UNIVERSITY OF TÜBINGEN

MASTER THESIS

A Nextflow Pipeline for the Predicion of Phenotype based on Metagenomc Data

Author:
Matthias Marquardt

Supervisor:
Dr. Sabrina Krakau
Examiner:
Dr. Sven Nahnsen
Prof. Dr. Nadine Ziemert

A thesis submitted in fulfillment of the requirements
for the degree of Master of Science

in the

Quantitative Biology Center - QBiC
Informatik

March 19, 2021

Declaration of Authorship

I, Matthias Marquardt, declare that this thesis titled, “A Nextflow Pipeline for the Predicion of Phenotype based on Metagenomc Data” and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

UNIVERSITY OF TÜBINGEN

Abstract

Mathematisch Naturwissenschaftliche Fakultät
Informatik

Master of Science

A Nextflow Pipeline for the Predicion of Phenotype based on Metagenomc Data

by Matthias Marquardt

The Thesis Abstract is written here (and usually kept to just this page). The page is kept centered vertically so can expand into the blank space above the title too…

Acknowledgements

The acknowledgments and the people to thank go here, don’t forget to include your project advisor…

Contents

Declaration of Authorship **iii**

Abstract **v**

Acknowledgements **vii**

1 Introduction **1**
1.1 Metagenomics 1
1.2 Machine Learning and Reproducibility 1
1.3 Thesis Objective 1

2 Theoretical Background **3**
2.1 Metagenomics 3
2.1.1 File formats 3
2.1.2 Preprocessing 3
2.1.3 Taxonomic profiling 4
2.2 Machine Learning 4
2.2.1 Support Vector Machines 5
2.2.2 Random Forests 5
2.2.3 Gradient Boosted Trees 5
2.2.4 Evaluation of Machine Learning algorithms 5
2.3 Workflowmanagers and Reproducibility 6
2.4 Previous work 6

3 Experimental Setup **7**
3.1 Design considerations 7
3.2 Pipeline architecture 7
3.2.1 Preprocessing 7
3.2.2 Taxonomic profiling 8
3.2.3 Training mode 8
3.2.4 Prediction mode 8
3.3 Datasets 10

4 Results and Data Analysis **11**
4.1 ML evaluation 11
4.2 Performance of taxonomic profilers 11
4.3 Influence of taxonomic level 11
4.4 Feature selection 11
4.5 Runtime measurements 11

5 Conclusion and Outlook **13**
5.1 Possible Extensions 13

Bibliography **15**

List of Abbreviations

LAH	**L**ist **A**bbreviations **H**ere
WSF	**W**hat (it) **S**tands **F**or

Chapter 1

Introduction

1.1 Metagenomics

Since the human genome project, where in a massive collaborative effort the first human genome has been sequenced and assembled, Next generation sequencing (NGS) methods have taken over. NGS methods like sequencing by synthesis, which is used in Illumina sequencing machines, enabled the creation of huge amounts of data.

The general importance of metagenomics has just become clear in the recent years. Certain illnesses are not only determined by genome and lifestyle but also by the microbiome composition of affected body parts.

For example some species and strains are implicated in causing inflammatory bowel disease (IBD)

1.2 Machine Learning and Reproducibility

Recent advances in computing power enabled the extensive use of Machine Learning (ML) algorithms. Frameworks like **scikit-learn** have become better and popular (Grisel et al., 2020).

Algorithms like Support Vector Machines (SVM), Random Forest (RF) and Gradient-Boosted Trees (XG-Boost) have been successfully used to predict illnesses based on metagenomic data before. These studies lack a focus on reproducibility and do not have best practices in mind.

(Chen and Guestrin, 2016)

1.3 Thesis Objective

Exploration of publicly available datasets

Create a pipeline which makes conducting Machine Learning experiments on metagenomic data for phenotype prediction reproducible. The pipeline should be designed following best practices so potential users do not have to focus on this.

Chapter 2

Theoretical Background

2.1 Metagenomics

Real world data is ugly. It contains a lot of noise. Microbiome reads can be contaminated with host DNA and with adapters from sequencing. In order to clean up the data preprocessing has to be performed.

2.1.1 File formats

Sequencing is the technique of translating the succession of nucleotides into a string. In order to save this digitized DNA information two standard file formats have prevailed: FASTA and FASTQ.

FASTA is a relatively simple format. It uses two lines per sequence. A read begins with the character `>` followed by the name of the read (unique identifier), a newline character `\n` and then the sequence of the read which is usually ended with a newline character, then this can be repeated in case of multiple reads. The DNA sequence usually is encoded with the letters A,C,T and G (can also be lower case) for the bases Adenine, Cytosine, Thymine and Guanine. The general file suffix is .fasta. The FASTA file format can also represent amino acid so in order to distinguish the two the recommended suffixes are .faa for amino acids and .fna for nucleotides. FASTQ encodes the sequence as well as its corresponding quality score. Per sequence 4 lines are used. The first line begins with the `@` character followed by the sequence identifier (systematic use with Illumina, encodes machine used, lane, etc). The second line contains the raw sequence. The third line starts with the `+` character optionally followed by the same sequence identifier as in used the first line. The fouth line contains the encoded quality scores corresponding to the second line, therefore it has to have the same number of symbols than the sequence itself. The quality scores are encoded using the order of ASCII (American Standard of Code Information Interchange) characters. Starting with the 33rd ASCII character `!` representing lowest quality the original encoding goes up to character 126 `~` as the highest encoded quality score possible. Today the most widely used encoding is called Pred+33 (i.e. used by Illumina). It only encodes up to character 74 `J` as highest quality score.

2.1.2 Preprocessing

The quality information in a FASTQ file can be leveraged to only keep part of a sequence which are reliable. If the quality score falls below a set threshold reads can be cut so that only meaningful data is used.

When using microbiome reads there is always the possibility of reads still containing host DNA. In order to clean up all the reads are compared to a library of the host genome and matches are removed. In Illumina sequencing the PhiX library can used as an internal standard. It is a library of well maintained and characterized phage genomes. They are not always removed correctly when using publicly available datasets. In order to ensure no contamination of the reads they can be removed by comparing to the known genome.

2.1.3 Taxonomic profiling

After preprocessing the more interesting questions can be asked. What kind of microbes are in the sample?

In order to classify groups of organisms and display their hierarchy biologists use the taxonomic rank. The main taxonomic ranks from generic to specific are: domain, kingdom, phylum, class, order, family, genus and species. For metagenomics the genus and species level are used the most. In 2020 3,433 (± 115) bacterial genus names were accepted (Rees et al., 2020). Strains are subtypes of species which is extremely useful for a even more fine grained resolution of the sample composition.

Taxonomic profilers use preprocessed reads to estimate the relative composition of microbes in the probe. There are different kinds of taxonomic profilers, MetaPhlan3 and kraken2 are two widely used examples.

MetaPhlAn3 (**Meta**genomic **Phyl**ogenetic **An**alysis) is the third iteration of a taxonomic profiling software from the Segata Lab. It is reference based, which means it uses the coverage of clade specific marker genes in a sample to estimate the taxonomic composition. The marker genes are chosen such that all strains in a clade posses them while other clades contain no orthologes which are close enough to match as well. (Beghini et al., 2020) MetaPhlAn3 contains 1.1M marker genes from around 13.5 k species.

kraken2 is reference based and uses a huge database to compare the sample to. From the query sample k-mers are extracted and matches in the database to assign the Lowest Common Ancestor (LCA). This information is used to estimate the relative species composition of the sample. (Wood, Lu, and Langmead, 2019)

2.2 Machine Learning

In order to get meaningful results with ML problems have to be framed correctly. The goal of a study determines the methods and algorithms that have to be used. What is the objective has to be one of the first questions asked. Is it sufficient to put samples in two or more distinct categories or are the samples differing in severity? The first would be a classification problem while the second case would be a regression problem. Another important distinction of ML algorithms is weather they are using supervised or unsupervised learning. This is dependent on the data as supervised algorithms need clearly labeled training data. The training data is used to optimize the performance so that unseen data can be categorized correctly by the algorithm. Unsupervised learning uses untagged data where algorithms try to find underlying patterns.

A pitfall to be aware of when using supervised learning problems is overfitting. The internal model used is to complex. This means that the algorithm has learned specifics of the training data and it is not generalizing well. It then has problems predicting new and unseen data.

Cross-validation is a technique that can be used to asses the generalization of an given algorithm. It can thus be used to prevent overfitting. Data is split in training and validation set, this can be done multiple times following certain schemes.

Feature selection can also be used to prevent overfitting. It can shorten training times and help to avoid the curse of dimensionality.

Implementations of ML algorithms usually need more input than the training data, they have so called hyperparameters which need to be adjusted in order to optimize the results. Each algorithm has its own set of hyperparameters and the tuning of them is a very important but time intensive step.

In recent years a wide range of ML algorithms has been developed. The following sections describe some of the most widely used algorithms for supervised classification.

2.2.1 Support Vector Machines

Suppose a dataset with two classes. In order to classify the data points a SVM will try to construct a hyperplane which separates the data points of those two classes. There can be many ways that a hyperplane separates the data so a reasonable choice is to maximize the margin, which is the distance between the data points and the hyperplane. Nonlinear data can be classified with a SVM as well. To enable this the kernel trick is needed. The feature space is transformed to a higher dimension which makes it possible to fit a linear hyperplane. For this polynomial or Gaussian radial basis function kernels can be used.

2.2.2 Random Forests

Random forests are a ensemble learning method taking multiple decision trees and averaging their results.

Individual decision trees tend to overfit the training data. RF learning averages over multiple deep trees trained from different parts of data set. This reduces variance, but leads to a small increase in bias and a marginal decreased interpretability compared to individual decision trees. Generally the interpretability is a wished feature and it is still given through the variable importance.

2.2.3 Gradient Boosted Trees

Ensemble method boosting: multiple weak learners into a strong one iteratively learning weak learners, adding to the strong learner after adding data weights are readjusted (re-weighing) misclassified input gets higher weight so future learners focus more on them weak learners are added using a gradient descent optimization algorithm

2.2.4 Evaluation of Machine Learning algorithms

Different metrics have been proposed and are being used to optimize and evaluate the effectiveness of ML algorithms. Which of these are best is not clear. There are no standards yet as the importance of respective advantages and disadvantages are still being debated. It is better to use many metrics and not completely rely on one.

2.3 Workflowmanagers and Reproducibility

In order to generate meaningful results preprocessing, profiling, training and prediction have to be brought together. For this different scripts have to be executed sequentially. For one dataset this is fairly straight forward. But for multiple datasets it can get repetitive and error prone as the same scripts have to be executed over and over with slightly different parameters. Workflow management frameworks can be used to build pipelines which automate the execution of scripts and thus help to avoid time costly errors. In combination with containers they can enable experiments being conducted independent of the hardware used making them portable. Using a workflowmanager is a big step towards reproducibility as the execution of a pipeline with the same parameters should yield the same result. This only holds if there inherent randomness in the pipeline. For machine learning this means that only CPU powered learning can be truly reproducible and only if all seeds are set to fixed values.

A lot of workflow management frameworks have been developed and new ones are being published continuously. Snakemake and Nextflow are among the most popular as they have strong community support. Both are Command Line Interface (CLI) programs and support the usage of containers as well as version control systems like git. They use their own Domain Specific Language (DSL) as extensions of the underlying programming language, Groovy for Nextflow and Python for Snakemake.

Snakemake executes operations based on a directed acyclic graph (DAG). Each step is represented by rules which describe the handling of input and output files. The DAG is derived inferred from the set of all rules (Mölder et al., 2021).

In the Nextflow dataflow model operations are executed in their own isolated process. The output of a process then is streamed through a channel into processes dependent on the previous output. This enables parallel execution of tasks and a high scalability (Di Tommaso et al., 2017). Around Nextflow a vivid community has developed with nf-core. Best practices pipelines for a wide range of use cases have been developed and are being curated. They are easily accessible and have extensive documentation. Questions can be asked in the very active Slack workspace. A starter template helps developers following best practices (Ewels et al., 2020).

2.4 Previous work

strengths and weaknesses reasoning and importance of my work, placement in field of research -> no pipeline yet, stuff is not reproducible

Chapter 3

Experimental Setup

3.1 Design considerations

One of the main goals of this master thesis was to design a Nextflow pipeline performing reproducible machine learning. There are some research questions which influence the design choices taken. With MetaPhlAn3 and kraken2 two different taxonomic profilers have been implemented. Other taxonomic profilers could be added in the future as well. This flexibility enables potential users to chose the best fitting profiler for their task or experiment if unsure.

Another important aspect is the taxonomic level on which the ML algorithm is learning its patterns. Depending on the illness genus, species, strain or a combination of species and strain is the more important. The user can decide which of these levels to looked at, or even use all of the above in order to learn which the most distinguishing levels for a certain illness are.

There are different ML algorithms implemented. The user can chose between SVM, RF, XG-Boost and L2-Linear Regression or take some or all of them. Weather a feature selection should be performed can be indicated with its corresponding threshold value. The metric on which the algorithm optimizes is customizable.

3.2 Pipeline architecture

The Nextflow pipeline developed within this master thesis takes metagenomic data as input and trains a phenotype prediction algorithm with this information. It then can be used on unseen data for prediction. Figure 3.1 shows a schematic overview of the underlying architecture. The pipeline has two distinct modes: training and prediction. Both modes share a common processing steps. Reads are trimmed based on their quality, host contamination is removed, taxonomic profiling is performed and summarized.

3.2.1 Preprocessing

Quality based trimming is performed with fastp (Chen et al., 2018). The 3′-end and the 5′-end are trimmed if the quality score in a sliding window falls below a preset threshold. Fastp is executed with these parameters:

```
fastp -q mean_quality --cut_by_quality5 --cut_by_quality3 --cut_mean_quality
trimming_quality
```

The quality score in the sliding window is set with the pipeline parameters `trimming_quality` and `mean_quality` which have a default value of 15.

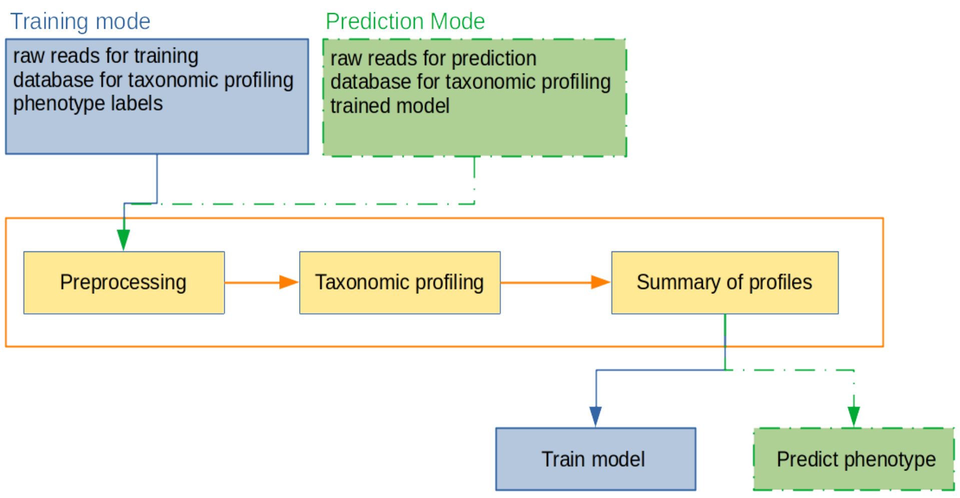

FIGURE 3.1: Schema of the pipeline architecture. Train mode is blue colored, predict mode is green and the shared components are colored yellow.

If host DNA is present it can be removed with bowtie2 (Langmead and Salzberg, 2012). To achieve this a fasta file of the host has to be provided. bowtie2 then finds and removes matches of the host DNA in the samples.

3.2.2 Taxonomic profiling

By default both MetaPhlAn3 and kraken2 are used for taxonomic profiling. `--skip_metaphlan` and `--skip_kraken2` can be specified on pipeline execution when just one profiler is wished. The databases are downloaded and configured automatically. If they are already present on the executing system their path can be specified with `--kraken2_db "$PATH"` or `--metaphlan_db "$PATH"` where `"$PATH"` is a placeholder for the database path on the executing machine.

For each taxonomic profiler a summary table of all sample profiles is created with a custom python script using the pandas framework (Reback et al., 2020).

3.2.3 Training mode

Figure 3.2 shows the data splitting scheme which is used during training mode. A custom python script uses the previously created summary table of taxonomic profiles to train the specified classifiers. All of the classifiers are implemented either in the scikit-learn or XGBoost packages.

Finally mean values of ROC, (Balanced) Accuracy, Precision, Recall, F1-Score and MCC are reported together with the runtime. The trained classifiers are pickled and saved so that they can be used in the prediction mode.

3.2.4 Prediction mode

For prediction mode one or more pretrained classifiers and the samples have to be provided. The same preprocessing and taxonomic profiling which has been used in training mode is performed on the samples. For the classification a python script uses the summary table of the taxonomic profiles and the classifiers yielding a prediction.

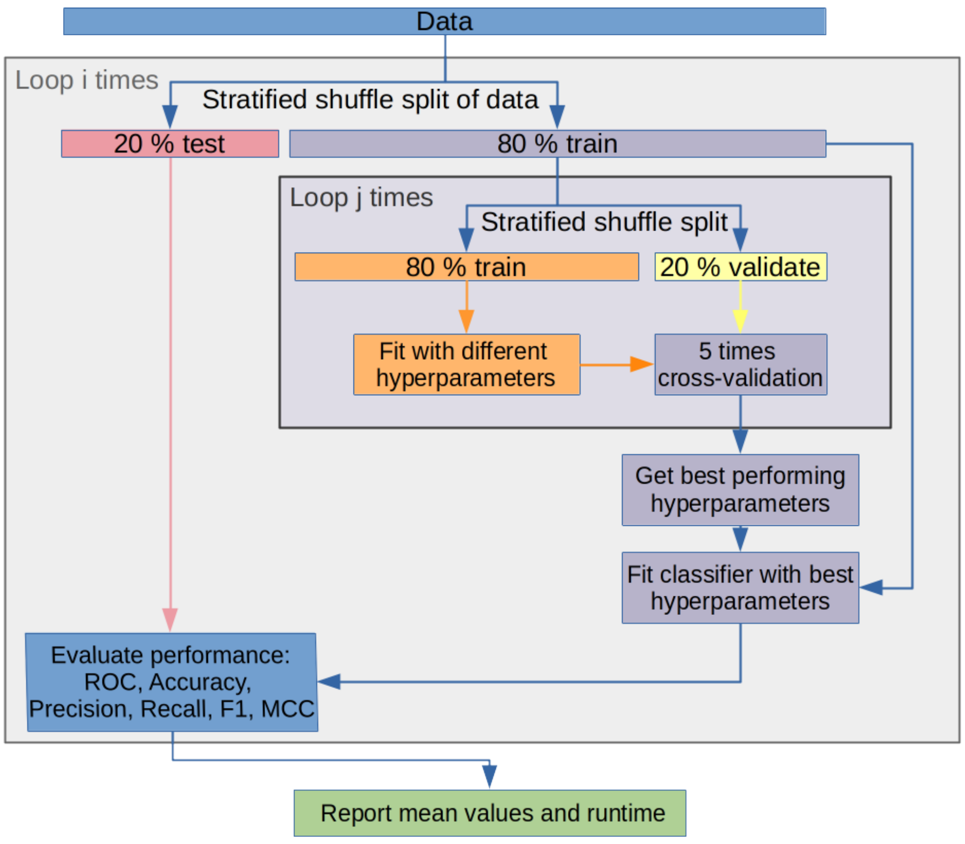
Data
Loop i times
Stratified shuffle split of data
20 % test
80 % train
Loop j times
Stratified shuffle split
80 % train
20 % validate
Fit with different hyperparameters
5 times cross-validation
Get best performing hyperparameters
Fit classifier with best hyperparameters
Evaluate performance: ROC, Accuracy, Precision, Recall, F1, MCC
Report mean values and runtime

FIGURE 3.2: Data splitting schema for testing and training. A stratified shuffle split is performed i-times on all of the data, respectively putting 20% into the testing and 80% in training set. Each training set is split j-times into 20% validation and 80% hyperparameter-training set. Five times cross-validation is used with this validation and training set to get the best performing hyperparameters. The performance then is evaluated i-times on the original test/train splits. Mean values and runtime are reported.

TABLE 3.1: Cases and sample size of considered datasets. The size is given in Terabasepairs.

	Disease Cases	Healty Cases	Total Samples	Size [Tbp]
Liver Cirrhosis	123	114	237	1.2
Obesity	169	123	292	1.6
IBD	25	99	124	0.39

3.3 Datasets

There are several benchmark datasets which have been used for phenotype prediction. Liver cirrhosis has yielded very good results before. Obesity is a hard case as researchers have struggled to get good performing predictions. Inflammatory Bowel Disease (IBD) is used as a smaller and imbalanced dataset yielding promising results. The number of samples and the size of the datasets is shown in table 3.1.

Chapter 4

Results and Data Analysis

4.1 ML evaluation

4.2 Performance of taxonomic profilers

4.3 Influence of taxonomic level

4.4 Feature selection

4.5 Runtime measurements

Chapter 5

Conclusion and Outlook

5.1 Possible Extensions

bracken mlf-core dsl2

Bibliography

Beghini, Francesco et al. (2020). “Integrating taxonomic, functional, and strain-level profiling of diverse microbial communities with bioBakery 3”. In: *bioRxiv*.

Chen, Shifu et al. (2018). “fastp: an ultra-fast all-in-one FASTQ preprocessor”. In: *Bioinformatics* 34.17, pp. i884–i890.

Chen, Tianqi and Carlos Guestrin (2016). “Xgboost: A scalable tree boosting system”. In: *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794.

Di Tommaso, Paolo et al. (2017). “Nextflow enables reproducible computational workflows”. In: *Nature biotechnology* 35.4, pp. 316–319.

Ewels, Philip A et al. (2020). “The nf-core framework for community-curated bioinformatics pipelines”. In: *Nature biotechnology* 38.3, pp. 276–278.

Grisel, Olivier et al. (May 2020). *scikit-learn/scikit-learn: scikit-learn 0.23.1*. Version 0.23.1. DOI: 10.5281/zenodo.3834804. URL: https://doi.org/10.5281/zenodo.3834804.

Langmead, Ben and Steven L Salzberg (2012). “Fast gapped-read alignment with Bowtie 2”. In: *Nature methods* 9.4, p. 357.

Mölder, Felix et al. (2021). “Sustainable data analysis with Snakemake”. In: *F1000Research* 10.33, p. 33.

Reback, Jeff et al. (June 2020). *pandas-dev/pandas: Pandas 1.0.5*. Version v1.0.5. DOI: 10.5281/zenodo.3898987. URL: https://doi.org/10.5281/zenodo.3898987.

Rees, Tony et al. (2020). “All genera of the world: an overview and estimates based on the March 2020 release of the Interim Register of Marine and Nonmarine Genera (IRMNG)”. In: *Megataxa* 1.2, pp. 123–140.

Wood, Derrick E, Jennifer Lu, and Ben Langmead (2019). “Improved metagenomic analysis with Kraken 2”. In: *Genome biology* 20.1, pp. 1–13.