Danmarks Tekniske Universitet

DTU

Special Course Report

Feature Engineering for Microbiome Data

Author

Siddhi Yash Jain - s212937

email: s212937@dtu.dk

Persimune (CHIP), Rigshospitalet

November 29, 2022

Contents

1 Introduction

Metagenomics is defined as the process of sampling and studying genomes in an environmental sample. Studying the micro-organisms of an environment can help understand the microbial composition, functionality of the microbes and how they affect the host or the environment (1). Metagenomic data is obtained by sequencing an environmental sample to obtain the RNA amplicon or DNA sequences of all the microorganisms in that sample. These reads are then assembled into contigs, which are overlapping reads that make up a continuous sequence. This is followed by binning, where contigs are sorted into groups that represent a single genome or genomes of closely related organisms. After quality control of these *bins*, the obtained genomes are annotated with the help of reference databases to identify genes of interest (2). There exist several algorithms to estimate taxonomic abundance profile of a metagenomic sample, either from annotated functional genes or from the genome sequence. Once the sequences are classified into taxa, similar sequences are grouped together based on an identity threshold to form Operational Taxonomic Units (OTUs) and obtain their counts(3).

Metagenomic data can be defined as a random, fixed size sample of the relative abundance of the micro-organisms in an environmental sample, as the sample size is determined by the sequencing machine. This implies that metagenomic data is compositional in nature, and results can be greatly flawed if non-compositional statistics are used to analyze this data (4). Generally, metagenomic data also has high dimensionality and sparsity, with as much as 90% of the values being zero entries or close to zero (5).The data generally needs to be normalized to account for biological differences and technical errors. Using common methods like proportion measures or rarefaction for normalization of compositional data can fail to reveal the differential abundances of microbes, especially in microbial communities with large variance among the species abundance. One way around the constant sum constraint of compositional data is taking the logarithm of ratios which maintains the correlation and covariance of the data, while also making it useable for calculating relative abundance. There are three main methods to do this; Additive log-raio (ALR), Centered log-ratio (CLR), Isometric log-ratio (ILR) and their variations (6).

Although there is a lot of literature available on normalization methods for microbiome data, there is not a lot of research done on feature engineering techniques that can be applied to this data for better target prediction using machine learning. With modern sequencing technologies, it is possible to sequence vast amounts of data at a fraction of the cost and time. The information that can be extracted from metagenomic data is vital for studying development of diseases in humans and animals and predicting the disease condition given the metagenomic profile. But because of the high dimensionality of the data, it is often difficult to efficiently apply machine learning tools as it might be too computationally expensive. To improve scalability and generalization power of machine learning tools when dealing with high dimensional data, one solution is feature selection. But feature selection comes with the cost of loss of information, and in microbiome data eliminating features may give the model a false notion of the presence or absence of the removed features.

Currently there exist many feature selection tools for metagenomics like Fizzy (7) which

utilizes Information-theory based algorithms and MetAML (8) which applies feature selection steps implicitly incorporated in Random Forests, Lasso and ENet algorithms. Another sophisticated algorithm developed by Henschel Lab uses the information hidden in phylogenetic hierarchy of the features to extend the feature set and then filter features based on Information Gain (IG)(9). Although tools like Fizzy can effectively reduce the size of a dataset, it may also remove features with biological significance in understanding disease pathophysiology. The information in these features may be explainable mathematically, but biological interpretation may not be easy. Similarly, intrinsic feature selection tools in models like random forests are efficient but not explainable. The hierarchical feature engineering tool from Henschel lab requires the correct taxonomic classification of each feature, which is highly influenced by how the data was processed, missing data, and the presence of novel species.

The aim of this research was to find novel feature engineering methods for microbiome data, which preserve information encoded in this data while also possibly reducing dataset size. With biological data, it is essential to have explainability and the methods here have been tried with consideration of this fact. The tools tried are based on a machine learning classification task of disease vs control. Along with this, the methods tried in this research have been performed with and without compositional normalization of data. This was done to help understand why it is important to consider the compositional nature of microbiome data when doing any analysis.

2 Methods

2.1 Dataset

The datasets used for this research are metagenomic data obtained from fecal samples of colorectal cancer subjects from 7 different countries (10). Each dataset has samples from control and cancer subjects, roughly equally distributed as shown in Table 1. The datasets were obtained from a Kaggle challenge, where the data has been processed from sequences to obtain species compositions for each individual. All the datasets consist of the same 850 features which are taxonomically classified bacteria and one of the features is the classification of the individual as "control" or "CRC". The data has a zero-inflated distribution as normally seen in metagenomic data (Appendix 7).

Country	Total samples	Condition	Count	Ratio (%)
AT-CRC_species (Austria)	109	CRC	46	42.2
		control	63	57.8
CN-CRC_species (China)	128	CRC	74	57.8
		control	54	42.2
DE-CRC_species (Germany)	120	CRC	60	50.0
		control	60	50.0
FR-CRC_species (France)	114	CRC	53	46.5
		control	61	53.5
IT-CRC-2_species (Italy)	60	CRC	32	53.3
		control	28	46.7
IT-CRC_species (Italy)	53	CRC	29	54.7
		control	24	45.3
JP-CRC_species (Japan)	80	CRC	40	50.0
		control	40	50.0
US-CRC_species (United States)	104	CRC	52	50.0
		control	52	50.0

Table 1: Details of all datasets used

The advantage of using multicentered data is that it allows for assessment of the generalization power and robustness of the machine learning methods. Since the samples in these datasets are collected from different countries, the models account for genetic, physiological, environmental and other differences between the populations of these countries, and give us more information about true performance of the machine learning models.

2.2 Machine Learning Setup

The general design of the experiments was setup with random forest as it has shown an overall better performance among common classification methods (8). All the datasets were concatenated into a single CSV file with labels for the datasets, and this file was used for further steps as shown in Figure 1. The analysis performed was two-fold, one without any preprocessing on the data, and two with applying Centered log-ratio (CLR) on the data. Application of a compositional normalization method like CLR was done to help us understand the consequences when compositional nature of the data is not taken into consideration.

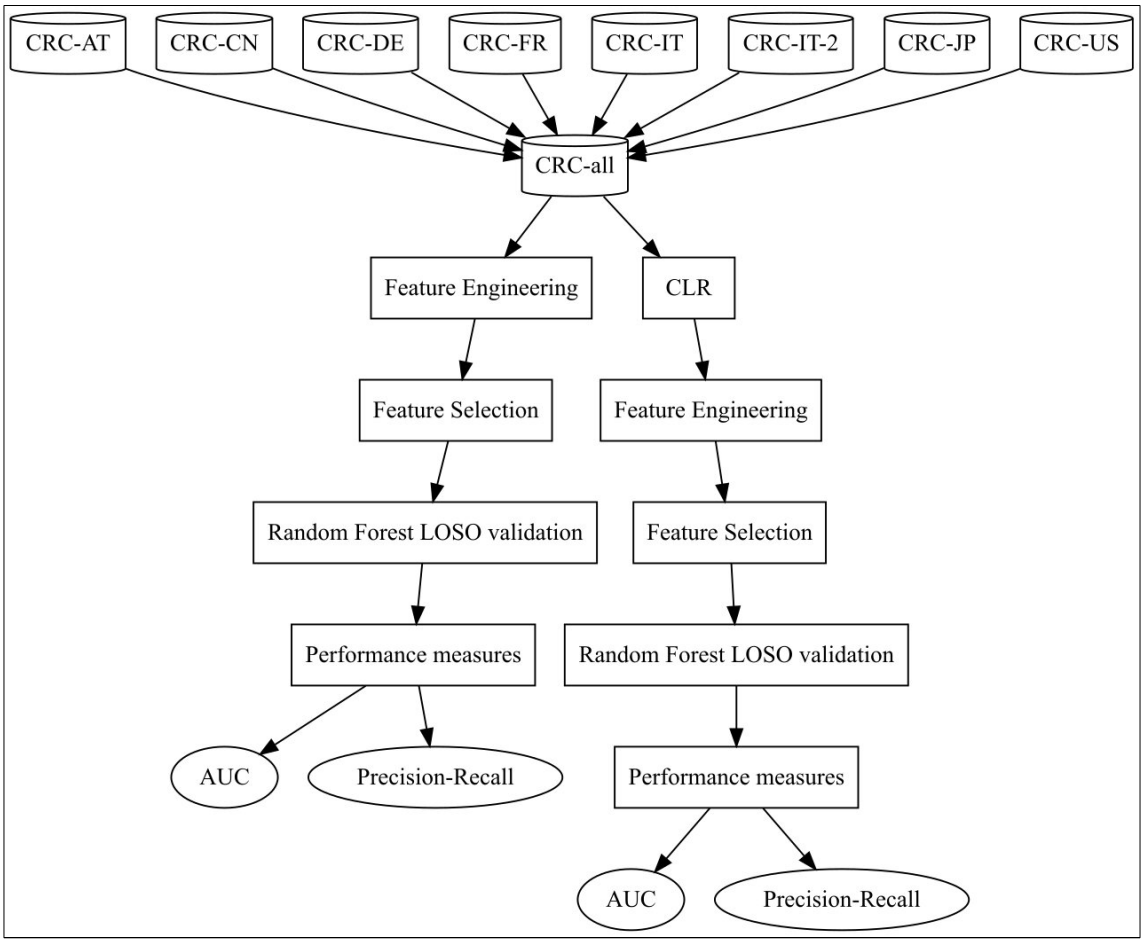


Figure 1: General machine learning methodology followed

To apply CLR transformation to the data, we first had to replace zeros in the dataset which was performed using zCompositions method *cmultRepl* which uses Bayesian multiplicative replacement (11). This was followed by CLR transform implemented using robCompositions method *cenLR* (12).

All methods were validated using a leave one study out approach as shown in Figure 2. Here, a nested cross validation structure is adapted to use each dataset once as a test set,

while every other dataset is used as a train and validation set. Using this technique, we get AUC scores and precision-recall values for each dataset when it used as a test and also as validation.

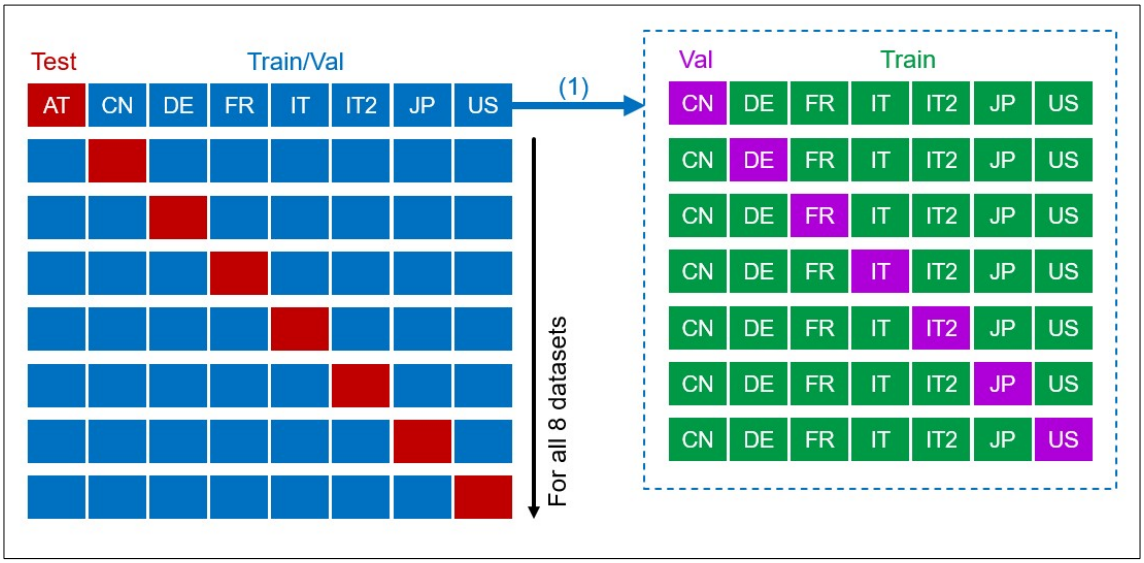


Figure 2: Leave one study out validation framework

Random forests (RF) is an ensemble method where a large number of decision trees are generated from the data, and in a classification problem the probability predictions of these trees is averaged to make a class prediction (13). Random forests was implemented using the scikit-learn 1.1.3 RandomForestClassifier method (14). The parameters used are -

- number of trees = 500
- fucntion to check quality of split = Gini impurity
- number of features to look for best split = square root of n features
- bootstrap samples = True
- out-of-bag estimates = False

The parameters were adapted from the work by Pasolli *et al.* (8), where RFs have shown significant performance on disease prediction from microbiome data. The performance of the models was evaluated using Area Under the Curve (AUC) value and Receiver operating characteristic (ROC) curve, also using scikit-learn 1.1.3 (14).

2.3 Feature engineering methods

The following feature engineering methods were applied to both, data with preprocessing using CLR and without.

2.3.1 Control performance measures

This analysis was performed to setup a baseline for comparison of all the methods implemented. Here, we used the data in its original form and also after applying CLR to perform a LOSO validation, using random forests for classification. The train/validation/test split was performed according to the protocol shown in Figure 2. All the features were used in the training, validation, and testing of data. The performance measures AUC and ROC curve were obtained as results.

2.3.2 Feature summation

The initial idea with this feature engineering method was to perform feature extension by summing the values of all possible pairs of features, and creating new features as shown in Figure 3. Then the new features would be reduced in size using feature selection, followed by classification. But the first step in the process generated a dataset which was ~2.0 gigabytes in size, and further processing like feature selection was not feasible even with available High Performance Computing resources.

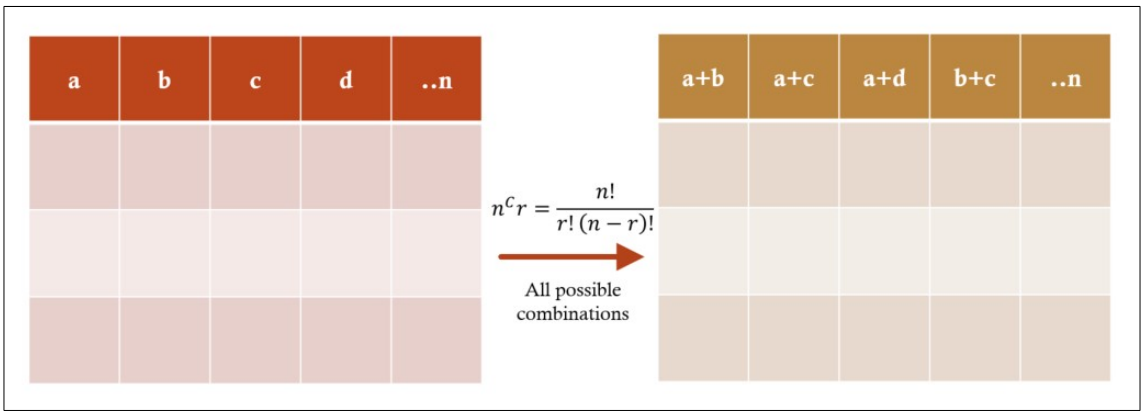


Figure 3: Feature extension by summation

As a solution we came up with was a modified implementation of feature summation by clustering. The original features of the CRC-all dataset were clustered using agglomerative hierarchical clustering from the SciPy toolkit 1.9.3 (15) . The function used is called *linkage* and the parameters applied are:

- Metric for calculating distance = Euclidean
- Method for assigning clusters = Ward variance minimization algorithm

The function *fcluster* was then used to make 20 clusters, and the features in each cluster were summed to make a dataframe with 20 features and all the samples. The importance of these new features was checked using Mutual Information (MI) with the *mutual_info_class* method from scikit-learn 1.1.3 (14). This function calculates MI using k-nearest neighbours which allows for estimation of shared information between continuous and discrete variables (16). The features with importance > 0 were selected to perform classification using random forests LOSO validation.

2.3.3 Symbolic regression

Symbolic regression is a supervised machine learning method, where the idea is to search through the entire parameter space of mathematical operations between the variables, to find a combination that accurately predicts the outcome. The tool that has been applied here, Feyn from Abzu is inspired by Richard Feynman's path integral formulation in quantum physics. The most optimal path is determined using probabilities, which are updated as the program iterates through the possible parameter space, and the fitness of a path is assessed using a loss function (17).

To better understand its working, Feyn was run on a subset of the CRC-all dataset. The program was run using the *auto_run* module on a classification task for 50 epochs. The best model obtained is displayed as a diagrammatic representation, along with the train and test metrics in Figure 4 and Figure 6.

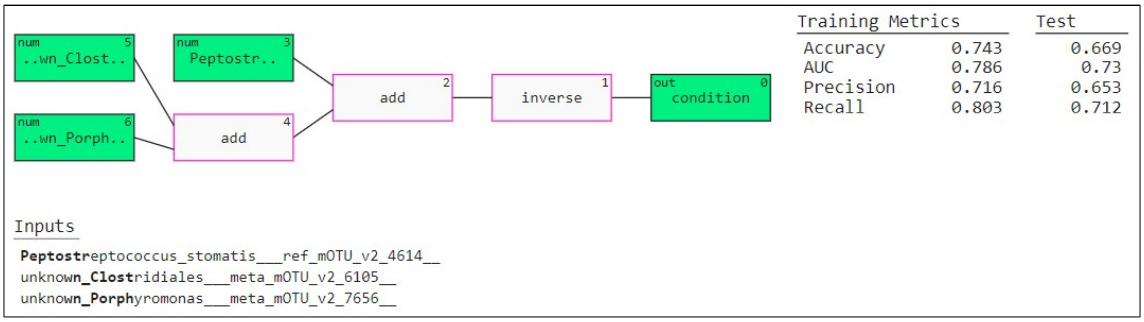


Figure 4: Model derived by running Feyn on a subset of CRC-all

They also provide a tool to convert the model into a mathematical equation using the SymPy library for better interpretability as shown in Figure 5.

$$\text{logreg}\left(-2.89 - \frac{0.951}{\begin{array}{l}-1060.0\,Peptostreptococcusstomatisref mOTUv_{24614} - 933.0\,unknownClostridialesmetamOTUv_{26105} \\ - 432.0\,unknownPorphyromonasmetamOTUv_{27656} - 0.241\end{array}}\right)$$

Figure 5: SymPy representation of model

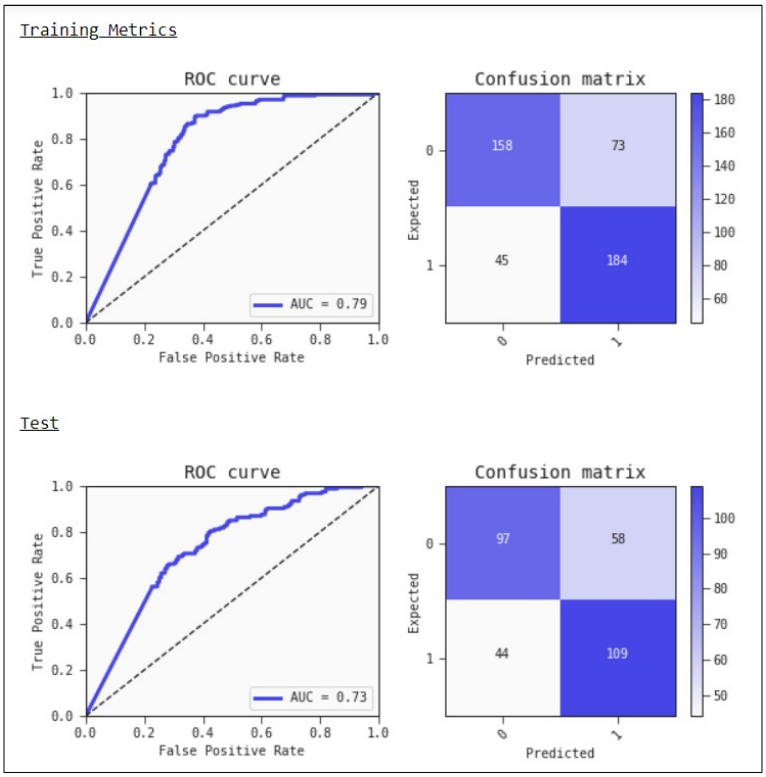


Figure 6: Train and test metrics for Feyn model

Feyn intrinsically performs the assigned regression or classification task using the obtained models. Hence this tool was applied in the LOSO framework without random forests to obtain the same evaluation metrics as the other methods, for ease of comparison. This process was followed for both, data with CLR and without. The parameters used are:

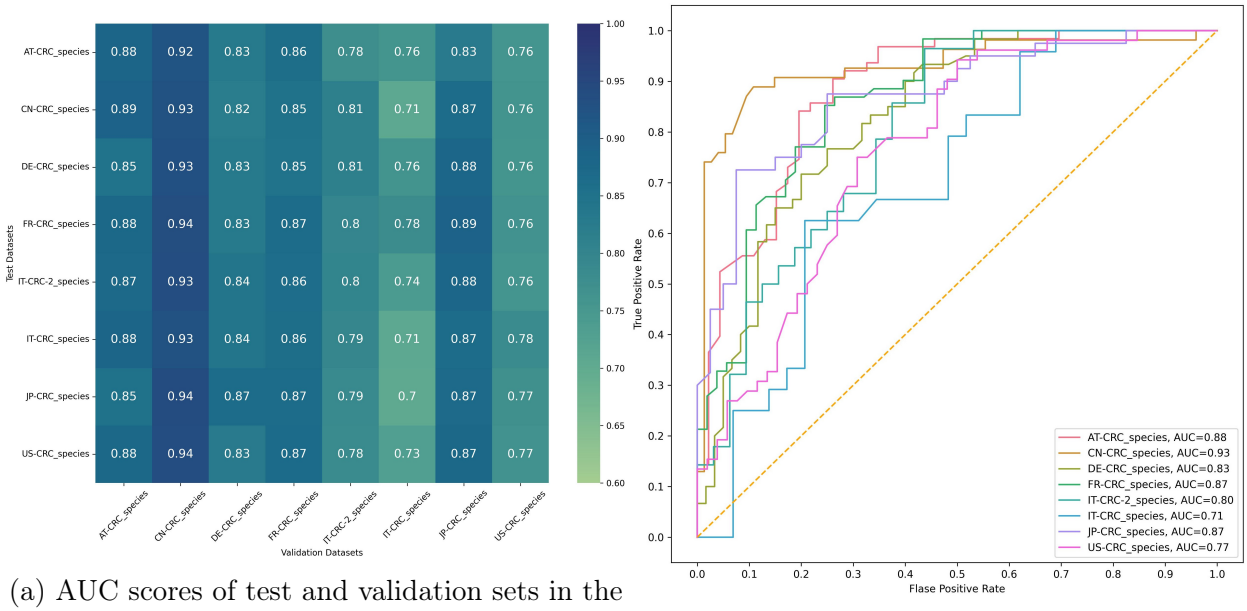
- Task = Classification
- Loss function = Binary cross entropy
- Epochs = 10 in train/validation loop, 50 in test loop

3 Results

3.1 Control

3.1.1 Without CLR

First we performed a baseline analysis using the data in its original form. The LOSO validation resulted in different AUC values for the test and validation sets as shown in Figure 7a. Here the diagonal values are the test set AUC corresponding to those shown in Figure 7b. The highest AUC obtained is for the Chinese dataset (CN-CRC) with 0.93 and the lowest is for the Italian-1 dataset (IT-CRC) with 0.71. The ROC curves in Figure 7b show that the classifiers perform with a lot of variablility among them, some even being close to the random classifier decision boundary.



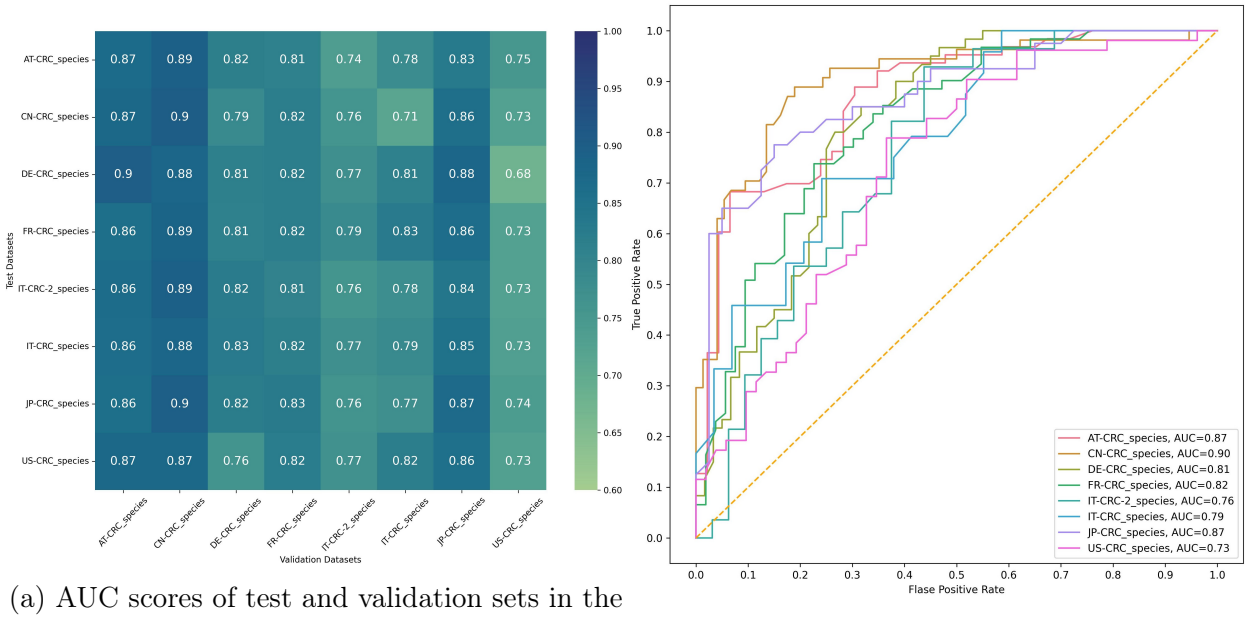
(a) AUC scores of test and validation sets in the LOSO framework (Diagonal values are test AUC and non-diagonal values are validation AUC)

(b) ROC curves of test datasets from LOSO framework

Figure 7: Results for control on data without CLR

3.1.2 With CLR

The data was CLR transformed as described in subsection 2.2, before performing LOSO validation. As seen in Figure 8a, AUC for the Chinese dataset (CN-CRC) is still the highest with 0.90 but lower than that seen in the previous section. Here the lowest AUC obtained was for the American dataset (US-CRC) with the value 0.73. But on an average, the ROC curves in Figure 8b are closer together than in Figure 7b, implying there is lesser variance among the performance of the model between datasets.



(a) AUC scores of test and validation sets in the LOSO framework (Diagonal values are test AUC and non-diagonal values are validation AUC)

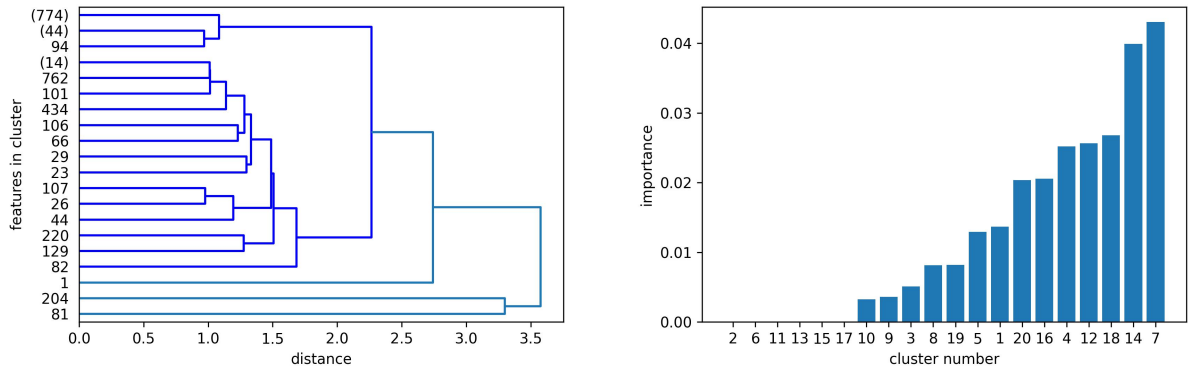
(b) ROC curves of test datasets from LOSO framework

Figure 8: Results for control on data with CLR

3.2 Feature summation

3.2.1 Without CLR

Hierarchical clustering was used to create 20 clusters in the unprocessed dataset, based on euclidean distance between the features. The clusters created varied greatly in number, with one cluster containing 774 features and many others only containing 1 feature each as shown in Figure 9a. The maximum distance between any two clusters was not more than 3.5. The features in each cluster were combined by summation to create a single feature for the cluster. The mutual information importance of these 20 clusters is shown in Figure 9b, of which 6 had importance value 0.

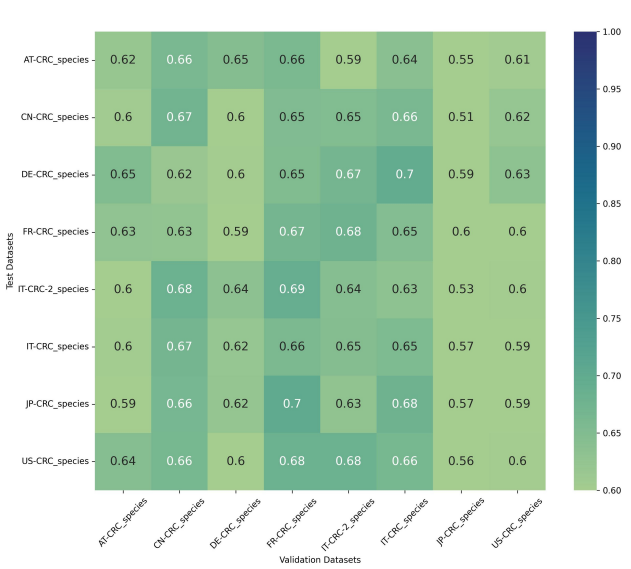


(a) Hierarchical clustering dendrogram

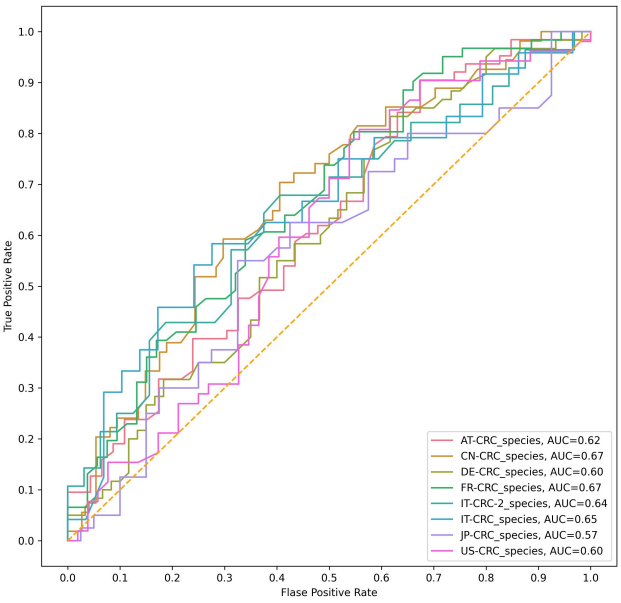
(b) Importances of new features created

Figure 9: Clustering of features without CLR

Only clusters with mutual information more than 0 were used for the LOSO validation. The resulting AUC values were much lower than seen in the baseline results as seen in Figure 10a. The highest AUC obtained for test sets was 0.67 for FR-CRC and CN-CRC and lowest was again JP-CRC with 0.57. The ROC curves in Figure 10b also determine that the classifiers were very close to being random estimators.



(a) AUC scores of test and validation sets in the LOSO framework (Diagonal values are test AUC and non-diagonal values are validation AUC)

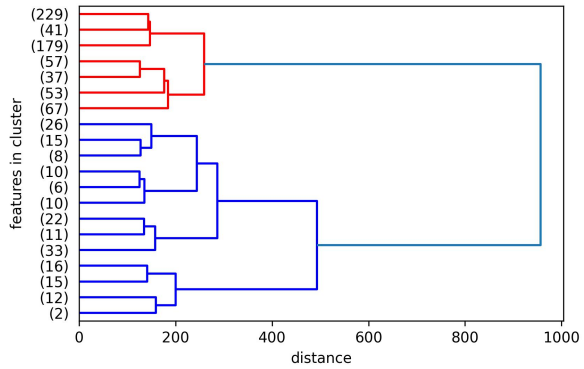


(b) ROC curves of test datasets from LOSO framework

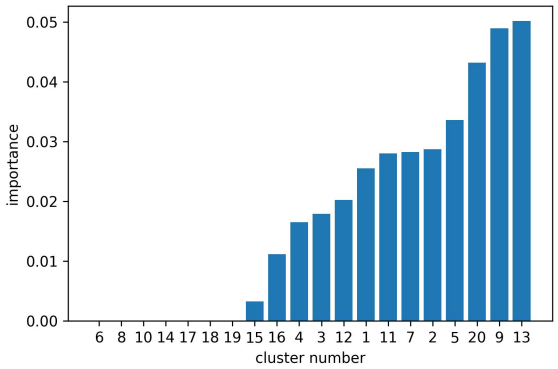
Figure 10: Results for feature summation on data without CLR

3.2.2 With CLR

Similar to the process described above, the CLR transformed data was hierarchically cumulated into 20 clusters. Here, the number of features in each cluster, as shown in Figure 11a were more evenly distributed as compared to those in Figure 9a. After summation of the features in each cluster, of the 20 clusters 7 showed importance value of 0 and were discarded before further analysis.



(a) Hierarchical clustering dendrogram

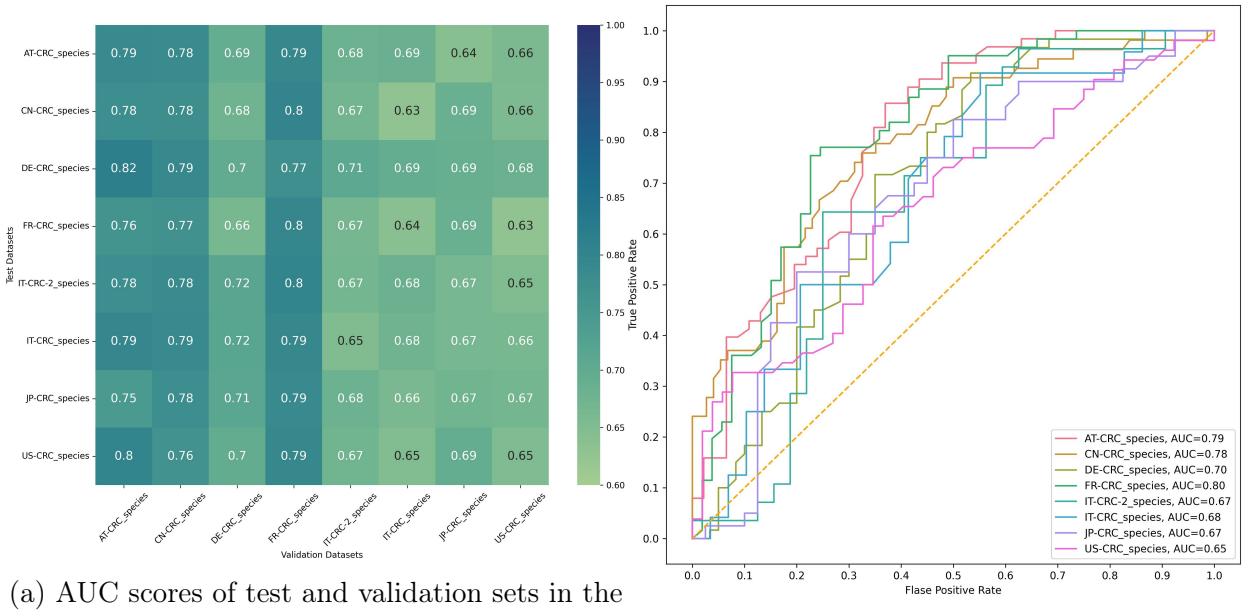


(b) Importances of new features created

Figure 11: Clustering of features with CLR

The AUC values obtained for this experiment were much higher than its counterpart without

CLR as shown in Figure 12a. We can see significantly improved performance for FR-CRC, AT-CRC and CN-CRC validation datasets, with highest performance at 0.8 AUC for FR-CRC test dataset. The lowest AUC is for the US-CRC dataset at 0.65. The ROC curves in Figure 12b are also closer to the upper left corner of the graph, which implies better performance as compared to Figure 10b.



(a) AUC scores of test and validation sets in the LOSO framework (Diagonal values are test AUC and non-diagonal values are validation AUC)

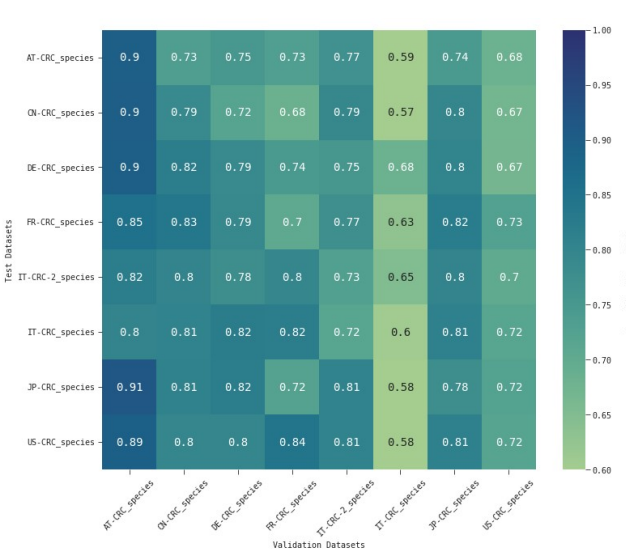
(b) ROC curves of test datasets from LOSO framework

Figure 12: Results for feature summation on data with CLR

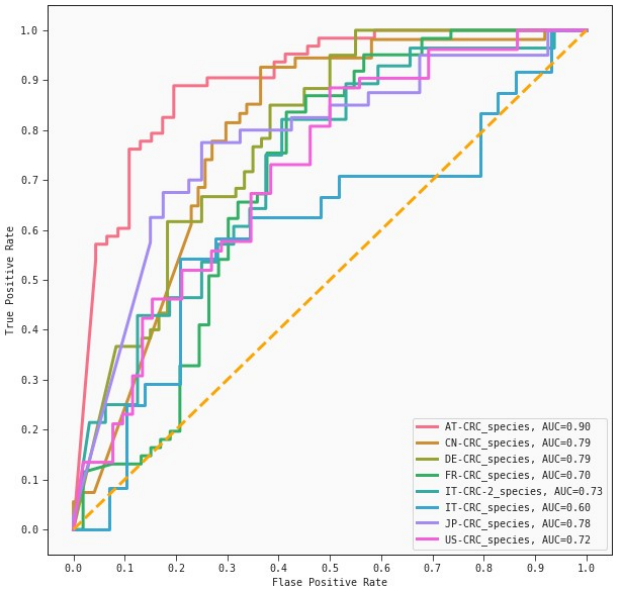
3.3 Symbolic regression

3.3.1 Without CLR

The Feyn library was used to implement symbolic regression on the datasets, followed by classification using the found model. Although random forest was not used in this experiment, the Feyn classification was structured similar to the previous analysis, to make the results comparable. The complexity of the models obtained ranges from 4 to 10 edges in the graph as depicted in Figure 4. The models use very few features (between 4 and 8) out of 850 from the dataset to classify the data, with a relatively high accuracy as seen in Figure 13a. IT-CRC seems to generally perform below average as a test and a validation set. The Austrian test set has the highest AUC at 0.9, and Italian the lowest with 0.6. The ROC curves of IT-CRC and FR-CRC in Figure 13b are very close to being random classifiers, while others show slightly better performance.



(a) AUC scores of test and validation sets in the LOSO framework (Diagonal values are test AUC and non-diagonal values are validation AUC)

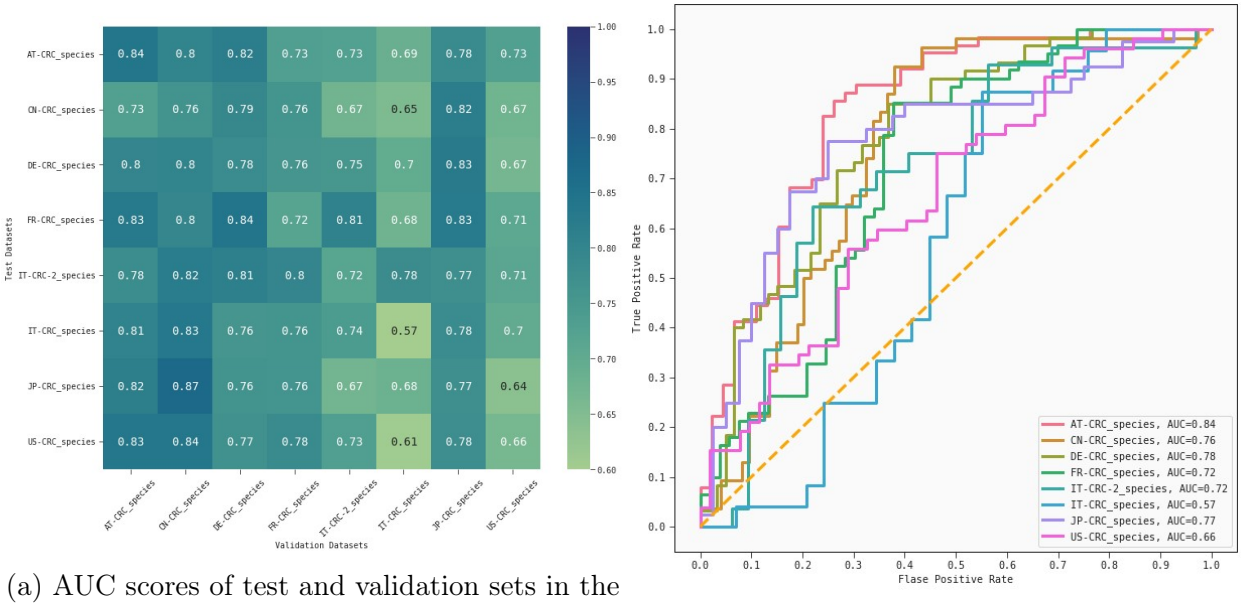


(b) ROC curves of test datasets from LOSO framework

Figure 13: Results for symbolic regression on data without CLR

3.3.2 With CLR

The same steps as subsubsection 3.3.1 were performed on CLR transformed data, and the resulting models seem to show slightly lower performance. The AUC for all test datasets is lower in Figure 14a than in Figure 13a, with the exception of FR-CRC. The first half of ROC curve for IT-CRC is below the random classifier decision boundary implying it is a bad classifier.



(a) AUC scores of test and validation sets in the LOSO framework (Diagonal values are test AUC and non-diagonal values are validation AUC)

(b) ROC curves of test datasets from LOSO framework

Figure 14: Results for symbolic regression on data with CLR

4 Discussion

4.1 Effect of compositionality and dataset size

It is evident from our results that the performance of a machine learning model on metagenomic data is affected by both size and compositionality. When the models are tested on a smaller dataset, we get low AUC as shown in Figure 15 for IT-CRC dataset which has only 53 samples. This implies that the model is overfitting on the training set and not generalizing well. But on the contrary, we also see low AUC for the US-CRC dataset which is relatively larger in size, which can be due to geographical and racial differences of human gut microbiomes, as proved in other studies (18). Since there is only one dataset from the US, the model may not be able to generalize for similar metagenomes.

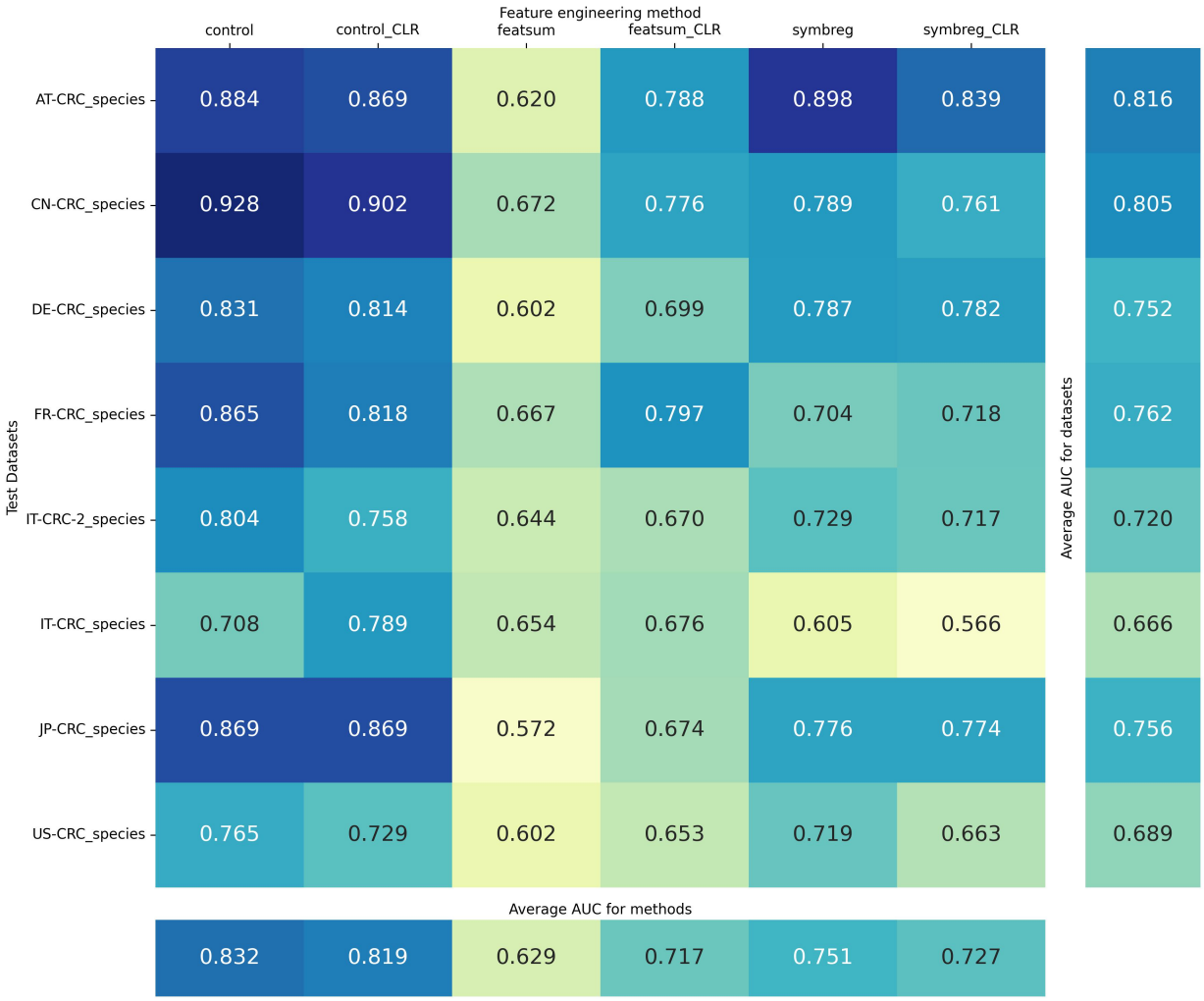


Figure 15: Comparison of AUC of test datasets from all methods

(**featsum**-Feature summation, **symbreg**-Symbolic regression, **CLR**-Centered Log Ratio transformed data)

Random forests performs better on the original dataset than on the CLR transformed dataset for the control group except for IT-CRC, where the CLR transformed data showed better results. It is possible that the data obtained from the Kaggle challenge has been preprocessed in some manner which has not been mentioned, and performing CLR on top of it takes some information away from the data.

4.2 Findings from different methods

The idea behind the original feature summation method described in subsubsection 2.3.2 was to find pairs of features that correlate directly to the outcome in some manner. Feature selection would have helped narrow down the new features, which would then be combined again in a similar manner, and tested against the outcome. This combination would be continued hierarchically to find the minimum number of features required to perform classification. In the adapted method, clusters are made based on distance measures, placing features that are closer to each other in a 2D space into the same cluster. Although computationally feasible, this method does not convey the same information as the original method. It is possible that features in the same cluster may not actually be correlated to the output together, hence summing them may obscure information carried by these features individually. This is evident in the AUC scores obtained for all the datasets as seen in Figure 15. It is noteworthy to mention that the performance of this method was better with CLR transformed data, probably because the extreme values in the dataset are normalized which reduces the distance between them. As seen in Figure 9a and Figure 11a, the clusters in CLR transformed data are more evenly distributed in terms of number of features per cluster, whereas in the original data one cluster consists of ∼91% of features.

Feyn, which has been used to perform symbolic regression with the dataset, has shown slightly lower performance than baseline, but higher than that of feature summation. The most fascinating aspect about this tool is that it has shown this performance using only ∼0.5% of the entire feature set. For each training set, the tool outputted a model which was a mathematical formulation of features similar to that shown in Figure 5. This gives the model high interpretability and transparency, unlike non-parametric models like random forests where it is difficult to understand *how* the outcome was obtained. These results also prove that very few features actually contribute to the output, and using symbolic regression helps us find them (19).

Another observation from Figure 15 is that Feyn performs slightly worse on CLR processed data, except for the FR-CRC dataset. This may be because Feyn finds interactions between the features that correlate to the output, and applying a log transformation on these features masks some of the information carried by them. As seen in the work by N. J. Christensen, *et al.* (20), this tool can be applied to multiple omics datasets with minimal pre-processing to obtain high performing models.

4.3 Conclusion

In this project we experimented with different feature engineering methods on microbiome data to find novel ways to reduce data size while preserving information. We also

evaluated the performance of these methods with and without compositional normalization of the data. We can conclude that the performance of a random forest classification model is affected by both compositionality and size of dataset. As for methods, we attempted a unique feature engineering method in this project and would like to further experiment with it when the resources are available. It will be interesting to investigate if we can find novel interactions with this method. The tool Feyn shows high potential when dealing with data that requires explainability, and can be further exploited to better understand the models it computes.

5 Code Availability

All the data and code used in this project can be found here -

https://github.com/siddhijain25/CHIP-project

6 Acknowledgements

I would like to thank my supervisor Ramtin Zargari Marandi, PhD for his guidance and continuous feedback which kept my inspiration fueled. I am grateful that you shared your expertise with me in the course of these three months, and I learnt so much working with you.

I would also like to thank my DTU supervisor Anders Gorm, without whose support this project would not be possible. I am thankful to have access to your knowledge and to receive feedback from you.

I am glad I had the opportunity to work on this project, and I hope to work on many more such projects in the future.

Thank you.

List of Figures

List of Tables

References

[1] L. Coughlan, P. Cotter, C. Hill, and A. Alvarez-Ordonez, "Biotechnological applications of functional metagenomics in the food and pharmaceutical industries," *Frontiers in Microbiology*, vol. 6, 2015.

[2] T. Thomas, J. Gilbert, and F. Meyer, "Metagenomics - a guide from sampling to data analysis," *Microbial Informatics and Experimentation*, vol. 20, no. 1, 2012.

[3] M. B. Pereira, M. Wallroth, V. Jonsson, and E. Kristiansson, "Comparison of normalization methods for the analysis of metagenomic gene abundance data," *BMC Genomics*, vol. 19, no. 1, p. 274, 2018.

[4] G. B. Gloor, J. M. Macklaim, V. Pawlowsky-Glahn, and J. J. Egozcue, "Microbiome Datasets Are Compositional: And This Is Not Optional," *Frontiers in Microbiology*, vol. 8, 2017.

[5] H. Lin and S. D. Peddada, "Analysis of microbial compositions: a review of normalization and differential abundance analysis," *npj Biofilms Microbiomes*, vol. 6, no. 60, 2020.

[6] M. C. Tsilimigras and A. A. Fodor, "Compositional data analysis of the microbiome: fundamentals, tools, and challenges," *Annals of epidemiology*, vol. 26, no. 5, pp. 330–335, 2016.

[7] G. Ditzler, J. Morrison, Y. Lan, and G. Rosen, "Fizzy: feature subset selection for metagenomics," *BMC Bioinformatics*, vol. 16, no. 358, 2015.

[8] E. Pasolli, D. T. Truong, F. Malik, L. Waldron, and N. Segata, "Machine Learning Meta-analysis of Large Metagenomic Datasets: Tools and Biological Insights," *PLOS Computational Biology*, vol. 12, no. 7, p. e1004977, 2016.

[9] M. Oudah and A. Henschel, "Taxonomy-aware feature engineering for microbiome classification," *BMC Bioinformatics*, vol. 19, no. 227, 2018.

[10] J. Wirbel, P. Pyl, E. Kartal, and et al., "Meta-analysis of fecal metagenomes reveals global microbial signatures that are specific for colorectal cancer," *Nature Medicine*, vol. 25, pp. 679–689, 2019.

[11] J. Palarea-Albaladejo and J. Martin-Fernandez, "zcompositions – r package for multivariate imputation of left-censored data under a compositional approach," *Chemometrics and Intelligent Laboratory Systems*, vol. 143, pp. 85–96, 2015.

[12] M. Templ, K. Hron, and P. Filzmoser, *robCompositions: an R-package for robust statistical analysis of compositional data*. John Wiley and Sons, 2011.

[13] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, 2001.

[14] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[15] "scipy.cluster.hierarchy.linkage." https://docs.scipy.org/doc/scipy/reference/generated/scipy.cluster.hierarchy.linkage.html. Accessed: 2022-11-27.

[16] B. C. Ross, "Mutual information between discrete and continuous data sets," *PLOS ONE*, vol. 9, no. 2, p. e87357, 2014.

[17] K. R. Broløs, M. V. Machado, C. Cave, J. Kasak, V. Stentoft-Hansen, V. G. Batanero, T. Jelen, and C. Wilstrup, "An approach to symbolic regression using feyn," *CoRR*, vol. abs/2104.05417, 2021.

[18] V. Gupta, S. Paul, and C. Dutta, "Geography, ethnicity or subsistence-specific variations in human microbiome composition and diversity," *Frontiers Microbiology*, vol. 8, p. 1162, 2017.

[19] C. Wilstrup and J. Kasak, "Symbolic regression outperforms other models for small data sets," *CoRR*, vol. abs/2103.15147, 2021.

[20] N. J. Christensen, S. Demharter, M. Machado, L. Pedersen, M. Salvatore, V. Stentoft-Hansen, and M. T. Iglesias, "Identifying interactions in omics data for clinical biomarker discovery using symbolic regression," *Bioinformatics*, vol. 38, pp. 3749–3758, 06 2022.

7 Appendix

Distribution of data