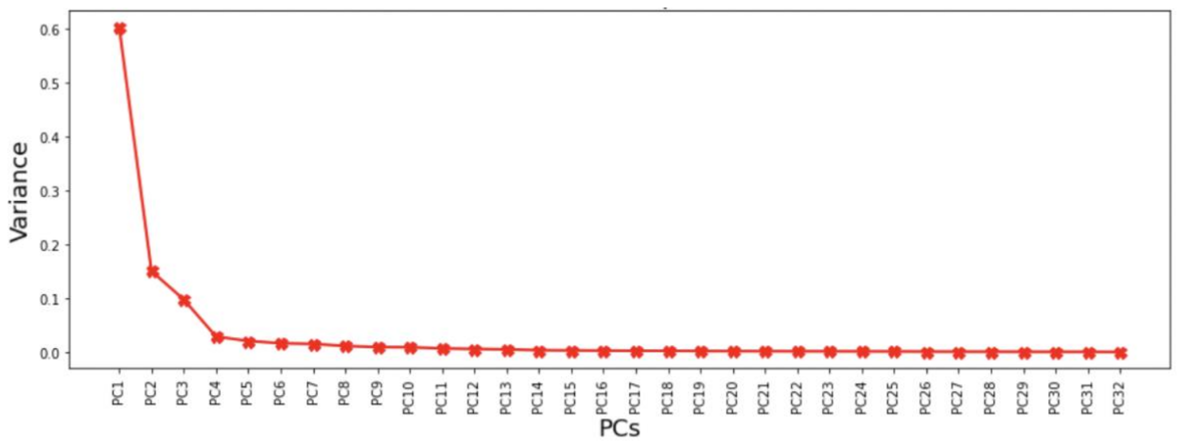

Figure 1 PCA for sample type ***blood****. The first principal component is responsible for 60.38% variance. Similarly, the second principal component causes 15.02% variance in the dataset and third principal component 9.74% variance in the dataset. Combining the first three components 85,14%.*

After using PCA for sample type breath the data has been reduced to two principal components. PC1 contains 82,27% of the classification information and PC2 stores 11,35% of the variance. Together both principal components cover 93,62% of the variance in this feature set (Fig. 2).

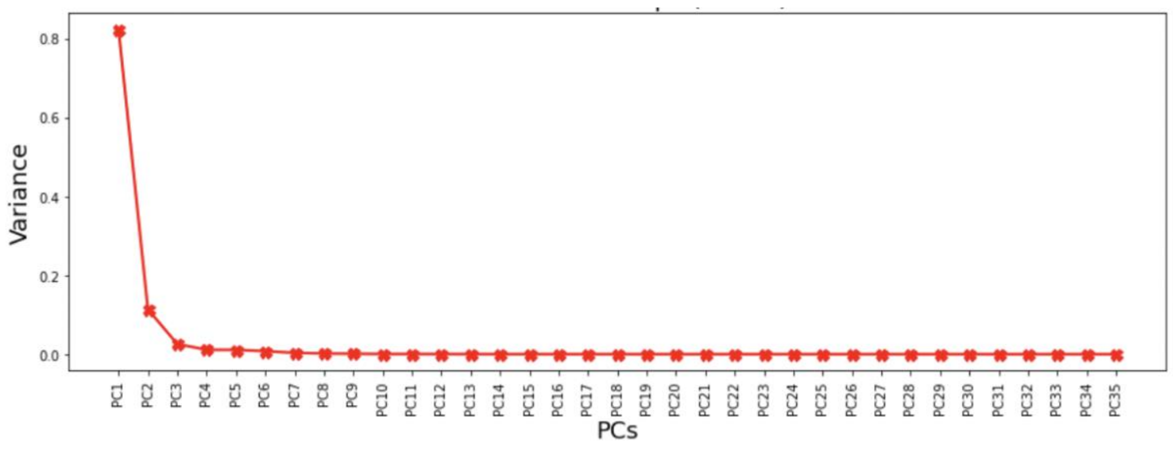

Figure 2 PCA for sample type ***breath****. The first principal component covers 82,27% variance. The second PC is responsible for 11,35%. Collectively PC1 and PC2 store 93,62% of the classification information contained in the feature set of sample type breath.*

For the faeces sample type PC1 is responsible for 35.06% variance. Similarly, PC2 causes 21.68% variance in the dataset. Additionally, PC3 covers 14.49% and PC4 captures 6.45% variance in the dataset. Collectively, (35.06 + 21.68 + 14.49 + 6.45) = 77.68% of the classification information is set by the first four principal components (Fig 3).

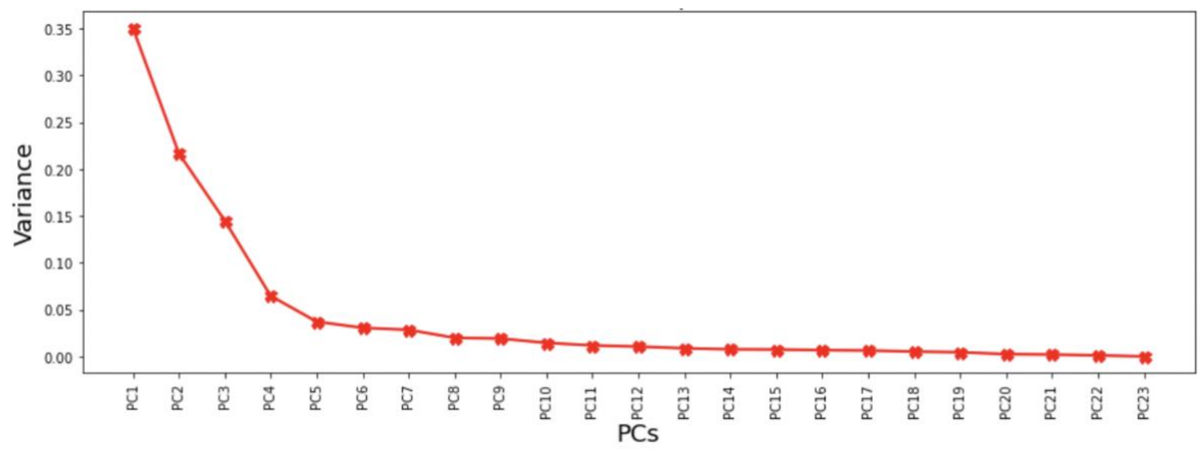

Figure 3 PCA for sample type ***faeces****. The first principal component covers 35,06% variance. The second PC is responsible for 21,68%. PC3 covers 14.49% and PC4 6.45%. Collectively PC1 – PC4 store 77,68% of the classification information.*

Looking at the urine sample type, the first principal component is responsible for 75.39% variance and the second principal component causes 10.54% variance in the dataset. Collectively, (75.39 + 10.54) = 85.93% of the variance is contained in the first two PCs and for the following principal components the variance almost diminishes (Fig. 4). Therefore, first two components can be selected for the sample type urine.

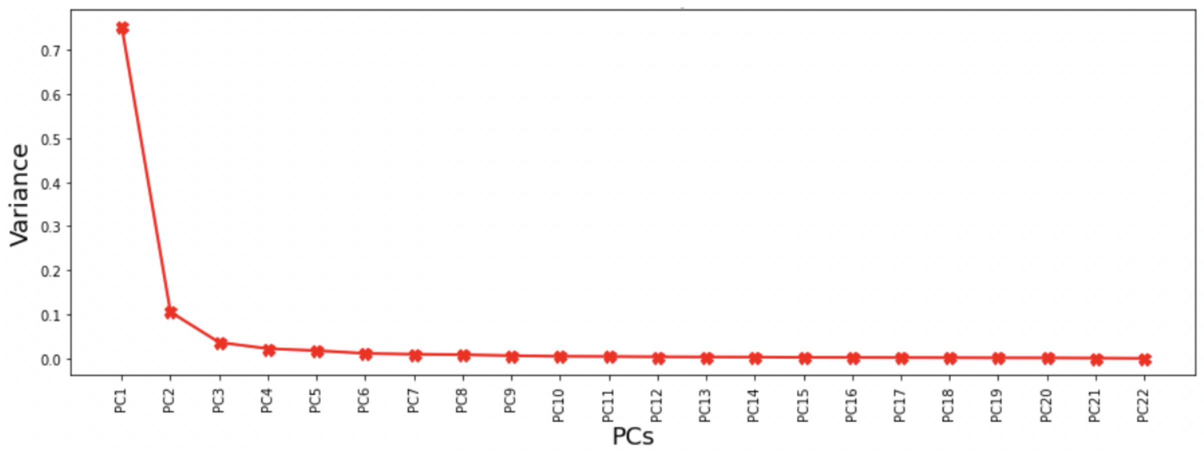

Figure 4 PCA for sample type urine. The first principal component covers 75,39% variance. The second PC is responsible for 10,54%. Collectively PC1 and PC2 store 85,93% of the classification information.

Sklearn-kit: SVM ensemble

First, the average test accuracy for the sample type blood was calculated, using support vector machines and cross-validation, adding up to 1500 ensembles to the model. With around 1000 models added to the ensembles the accuracy tended to be stable with ~63% (+/2) prediction accuracy (Fig. 5). However, there is no strongly robust accuracy visible with the number of models added to the ensemble, and the prediction error is getting smaller but is not diminishing.

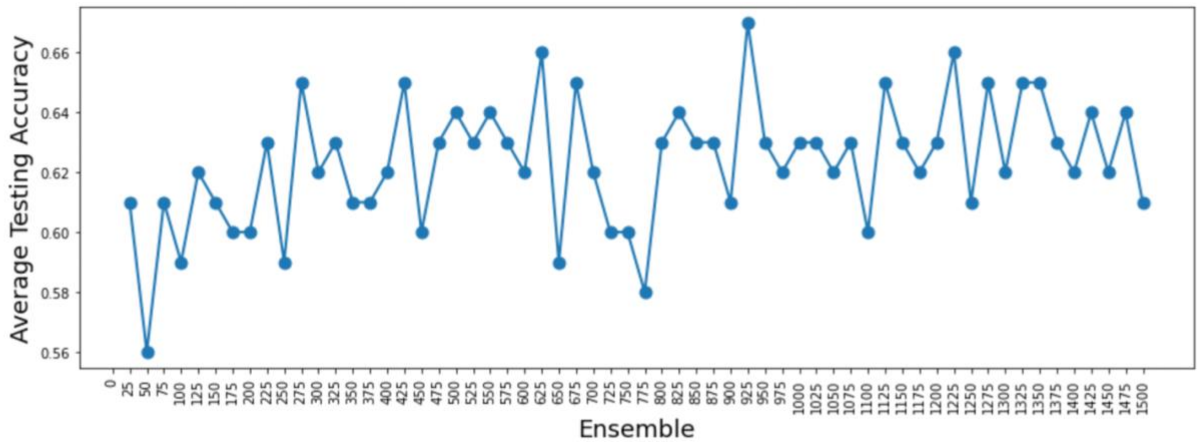

*Figure 5 SVM average testing accuracy for sample type **blood** in comparison to number of ensembles. Fitting 10 folds for each of 60 candidates totaling 610 fits. Average test accuracy tends to be most stable ~63 % (+/-2) with at least 1000 ensembles.*

Second, the sample type breath tends to show an average testing accuracy of ~58,5% (+/- 2) after 850 models added to the ensemble (Fig. 6). Similar to the sample type blood there is still a larger variation visible.

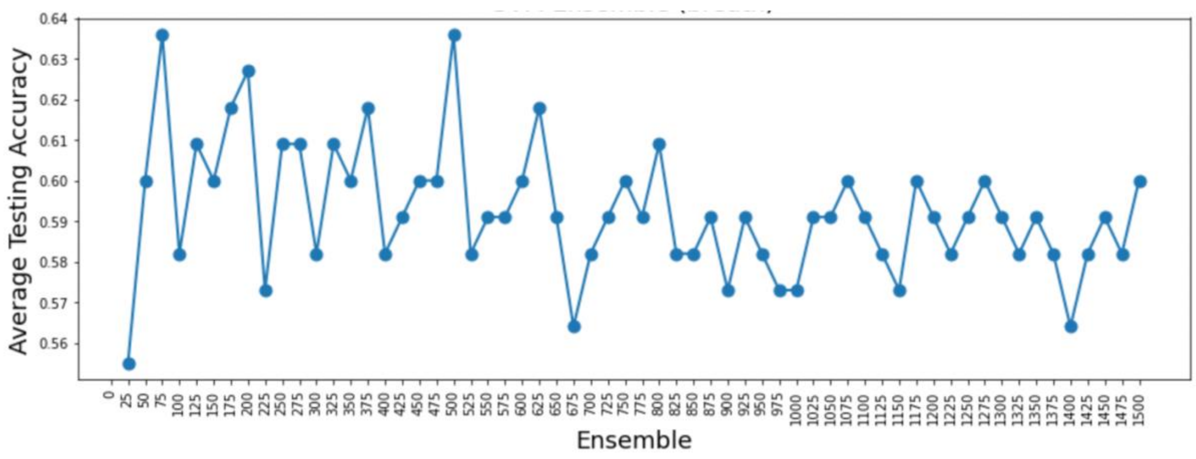

*Figure 6 SVM average testing accuracy for sample type **breath** in comparison to number of ensembles. Fitting 10 folds for each of 60 candidates totaling 610 fits. Average test accuracy tends to be most stable ~58,5 % (+/-2) with at least 850 ensembles.*

Third, the sample type faeces shows an average testing accuracy of ~89% (+/-1) after 400 models added to the ensemble (Fig. 7). In this case the prediction error is much more robust in comparison to the other sample types.

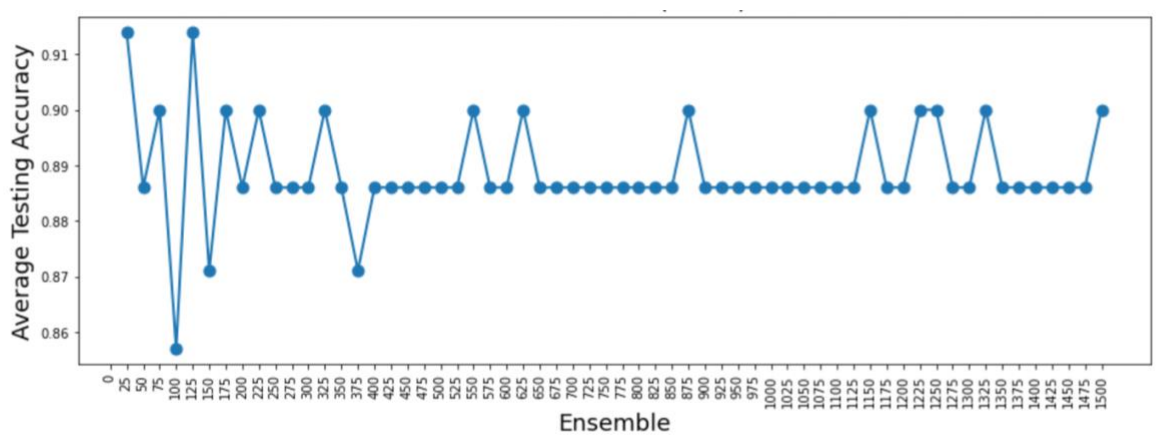

Figure 7 SVM average testing accuracy for sample type ***faeces*** *in comparison to number of ensembles. Fitting 10 folds for each of 60 candidates totaling 610 fits. Average test accuracy is stable ~89% (+/-1) with at least 400 ensembles.*

In the end the sample type urine shows an average testing accuracy of about 55% (+/- 1) but tends to also be more robust than "blood" and "breath" after 450 ensembles added to this model (Fig. 8). The deviation in testing average accuracy after 450 models added is mostly diminishing.

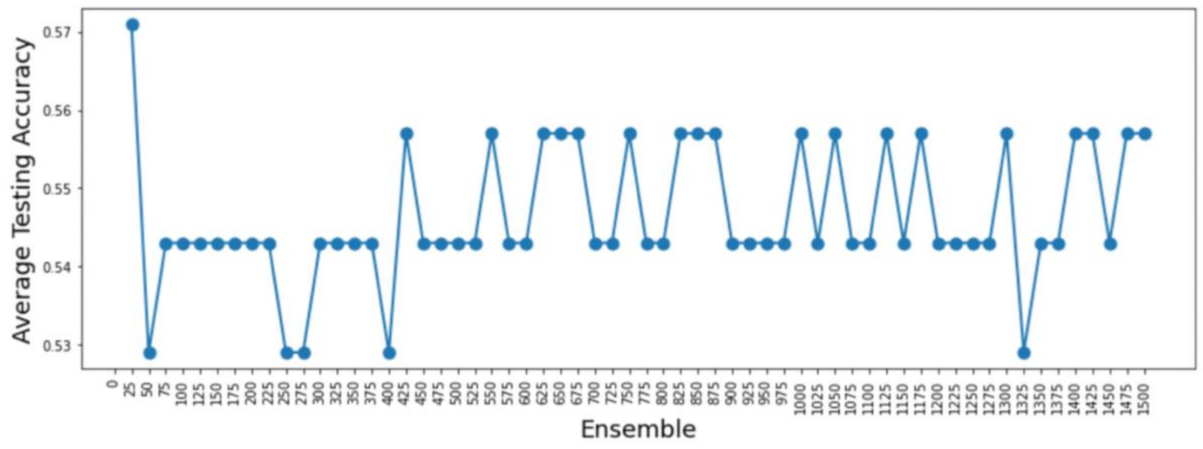

Figure 8 SVM average testing accuracy for sample type ***urine*** *in comparison to number of ensembles. Fitting 10 folds for each of 61 candidates totaling 610 fits. Average test accuracy is stable ~55% (+/-1) with at least 450 ensembles.*

Sklearn-kit: Permutation test

Fitting the models on a permutation test compared the model classification accuracy with random chance. When randomly shuffling the class vectors from real class vectors 100 times (100 permutations) it has shown that the blood sample type score was around 63% (p-value: 0.079) which is not much better than those obtained by using permuted data (Fig. 9). It indicates that there is a higher likelihood that this score is also achievable by random chance alone and the dataset may not contain real dependency between features and labels. Similarly, for the breath sample type the score obtained was around 58% (p-value: 0.188) showing no significant dependency between the dataset's variables and groups (Fig. 10). However, for faeces sample type there has been a stronger dependency between the datapoints in the given GC-MS data detected (p-value: 0.01) (Fig.11). This indicates a lower likelihood that the original score of 83% would be obtained by random chance. In the end, the permutation test from urine sample type has been evaluated with a score on the original data of 55% (p-value: 0.485), given no significant real dependency between the features and labels of the dataset (Fig. 12).

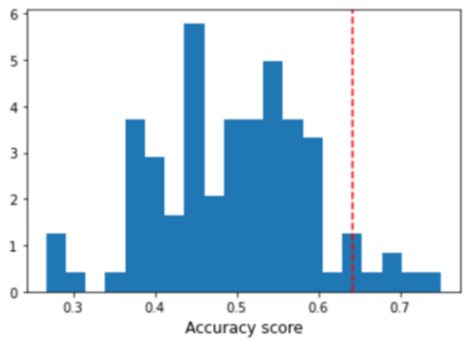

*Figure 9 Permutation test on **blood** sample type with 100 permutations (Score (red line) on original data: 0.63, p-value: 0.079).*

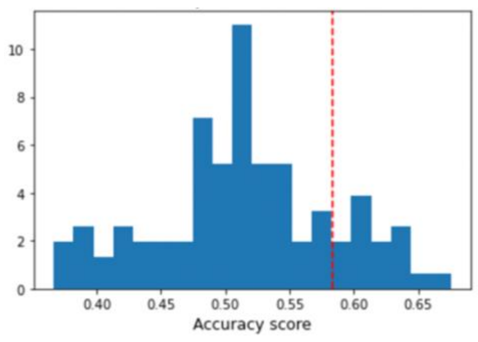

*Figure 10 Permutation test on **breath** sample type with 100 permutations (Score (red line) on original data: 0.58 , p-value: 0.188).*

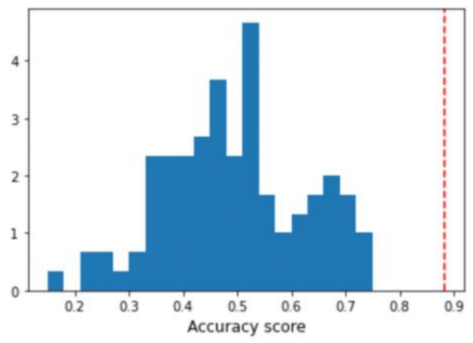

*Figure 11 Permutation test on **faeces** sample type with 100 permutations (Score (red line) on original data: 0.89, p-value: 0.01).*

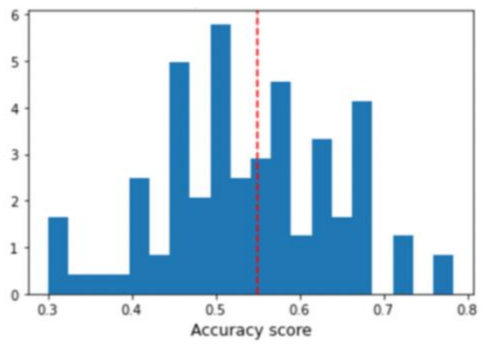

*Figure 12 Permutation test on **urine** sample type with 100 permutations (Score (red line) on original data: 0.55 , p-value: 0.485).*

Sklearn-kit: Random forest

Using random forest algorithm with a test size of 30% on the best ranked sample type faeces shows an average testing accuracy of ~85% (+/-1) (Fig. 13). After adding 500 trees the average testing accuracy variation seems to be robust.

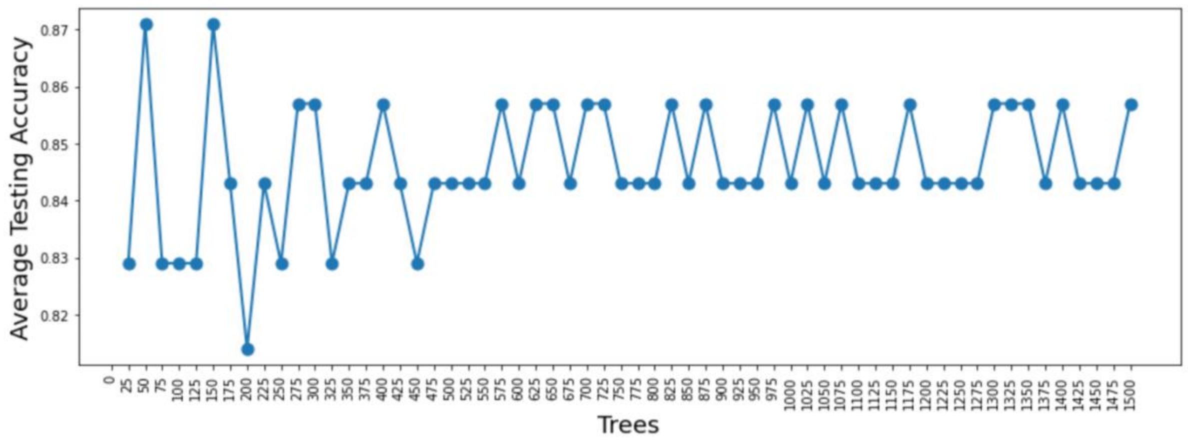

Figure 13 Random forest average testing accuracy for sample type ***faeces*** *in comparison to number trees in the forest. Fitting 10 folds for each of 60 candidates totaling 610 fits. Average test accuracy is stable ~85% (+/-1) with at least 500 trees added.*

classyfiRe: SVM

Using support vector machines in *classyfiRe* package, the average test accuracy for the sample type blood was calculated, using bootstrap validation with 100 bootstrap iterations, adding up to 1500 ensembles to the model. With around 740 models added to the ensembles the accuracy tends stable with ~51% (+/- 0.1) prediction accuracy in sample type blood (Fig. 14). Moreover, the accuracy is visible as strongly robust with this number of models added to the ensemble, and the prediction error between the datapoints is diminishing.

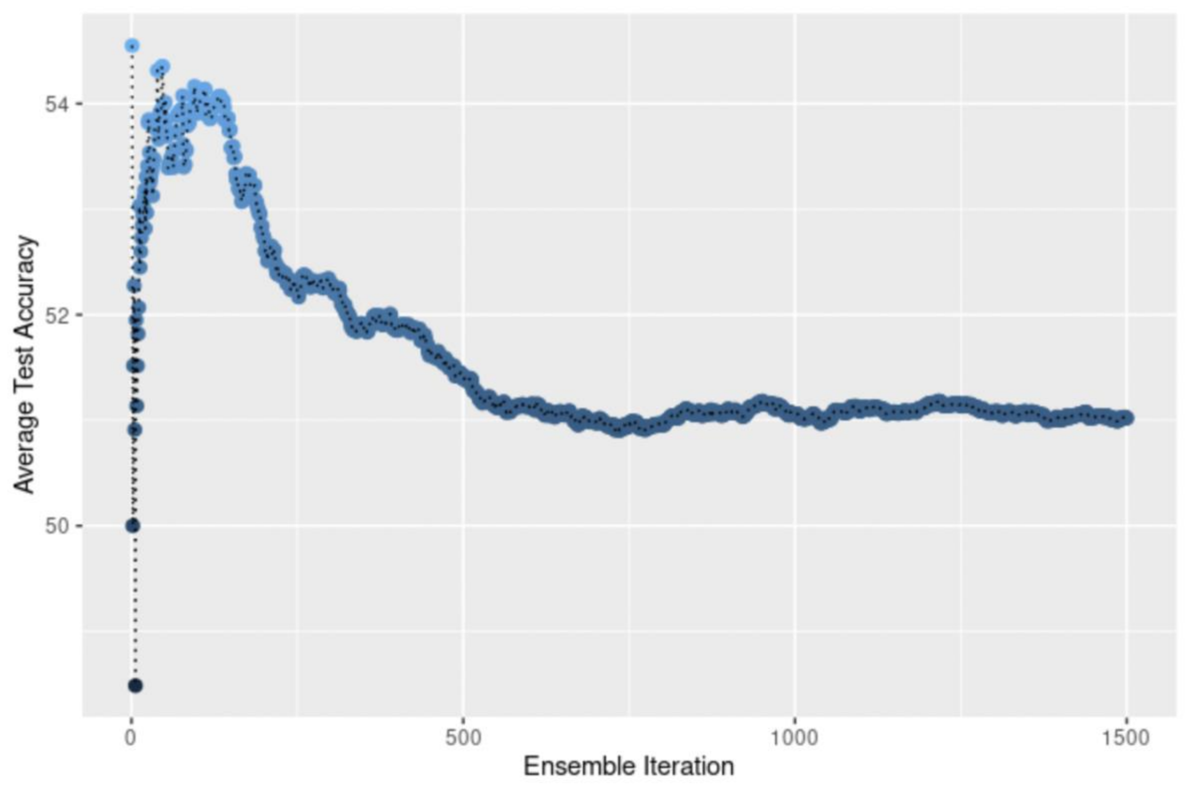

*Figure 14 Average test accuracy for the sample type **blood** with 100 bootstrap iterations and up to 1500 models added to the ensemble. With an ensemble size of around 750 models the Average test accuracy tends to be stable at ~51% (+/- 0,1).*

The average test accuracy for sample type breath is stable with ~57.75% (+/- 0.1) prediction accuracy (Fig. 15) with adding at least up to 750 models to the ensemble. In fact, the accuracy is robust with this number of models added to the ensemble, and the variation between the datapoints is diminishing creating a nearly linear line.

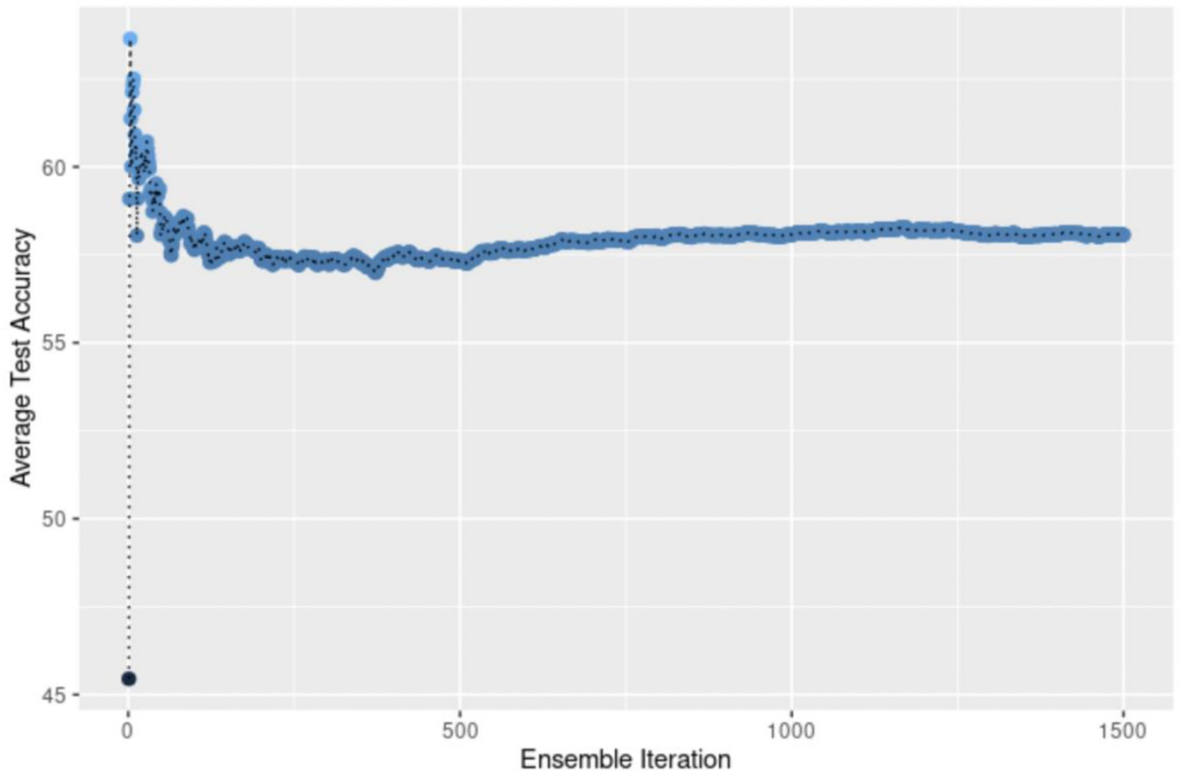

*Figure 15 Average test accuracy for the sample type **breath** with 100 bootstrap iterations and up to 1500 models added to the ensemble. With an ensemble size of around 750 models the average test accuracy tends to be stable at ~57.75% (+/- 0,1).*

Looking at sample type faeces the average test accuracy is stable with ~81.5% (+/- 0.2) prediction accuracy (Fig. 16) when adding at least up to 800 models to the ensemble. Similar to previous analysed sample types, the accuracy is robust with this number of models added to the ensemble, and the variance between the datapoints is diminishing.

The average test accuracy for sample type urine is stable with ~62.5% (+/- 0.1) prediction accuracy (Fig. 17) when adding at least up to 1000 models to the ensemble underlying 100 bootstrap iterations. As shown in the graphs before the accuracy is robust with this number of models added to the ensemble, and the prediction error between the datapoints is diminishing showing no heavy outliers.

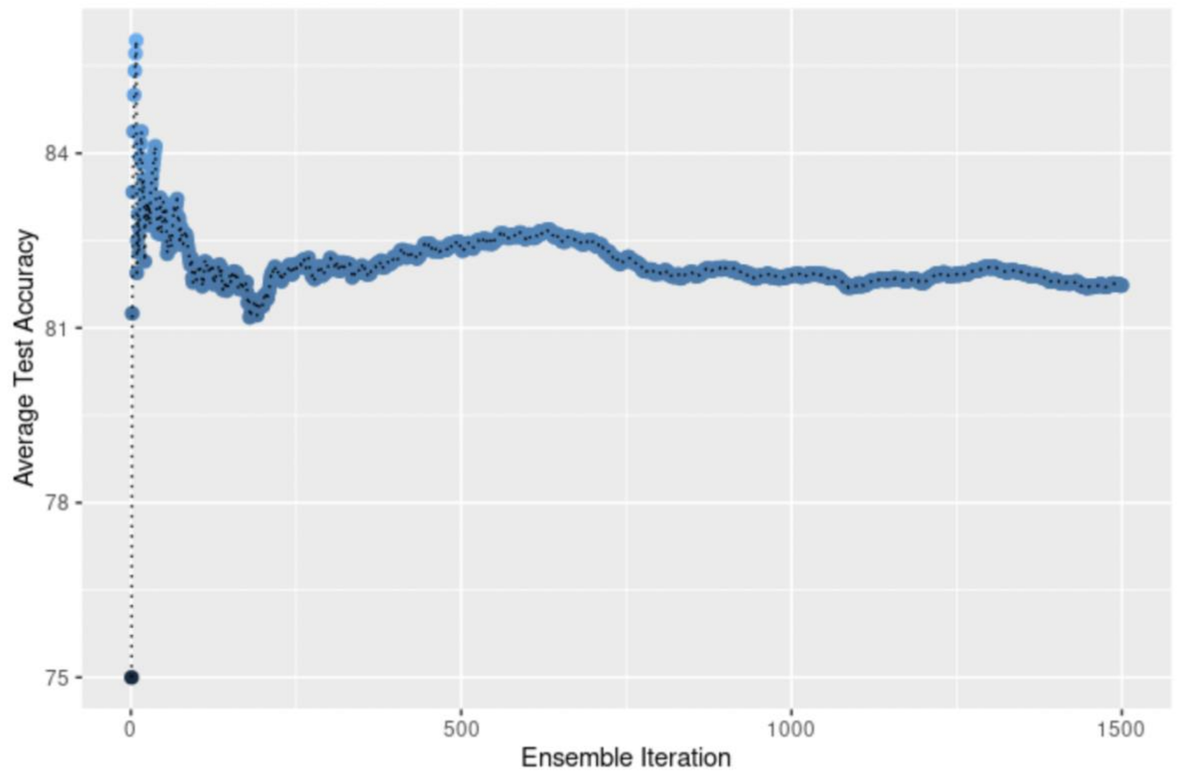

*Figure 16 Average test accuracy of sample type **faeces** with 100 bootstrap iterations and up to 1500 models added to the ensemble. With an ensemble size of around 800 models the average test accuracy tends to be stable at ~81,5% (+/- 0,2).*

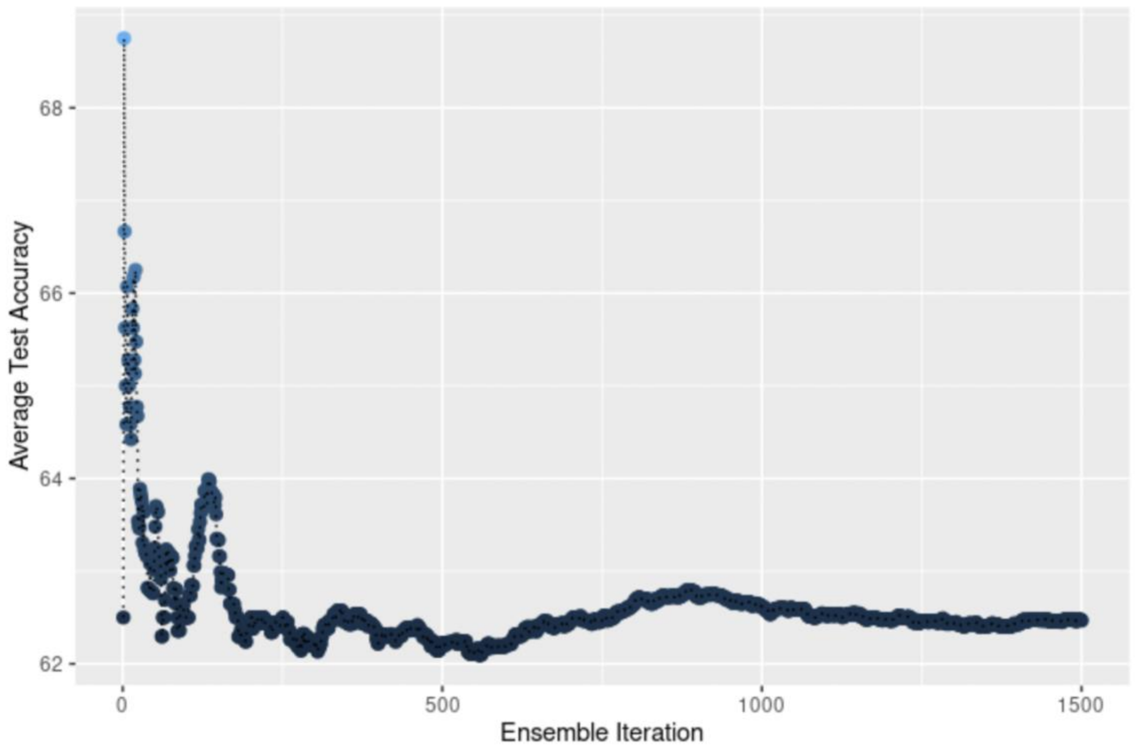

*Figure 17 Average test accuracy of sample type **urine** with 100 bootstrap iterations and up to 1500 models added to the ensemble. With an ensemble size of around 1000 models the average test accuracy tends to be stable at ~62,5% (+/- 0,1).*

Discussion

Ranking the sample types for the best in terms of diagnosing Crohn's disease both support vector machine approaches declare the faeces sample type in being rank one. Moreover, cross-validation in combination with shuffling (*StratifiedShuffleSplit)* and bootstrapping (*BaggingClassifier)* builds a model of 89% (Fig. 7) accuracy in comparison to 81.5% (Fig. 16) accuracy implementing *classyfiRe*'s *cfBuild* method. In fact, the difference is not tremendously high but still considerable. In the end both approaches build a model with over 80% accuracy which is an excellent result for machine learning algorithms. However, the other three sample types show divergent outcomes with different methods applied. In case of SVM in R with bootstrap validation only applied, rank two is considered to be the urine sample type (62.5%, Fig. 17). Increasing rank three breath (57.75%, Fig. 15) and rank four blood (51%, Fig. 14). In contrast to that using SVM from *sklearn-kit* and a combination of *StratifiedShuffleSplit,* with 10-fold splits, into *BaggingClassifier*, thus cross-validation in combination with a bootstrapping approach gets a slightly different ranking. Here in case of test prediction accuracy, rank two is blood (63%, Fig. 5), rank three breath (58%, Fig. 6), and rank four urine (55%, Fig. 8)). However, the predicting accuracies for rank two to four are in each case very low, thus comparing to the permutation test (Fig. 9, 10, & 12) and random permuted data no real dependencies between features and labels were discovered. This underlines the fact that the models for blood, breath and urine create unreliable machine learning models to diagnose Crohn's disease. In fact, low sample sizes might be the reason for this, but considering each sample type has similar amounts of samples still leaves feaces sample type as best for diagnosing Crohn's disease. Permutation testing was not being applied in R due to computational shortcomings, but the similar results of the average test prediction accuracies produced by Pythons *sklearn-kit* and R's *classyfire* package assume equal results. Considerably, for the best ranked sample type faeces it has been shown that true dependencies between features and labels in comparison to randomly permuted data were shown (p-value: 0.01, Fig. 11). Implementing random forest algorithm on the faeces sample type with cross-validation lowered the average test prediction accuracy by four percent points to 85% (Fig. 18). Furthermore, the prediction error was closely related to that shown in support vector machines assuming SVM a better classification algorithm for this kind of data.

Summarizing cross-validation for these kinds of datasets creates better results referring to test prediction accuracies than bootstrapping alone. It is mentionable that the prediction error for bootstrapping stays relatively low but even for cross-validation and regarding the best ranked faeces sample type there are no heavy prediction errors measurable when fitting enough ensembles to the model. Nevertheless, previous studies have been shown that bootstrapping might be the preferred approach to cover biological data due to a more exemplary and stable performance than other validation techniques (Wehrens et al., 2000, Liland 2011, Chatzimichali and Bessant 2015) and referring to the classification models written for the other three sample types it still should be considered on being more weighable. However, results may be comparable due to the small sample size and also not only using cross-validation than also including bootstrapping by the use of *BaggingClassifier.*

References

1. Chatzimichali, E.A., Bessant, C. Novel application of heuristic optimisation enables the creation and thorough evaluation of robust support vector machine ensembles for machine learning applications. *Metabolomics* **12,** 16 (2016).

2. Harris, C.R. et al., 2020. Array programming with NumPy. Nature, 585, pp.357–362.

3. Jolliffe Ian T. and Cadima Jorge 2016 Principal component analysis: a review and recent developmentsPhil. Trans. R. Soc. A.3742015020220150202

4. R. Kohavi, 1995 A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection, Intl. Jnt. Conf. AI

5. Scikit-learn: Machine Learning in Python Pedregosa *et al.*, JMLR 12, pp. 2825-2830, 2011.