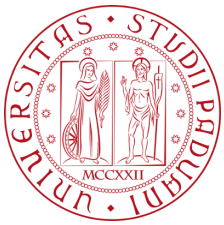

Università degli Studi di Padova
Dipartimento di Matematica "Tullio Levi-Civita"

Thesis

Towards Personalized Disease Risk Prediction from Metagenome Analysis of the Microbiome

Árpád Goretity
Data Science MSc
Matricola: 1205828

20 July 2020

Advisors:
Silvio C. E. Tosatto, PhD
Damiano Piovesan, PhD

Table of Contents

Abstract

The human gut microbiome is a central topic of research in bioinformatics and computational biology. The more and more widespread availability of high-throughput, whole-genome shotgun sequencing technologies makes it possible to obtain an unprecedented amount of genomic and metagenomic data. Various microbiota have been shown to correlate with health status and even with the early onset of diseases, the most prominent examples being Irritable Bowel Syndrome (IBS) and the more serious Inflammatory Bowel Diseases (IBD), such as Crohn's Disease (CD) and Ulcerative Colitis (UC). The desire naturally arises for predicting, based on the composition of the gut microbiome, whether patients suffer from or risk developing these conditions.

This is especially valuable in the context of a newly-arising industry: holistic, personalized, preventive health consulting. Such a data-centric approach to assessing future health risks opens up the possibility for combining metagenomic information with results of other 'omics (metabolomics, genetics, proteomics, etc.), in order to provide patients with a comprehensive and easier-to-understand picture of the effects of, and the associations between, various aspects of their lifestyle.

Most approaches to metagenomic analysis have so far focused on taxon-level resolution, quantifying taxa at the genus, species, or strain level. However, biological functions are not in a one-to-one correspondence with taxa: for instance, a certain species may (and does usually) fulfill more than one function, while the same function may be provided by more than one species.

In this Thesis, we develop a metagenomic analysis pipeline based on individual genes and families of homologous genes. In contrast with using a large database of thousands of complete prokaryotic genomes, our approach only requires sequence data for select individual genes, therefore it is less storage-demanding. Relying on state-of-the-art read alignment software, we demonstrate gains in processing speed, too. Finally, we also show that our approach performs only slightly worse than SHOGUN, a more resource-intensive state-of-the-art metagenome analysis toolkit.

We then apply both pipelines to several datasets and build machine learning models for classifying stool samples as either healthy or IBD/IBS. Finally, we analyze the association between gene families in both groups using the tools of network science applied to abundance correlation networks.

Sommario (Abstract in Italian)

Il microbioma intestinale umano è un argomento di rilevante importanza nell'ambito della ricerca nella bioinformatica e biologia computazionale. È stato dimostrato che vari microbiota sono in correlazione con lo stato di salute e persino con l'insorgenza precoce delle malattie: gli esempi più noti sono la Sindrome dell'intestino irritabile (IBS) e le più gravi malattie infiammatorie intestinali (IBD), come la Malattia di Crohn (CD) e la Colite ulcerosa (UC). Sorge dunque spontanea la volontà di provare a prevedere se i pazienti soffrono o rischiano di sviluppare questi problemi basandosi sulla composizione del microbioma intestinale.

Ciò è particolarmente utile nel contesto di un'industria emergente: consulenza sanitaria olistica, personalizzata e preventiva. Un tale approccio incentrato sui dati per valutare i rischi futuri per la salute apre la possibilità di combinare informazioni metagenomiche con risultati di altri campi di ricerca (quali metabolomica, genetica, proteomica) al fine di fornire ai pazienti un quadro più completo e facile da intendere riguardante gli effetti di vari aspetti del loro stile di vita.

La maggior parte degli approcci all'analisi metagenomica si sono finora concentrati sulla risoluzione a livello di taxon, quantificando i taxa a livello di genere, specie o ceppo. Tuttavia, le funzioni biologiche non sono in una corrispondenza uno a uno con i taxa: ad esempio, una determinata specie può (e di solito è quello che accade) svolgere più di una funzione, mentre la stessa funzione può essere fornita da più di una specie.

In questa Tesi, sviluppiamo una pipeline di analisi metagenomica basata su singoli geni e famiglie di geni omologhi. Contrariamente all'utilizzo di un ampio database di migliaia di genomi procariotici completi, il nostro approccio richiede solamente sequenze di singoli geni selezionati, e per questo è meno dispendioso in termini di memoria. Basandoci su un software di allineamento all'avanguardia, dimostriamo anche dei miglioramenti nella velocità di elaborazione. Infine, si può notare come il nostro approccio funzioni solo leggermente peggio di SHOGUN, un toolkit di analisi del metagenoma all'avanguardia che richiede più risorse.

Applichiamo quindi entrambe le pipeline a diversi dataset e costruiamo modelli di machine learning per classificare i campioni di feci come appartenenti a soggetti sani o affetti da IBD/IBS. Infine, analizziamo l'associazione tra famiglie di geni in entrambi i gruppi utilizzando gli strumenti di network science applicati alle reti di correlazione dell'abbondanza.

Résumé (Abstract in French)

La recherche du microbiome humain est actuellement un thème central de la bioinformatique et de la biologie computationnelle. La disponibilité de plus en plus répandue du séquençage haut débit (HTS ou NGS) nous permet d'obtenir une quantité de données jamais vue. De nombreux microbes sont en corrélation avec l'état de santé et également avec le commencement des maladies, les exemples les plus importants étant le syndrome de l'intestin irritable (IBS) et les maladies inflammatoires chroniques de l'intestin (IBD), y compris la maladie de Crohn (CD) et la rectocolite hémorragique (UC). La prédiction de ces maladies et du risque de les attrapper, basée sur la composition du microbiome, est donc fort désirable.

Ceci est particulièrement précieux dans le contexte d'un nouveau domaine : la médecine préventive personnalisée. Un point de vue «data-centric» nous permet d'intégrer les résultats des «omics» divers, comme par exemple la métabolomique, la génétique, et la protéomique, pour fournir aux clients une répresentation complète et facile à comprendre, sur l'association entre leurs santé et leurs mode de vie.

Jusqu'ici, la plupart des études du microbiome humain se sont concentrés sur les taxons, en quantifiant les microbes au niveau du genre, de l'espèce, et de la souche. Pourtant, les fonctions biologiques non correspondent pas de manière biunivoque avec les taxons.

Dans cette Thèse, nous allons développer un logiciel pour analyser la composition du microbiome, basé sur des familles de gènes homologues. En contraste avec l'application d'une base de données contentant plusieurs milliers de génomes complets, elle requiert donc seulement les gènes pertinents selon la littérature, alors elle demande moins d'espace disque. En utilisant un aligneur avancé de séquences, nous démontrons aussi une accélération. Puis, nous prouvons que notre modèle est seulement légèrement moins précis que SHOGUN, un logiciel de pointe pour l'analyse du microbiome.

Nous appliquons ensuite tous les deux logiciels a de diverses données, et formons des modèles d'apprentissage-machine pour classifier des échantillons soit comme «en bonne santé» soit «ayant une MII». Nous concluons enfin avec une analyse de la correspondance entre les familles de gènes en utilisant la théorie des réseaux, appliquée aux réseaux de corrélation.

Kivonat (Abstract in Hungarian)

Az emberi bélflóra kutatása a bioinformatika egyik fő jelenlegi fejlődési iránya. A több gigabyte adatot szolgáltató, ún. „shotgun" metagenom-szekvenálási technológiák egyre szélesebb körű elterjedése korábban elképzelhetetlen mennyiségű adathoz juttatja a kutatókat. Számos mikróbáról kimutatták, hogy meghatározóak vagy éppen indikátorok az egészségi állapotra és bizonyos betegségek korai fázisára nézve. Ez a legegyértelműbb az irritábilis bélszindróma (IBS) és a súlyosabb bélgyulladás (IBD), utóbbin belül például a Chrohn-betegség (CD) vagy a fekélyes vastagbélgyulladás (UC) esetében. Felmerül tehát az igény, hogy ezeket a betegségeket vagy a rájuk való hajlamot a bélflóra összetétele alapján felismerjük.

Ez különösen fontos egy új egészségügyi terület, a holisztikus, személyre szabott, preventív medicína esetében. Az adatközpontú szemlélet lehetővé teszi, hogy a különböző „omikák" (például metabolomika, genetika, proteomika) eredményeit összesítve a klienseknek egy átfogó, mégis könnyen értelmezhető képet adjunk az egészségük és az életformájuk egyes elemei közötti összefüggésekről.

A metagenomikai tanulmányok túlnyomó többsége ezidáig rendszertani megközelítéssel dolgozott, a nemzetség, faj, vagy alfaj illetve baktériumtörzs szintjén kvantifikálva a mikrobiom összetételét. A bélflóra biológiai funkciói azonban nem feleltethetők meg egyértelműen a taxonoknak: egy fajnak több funkciója is lehet, és fordítva, egy funkciót több faj is betölthet.

Jelen dolgozatban kifejlesztünk egy olyan metagenom-elemző eszközt, ami a rendszertani besorolás helyett homológ géncsaládokon alapul. Előnye, hogy több ezer faj teljes genomja helyett csupán az irodalom szerint releváns gének szekvenciáit igényli adatbázisként, így tárhelyigénye töredéke a taxonómiai alapon működő programokénak. Fejlett szekvenciaillesztő programok használatával futási időben is számottevő gyorsulást tudunk elérni a SHOGUN nevű metagenom-feldolgozó csomaghoz képest, a pontosság és a jóslóérték minimális csökkenése mellett.

Végül mindkét programot lefuttatjuk több adathalmazra, és az eredményként kapott abundancia-profilokat statisztikai és gépi tanulási modellek tanítására használjuk, amelyekkel a teszthalmazban lévő pácienseket klasszifikáljuk mint egészséges vagy bélbetegségben szenvedő. A profilokat alkotó változók (géncsaládok, fajok, funkcionális modulok) közötti korrelációkat ezek után a hálózatelmélet módszereivel is elemezzük.

1. Introduction

1.1. Setting the Scene

The last two decades have seen an explosion in computing power, as well as in new experimental techniques in molecular biology. Specifically, whole-genome shotgun metagenome sequencing and fast read alignment algorithms now allow researchers to obtain data about the human microbiome relatively quickly, and in a large enough quantity that is meaningful to analyze statistically. This naturally led to metagenomics taking a leading position within the field of bioinformatics.

Several other multidisciplinary areas of research followed this trend — or, more accurately, they developed in parallel. The word "omics" has become mainstream terminology, collectively describing fields like genomics, genetics, metabolomics, transcriptomics, and connectomics. The common theme of these disciplines is that they attempt to characterize biological systems in a wide context, taking a system approach instead of focusing on low-level description. As a result, scientists now need to be fluent not only in their principal area of interest, but they must also routinely handle tasks related to statistical analysis, scripting, and programming.

1.2. Tackling Inflammatory Bowel Diseases

Inflammatory Bowel Diseases (IBDs) such as Chron's Disease (CD) and Ulcerative Colitis (UC) are severe illnesses of the human digestive system. A related but less severe group of diseases is called Irritable Bowel Syndrome or IBS for short. Since these are directly related to the organs of the gut, studying them and trying to achieve early detection from metagenomic data is sort of a low-hanging fruit.

Yet, this problem is not nearly solved: the devil is in the details, and getting them right is crucial, as *many* things can go wrong during the planning and execution of a metagenomic study. Thus, microbial characterization and prediction of IBDs and IBS remain an active area of research.

1.3. Data-Oriented Risk Prediction

The emerging industry of personalized, preventive health consulting introduces new challenges for applied research. In particular, it is desirable to come up with quantitative risk indices for certain diseases. One way to solve this problem is to create simple, human-engineered metrics, such as some combination of the concentration of specific substances in the blood or the urine, e.g. sugar and cholesterol.

A more principled way to approach this problem, however, is to let the data speak for itself, and to that end, integrate as many kinds of data as possible, then use machine learning for extracting and exposing its hidden patterns. This is the method suggested by Knights et al. [1] The precursor of the development of production-quality risk metrics is to first model existing data and study their structure. Here we aim to address this need and carry out preparatory work for developing an IBD risk index, based on metagenomic data from the human gut microbiome.

Disclaimer. The present Thesis was written under the supervision of Medipredict, Ltd., a personalized health consulting company.

2. Inflammatory Bowel Diseases and the Gut Microbiome

The literature is rich in previous work characterizing the human gut microbiome with different goals. In this chapter, we provide an overview of some important and recent papers, which are relevant to our problem, and which we therefore built upon. These articles are either concerned with methodology and metagenome analysis workflows, or with the biological or pathological functions of taxa.

2.1. Current Best Practices in Microbiome Research

2.1.1. Recommended Workflows

Before jumping into metagenomic experiments or data analysis, it is critical to have at least a high-level understanding of the entire process. Knight et al. [2] describe an "optimal" workflow not only for metagenomics, shown in figure 2.1.

In addition, they summarize some technicalities that might often seem to be of secondary importance, but which are in fact key for the correctness and reliability for a study. Some of these are:

- When comparing healthy and ill patients and defining the control group, one should consider known variations in the microbial profile across countries and diets, as well as whether the subjects take any medications.
- Metadata should not be ignored.
- One should use manually-curated, high-quality, large databases. When researching narrow taxonomic or functional categories, it is best to reach for specialized databases. For instance, Resfams [3] is a database designed specifically for discovering antibioic resistance genes.
- There are a myriad of metagenome analysis tools to choose from, and one should not stick to just one for every kind of analysis. Pros and cons of every tool should be carefully inspected, and the software or model should be chosen based on whether it is adequate for the kind of data being analyzed.

For example, short reads require different treatment of long reads, and high error rate PacBio reads will need special treatment compared to low error rate Illumina reads. Knight et al. recommend toolkits such as MetaPhlAn-2, HUMAnN2, and MEGAHIT.

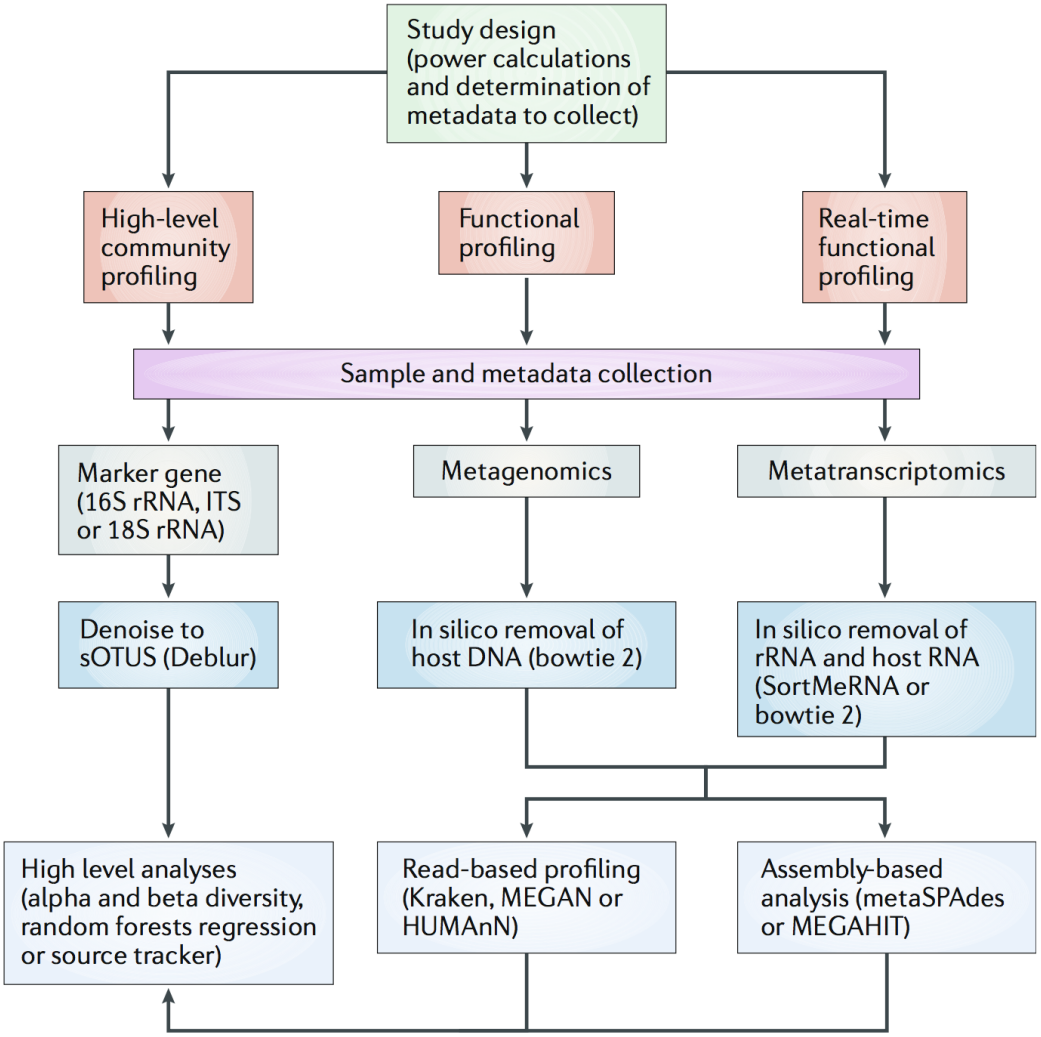

Figure 2.1. Recommended metagenomic analysis workflow.

2.1.2. Comparative Metagenomics

In "Toward Accurate and Quantitative Comparative Metagenomics", Nayfach et al. [4] assert that although metagenomics has a great potential of being quantitative, current experimental and analytical protocols are not in general being exercised satisfactorily. The paper lists a long set of potential pitfalls and sources of error, namely:

- Mixing of host DNA with microbial DNA

- Post-sampling environment (e.g. differences in how samples are being chemically treated, stored, or transported). They point to previous work showing that inter-laboratory variation is on the same order as biological variation.
- Non-uniform sampling of reads, due to DNA fragmentation or PCR biases
- Effects of bioinformatical preprocessing – although it is noted that these are among the least severe ones.

For the resolution of these problems, they make the following main suggestions:

- Evaluation of metagenomic data based on more meaningful and robust metrics, such as Average Genomic Copy Number, Cellular Absolute Abundance, or Gene Absolute Abundance.
- Compositional data should be scrutinized using specialized statistical methods, such as those proposed by Aitchison.
- Standardization and ease of access of data and metadata. The importance of metadata for repeatability and comparability across studies is re-emphasized. In particular, the authors state that most experiments and papers suffer from insufficient discoverability of metadata, and that it is hard and error-prone to parse metadata from journal supplements and link them to sequence libraries.

2.1.3. The State of the Art

In the paper "Current understanding of the human microbiome", Gilbert et al. [5] provide an overview of the field of human microbiome research. They state that the majority of studies to date have focused on factual description (the "what"), rather than mechanistic understanding (the "why"). Similar to other reviews, they also point out that the variation in microbial composition between individuals is very high, but they optimistically believe that this can even be considered an advantage in some situations, such as forensic applications.

The article also addresses the question of dynamics, i.e. the time evolution of the microbiome. The authors cite sources proving that the rate of temporal change is highly variable across individuals, and there is some (still limited) evidence that this is a clinical feature. This means that considering the time evolution of the microbiome would be a useful addition to studying its composition at only a single time point.

In addition, other factors influencing microbiome profiles and their behavior over

time are provided as well. Studies exist about the comparison of metagenomic samples from different body sites (such as skin, gut, and oral), while it is also shown that lifestyle patterns like co-habitation with pets or regular exercise also affect the composition of microbiota. Consequently, these factors should also be added to the metadata of metagenomic studies.

Furthermore, humans do not form discrete clusters based on their microbial profiles; instead, they are distributed across a continuum. It is thus foreseen that any attempts at classifying microbial samples will likely be met with difficult challenges.

2.2. Previous Results about IBD and the Gut Microbiome

2.2.1. Multi-omics of the Gut Microbiome in IBDs

Lloyd-Price et al [6] conducted a 1-year-long longitudinal and cross-sectional multi-omic study on 132 subjects. Their focus was mainly on dysbiosis during inflammatory bowel diseases. In the metagenomic analysis, samples with taxonomic profiles highly unlike those of non-IBD patients were defined to be "dysbiotic". For a comprehensive exploration of the data, several measurements were made, for example:

- Metagenomic species
- Species-level transcription ratios
- Metabolites
- Functional profiles, captured as Enzyme Commission (EC) gene families
- Serology
- Faecal calprotectin

The results once again reinforced, using PERMANOVA on concrete clinical data, the more abstract finding of review articles that in omic datasets, variation between individuals is larger than even variation arising out of strong biological effects, such as IBD phenotype or antibiotics. The strongest differences between individuals with and without IBD were the most pronounced in the metabolome. Specifically, IBD was associated with a lower diversity of metabolites.

A key actionable finding from time series analysis of the taxonomic data was that IBD patients exhibited significantly greater variation in composition over time than

healthy patients. Sometimes, variation in the microbiome of IBD patients was extreme, to the point where two samples taken at two different times *from the same person did not have any species in common.*

2.2.2. Microbiome-Metabolome Interactions in IBDs

Another paper concerned with the comparison of metabolomic and metagenomic profiles is Franzosa et al. [7]. The authors here took a more traditional approach and performed differential abundance (DA) analysis on metabolites as well as metagenomic species. By integrating data about differentially-abundant species and metabolites, 122 statistically-robust associations were discovered, leading to closer understanding of the mechanisms in the IBD gut.

The study also examined the metabolic state of subjects separately, and correlated the first PCoA axis of metabolic variation with faecal calprotectin levels, an indicator of the level of inflammation. The correlation was strong and highly statistically significant (Spearman's $r = 0.486$, $P < 10^{-6}$). This provides further evidence that IBD is associated with a higher variation in metabolites.

2.2.3. Longitudinal Changes in the Gut Microbiome of IBD Mice

In the paper "Development of Inflammatory Bowel Disease Is Linked to a Longitudinal Restructuring of the Gut Metagenome in Mice", Sharpton et al. [8] conducted a longitudinal gut metagenome study in mice, with the goal of understanding functional changes related to IBDs. For this reason, they focused on the abundances of functional groups, corresponding to KEGG modules.

Among others, a Kruskal-Wallis test was performed on Shannon entropy computed from KEGG modules at different time points. It found that functional alpha-diversity of IBD mice was more stable over time ($P = 0.47$) than that of healthy subjects ($P = 0.078$). Interestingly, this is the opposite of the pattern observed in metabolites, in which IBD was associated with higher, rather than lower, variation.

3. The Functional Genes Pipeline

3.1. Motivation

Taxonomic groups, such as species, are usually not in an unequivocal correspondence with biological functions. Many species are able to fulfill a certain function, while a given species may have several functions in itself. The association between genes and biological functions (or the lack thereof) is much stronger, although still not quite one-to-one. Therefore we explored the idea of using functional gene groups instead of taxa for quantifying the microbiome profile of persons, with the ultimate goal of classifying them as healthy or suffering from a certain group of diseases. Namely, we analyzed data from patients diagnosed with Irritable Bowel Syndrome and Inflammatory Bowel Diseases, which are diseases of the lower gastrointestinal tract, and which are thus hypothesized to correlate the strongest with the abundance profile of microbial genes.

Gene groups corresponding to various fundamental microbial functions have been selected by the help of a combination of the literature and biologists' expert opinion. Altogether, 121 orthologous families of genes were used for our analysis, categorized into 27 functional groups. The list of orthologous families and the corresponding function is reproduced in table 3.1. Gene families were retrieved from the KEGG database [9], and for this reason, they are identified by their KEGG ID.

Such a relatively short list of genes and functions is an advantage from a computational point of view as well. Taxon-based abundance analysis software typically comes with a large database of complete genomes of thousands or tens of thousands of microbial species and strains. It is not uncommon for such a database to take up several hundreds of gigabytes. Obtaining (e.g. downloading or building) and storing these databases thus incurs a significant overhead and is a constant source of hurdle, especially when the need arises for setting them up on a personal computer, usually for the purposes of small-scale research activity or developing and debugging the software itself.

KEGG ID	Abbrev./EC number	Function	KEGG ID	Abbrev./EC number	Function
K20708	E5.1.1.21	BCAA degradation	K22373	larA	l-lactate prod.
K03334	IL4I1	BCAA degradation	K22212	mleA, mleS	l-lactate prod.
K00271	vdh	BCAA degradation	K05357	VKORC1	menaquinone prod.
K00263	E1.4.1.9	BCAA synthesis	K00400	K00400	methanogenesis
K01652	ilvB, ilvG, ilvI	BCAA synthesis	K00399	mcrA	methanogenesis
K01653	ilvH, ilvN	BCAA synthesis	K00401	mcrB	methanogenesis
K00826	ilvE	BCAA synthesis	K03421	mcrC	methanogenesis
K01754	ilvA, tdcB	BCAA synthesis	K03422	mcrD	methanogenesis
K09011	cimA	BCAA synthesis	K00402	mcrG	methanogenesis
K00053	ilvC	BCAA synthesis	K15923	AXY8, FUC95A, afcA	mucin degradation
K01687	ilvD	BCAA synthesis	K01206	FUCA	mucin degradation
K01649	leuA, IMS	BCAA synthesis	K01205	NAGLU	mucin degradation
K00052	leuB, IMDH	BCAA synthesis	K12111	ebgA	mucin degradation
K01703	leuC, IPMI-L	BCAA synthesis	K12112	ebgC	mucin degradation
K01704	leuD, IPMI-S	BCAA synthesis	K17624	engCP, engBF, endoEF	mucin degradation
K03311	TC.LIVCS	BCAA transport	K01190	lacZ	mucin degradation
K01999	livK	BCAA transport	K07406	melA	mucin degradation
K02560	lpxM, msbB	LPS production	K01207	nagZ	mucin degradation
K00317	dmd-tmd	TMA degradation	K23550	nanH	mucin degradation
K14082	mtbA	TMA degradation	K01239	iunH	niacin-nicotinamide prod.
K14083	mttB	TMA degradation	K18153	nga	niacin-nicotinamide prod.
K14084	mttC	TMA degradation	K12410	npdA	niacin-nicotinamide prod.
K22443	cntA	TMA synthesis	K03783	punA, PNP	niacin-nicotinamide prod.
K22444	cntB	TMA synthesis	K03784	deoD	niacin-nicotinate prod.
K20038	cutC	TMA synthesis	K00763	pncB, NAPRT1	niacin-nicotinate prod.
K20037	cutD	TMA synthesis	K03462	NAMPT	niacin production
K01895	ACSS1_2, acs	acetate prod.	K08281	pncA	niacin production
K00128	ALDH	acetate prod.	K18427	hpdB	p-cresol sulfate prod.
K14085	ALDH7A1	acetate prod.	K18428	hpdC	p-cresol sulfate prod.
K00149	ALDH9A1	acetate prod.	K04069	pflA, pflC, pflE	p-cresol sulfate prod.
K00467	E1.13.12.4	acetate prod.	K09722	pps	pantothenate prod.
K01067	ACH1	acetate prod.	K01947	birA-coaX	pantothenate prod.
K18118	aarC, cat1	acetate prod.	K00867	coaA	pantothenate prod.
K01905	acdA	acetate prod.	K09680	coaW	pantothenate prod.
K24012	acdAB	acetate prod.	K03525	coaX	pantothenate prod.
K22224	acdB	acetate prod.	K20626	lcdA	propionate production
K00925	ackA	acetate prod.	K01604	mmdA	propionate production
K01512	acyP	acetate prod.	K13922	pduP	propionate production
K00138	aldB	acetate prod.	K05275	E1.1.1.65	pyridoxine production
K01026	pct	acetate prod.	K07758	PDXP	pyridoxine production
K00156	poxB	acetate prod.	K13248	PHOSPHO2	pyridoxine production
K01012	bioB	biotin prod.	K23998	PPOX	pyridoxine production
K01035	atoA	butyrate prod.	K00275	pdxH, PNPO	pyridoxine production
K01034	atoD	butyrate prod.	K18607	pno	pyridoxine production
K00929	buk	butyrate prod.	K03788	aphA	riboflavin production
K00798	MMAB, pduO	cobalamin prod.	K01093	appA	riboflavin production
K02303	cobA	cobalamin prod.	K09474	phoN	riboflavin production
K09882	cobS	cobalamin prod.	K00793	ribE, RIB5	riboflavin production
K04032	eutT	cobalamin prod.	K20861	ybjI	riboflavin production
K00287	DHFR, folA	folate production	K20862	yigB	riboflavin production
K13998	DHFR-TS	folate production	K14394	ACP1	thiamine production
K18589	dfrA1, dhfr	folate production	K01077	phoA, phoB	thiamine production
K19643	dfrA10, dfr10	folate production	K01078	PHO	thiamine production
K18590	dfrA12, dhfr	folate production	K06949	rsgA, engC	thiamine production
K19644	dfrA19, dfrA18	folate production	K01695	trpA	tryptophane prod.
K18591	dfrD, dhfr	folate production	K01696	trpB	tryptophane prod.
K01667	tnaA	indol production	K01427	URE	urease production
K19266	E1.2.1.22	l-lactate prod.	K01430	ureA	urease production
K00016	LDH, ldh	l-lactate prod.	K14048	ureAB	urease production
K07248	aldA	l-lactate prod.	K01429	ureB	urease production
			K01428	ureC	urease production

Table 3.1. Orthologous gene families and their functions

3.2. Architecture of the Software

Like many other pieces of metagenome analysis software, our functional genes pipeline applies a two-stage method. First, a database is built from the sequences of each gene family, including HMM profiles. Since a handful of heuristics are

applied for the creation of the HMM profiles (e.g. for multiple alignment and for determining conserved regions), a number of visualizations are also output by this stage of the program. This allows a biologist or bioinformatician to look at the results and determine whether they make sense, or, if they are wildly incorrect, discard them or rebuild the database with improved settings for the relevant heuristics. In the second stage, two different read alignment algorithms are applied: one is the usual short-read alignment to the raw reference sequences of individual genes, and the other is profile search against HMM profiles for entire orthologous families. In both cases, reads are counted with respect to several criteria of varying strictness, and the counts are then normalized and transformed according to industry best practices. In the following sections, we elaborate on the detailed modus operandi of both stages.

3.2.1. Building the Database

The specification for building the database, at a minimum, needs to be a list of KEGG Orthology IDs, one for each gene family. However, for technical reasons, this needs to be further refined. In particular, since KEGG Orthology IDs are not human-readable, we should allow a human-readable title (that contains e.g. the scientific name and the function of a family) to be specified as well. Furthermore, some families in the KEGG Orthology database comprise tens of thousands of genes – consequently, processing and downloading all of them is not practical (the KEGG REST API is notoriously slow to respond to HTTP requests). This means that the number of genes needs to be restricted in some way. For lack of a better method, we opted for randomly sampling a more manageable number of genes from the family. This number is also specified for each individual gene family.

Since the input to our program is apparently becoming structured, we choose to organize all the specifications into a single "manifest" file, written in the human-readable TOML format [10]. An example of such a manifest is given in listing 3.1.

This excerpt highlights two more important details about the code. First, besides the fact that each gene family contains sequence data for many individual genes, the software allows the user to BLAST for even more sequences, similar to those found by gene family ID. The reason behind this feature is that originally the KEGG database wasn't used for retrieving sequence data, and so only a single representative sequence was input, then homologous sequences were searched for in the RefSeq Representative Prokaryotic Genomes database (`ref_prok_rep_genomes` [11] [12]). However, since KEGG Orthology includes manually-curated sequences, using them

directly is preferred over trying to infer homology automatically via BLAST. Yet, the feature might be useful in the future and was therefore retained. If, however, running BLAST is not desired, and consequently we do not want to have to provide the corresponding BLAST database, we can turn the feature off completely by specifying that the number of required hits is zero.

Listing 3.1. Input manifest for database building

```
[[query]]
kegg-id = 'K00271'
title = 'BCAA degradation: K00271 (vdh)'
subsample = 100

[[query]]
kegg-id = 'K03334'
title = 'BCAA degradation: K03334 (IL4I1)'
subsample = 100

[blast]
num-hits = 0

[misc]
gene-cache-path = '/root/mp-userdata/.cache/'
```

The second feature that the `gene-cache-path` configuration option hints at is extensive caching. Since the act of downloading metadata and sequences through the KEGG HTTP API is pretty time-consuming, our pipeline caches every entry after downloading them for the first time. More precisely, the association between KEGG Orthology gene family IDs and the IDs of individual genes is cached, as well as all of the sequence data for the genes actually having been downloaded (i.e., caching occurs after subsampling). The cache expires after a configurable amount of time (by default, 30 days), so that it eventually synchronizes with any updates to the KEGG database.

The pipeline also reads from an internal configuration file, `config.toml` (please see the supplementary material), which contains default values for the configuration of the heuristic algorithms of this first stage as well as those of the second one. The default options can be overridden by specifying them in the input manifest file, structured in the same manner as they are laid out in the internal config file.

After downloading sequence data for each specified orthology group, it starts pro-

cessing them. In short, the following steps are performed on each set of sequences belonging to the same family:

1. The sequences are written out in a single FASTA file which will be used later as the reference database for read alignment.

2. A Multiple Sequence Alignment is created with Clustal Omega [13].

3. Conserved regions are determined using the following heuristics:

   a) At each position (column) i of the MSA, the entropy S_i is computed and normalized so that its maximal possible value is 1. This essentially means using the base-4 logarithm, since there are 4 possible symbols: A, C, T, G, i.e.:

   $$S_i = -\sum_{j \in \{A,C,T,G\}} P_{ij} \cdot \log_4(P_{ij}) \tag{3.1}$$

   Probabilities are computed using the total number of non-gap symbols at each position.

   b) The ratio of gaps G_i, i.e. the number of gaps divided by the number of aligned sequences, is computed for each position as well, this also gives a number between 0 and 1, inclusive.

   c) The raw conservation score is then computed for each position based on the scheme proposed by Valdar [14]:

   $$RC_i = (1 - S_i)^\alpha \cdot (1 - G_i)^\beta \tag{3.2}$$

   where α and β are configurable positive exponents, with default values of $\alpha = 0.5$ and $\beta = 1$.

   d) A "smoothed" conservation score C_i is then obtained by running the raw scores through a Gaussian filter, the width of which is also configurable and by default equals $\sigma = 4$.

   e) A low and a high score threshold is defined according to the following adaptive algorithm. The median of all scores $C_m = \underset{i}{\text{median}}(C_i)$ is computed. Next, four configurable values, called the low and high relative and absolute thresholds $0 \leq R_{lo} < 1 \leq R_{hi}$ and $0 \leq A_{lo} < A_{hi} \leq 1$, are used for deriving reasonable thresholds that work well with alignments of which the conservation score varies only very slightly, or is consistently very low or very high. We don't want to mark absolutely high-scoring regions (e.g. $C_i \geq 0.6$) as non-conserved or absolutely low-scoring regions (e.g. $C_i \leq 0.1$) as conserved just because their scores are

below or above a certain fraction of the median score, but we do want to adaptively take the general tendency of the scores into account. Thus, the low and high thresholds are defined respectively as:

$$T_{lo} = \begin{cases} A_{lo} & \text{if } C_m \cdot R_{hi} \leq A_{lo} \\ A_{hi} & \text{if } A_{hi} \leq C_m \cdot R_{lo} \\ \max(A_{lo}, C_m \cdot R_{lo}) & \text{otherwise} \end{cases} \tag{3.3}$$

and

$$T_{hi} = \begin{cases} A_{lo} & \text{if } C_m \cdot R_{hi} \leq A_{lo} \\ A_{hi} & \text{if } A_{hi} \leq C_m \cdot R_{lo} \\ \min(A_{hi}, C_m \cdot R_{hi}) & \text{otherwise} \end{cases} \tag{3.4}$$

f) Finally, hysteresis thresholding is applied to the smoothed score, similarly to the corresponding step of the Canny operator [15]. All positions of the MSA with a score exceeding the high threshold are marked as conserved, all positions not exceeding the lower threshold are marked as not conserved, and the remaining positions are marked as conserved if and only if they are connected to an unconditionally conserved region (i.e. one with scores above the high threshold).

4. The (ambiguous) consensus sequence of the alignment is obtained: at each position, the frequencies of all 4 kinds of bases are counted, and bases of which the frequency exceeds a certain (configurable) proportion of the count of the most frequent base at that position are included in the consensus sequence. This proportion is 0.55 by default. If more than one base is included in the consensus sequence at a given position, this fact is indicated with an IUPAC ambiguity code.

5. The MSA is written out in the Stockholm format, annotated with its conserved regions. Sub-alignments and consensus sequences for each conserved region are also written out to separate files for easier downstream processing.

6. The `hmmbuild` program from the HMMER3 suite [16] is then used for building HMM profiles.

7. The MSA is visualized in the form of a colorful HTML table. Conservation scores and conserved regions are plotted for easier inspection, and for each conserved region, a so-called "sequence logo" is created with the help of the Logomaker library [17]. This depicts the most prominent bases at each position in an intuitive manner.

Since the quality of a heuristically-computed multiple alignment may be negatively impacted by wildly varying sequence lengths, sequences are clustered by length, and the above procedure is repeated for each such cluster. Clustering is based on an unsupervised model. First, the probability density of the logarithm of the sequence lengths is obtained using Gaussian Kernel Density Estimation, as implemented in statsmodels [18], and with a bandwidth of 50% of the approximation given by Silverman's rule of thumb. (Using the rule-of-thumb value directly typically leads to oversmoothing, and as a result, important gaps between clusters remain undiscovered. However, this has only been verified by visual inspection of the results, and is therefore a subjective statement.) Cluster boundaries are then defined at the local minima of the estimated PDF.

Clustering turned out to be more useful (in the sense that it noticeably improved alignment quality in some cases) when homology groups had previously been defined as a single seed sequence and the corresponding BLAST hits. After our pipeline was switched over to KEGG, this clustering basically ceased to be necessary, because the quality of MSAs based on entire KEGG Orthology groups was readily deemed satisfactory.

3.2.2. Quality and Efficiency of the Database

In this section, we describe how we curated the automatically-generated alignments, profiles, and conserved regions, and we compare the storage requirements of our software to those of a state-of-the-art taxon-based metagenome analysis tool, SHOGUN [19].

The primary indicator of MSA quality is the alignment plot itself. An alignment is good-quality if homologous genes align reasonably well and as such, their conserved regions can be easily distinguished. The alignment is low-quality if for some reason the similar regions of each sequence do not align, therefore evolutionary relations cannot be deduced from it. Figure 3.1 shows part of the alignment plot for the `K18591` orthology group, a very conserved family of genes, while figure 3.2 depicts a non-conserved section of family K07406 – a low-quality alignment would look like this along the entire sequence.

The second kind of plot useful for quality checking is the (smoothed) conservation score and the thresholding, which indicate the boundaries of conserved regions. The heuristic is tuned correctly when there are a few long conserved regions and the corresponding MSA plots look uniform and well-aligned, as in figure 3.3; whereas an

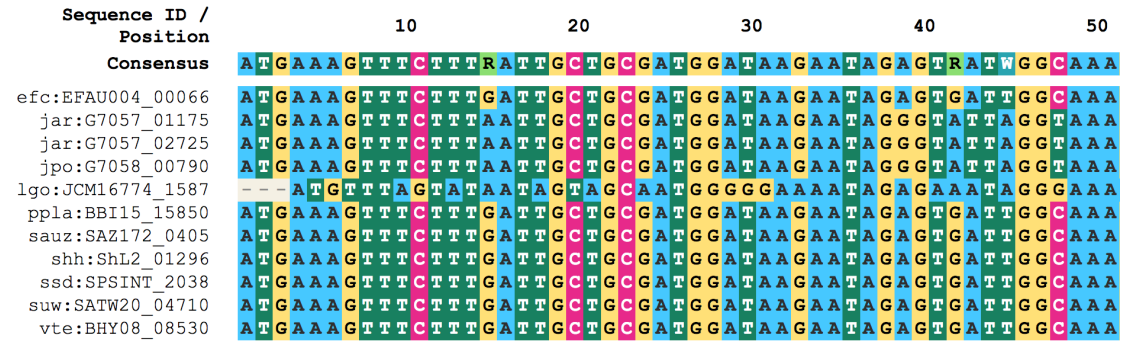

Figure 3.1. A good-quality alignment for orthology family K18591.

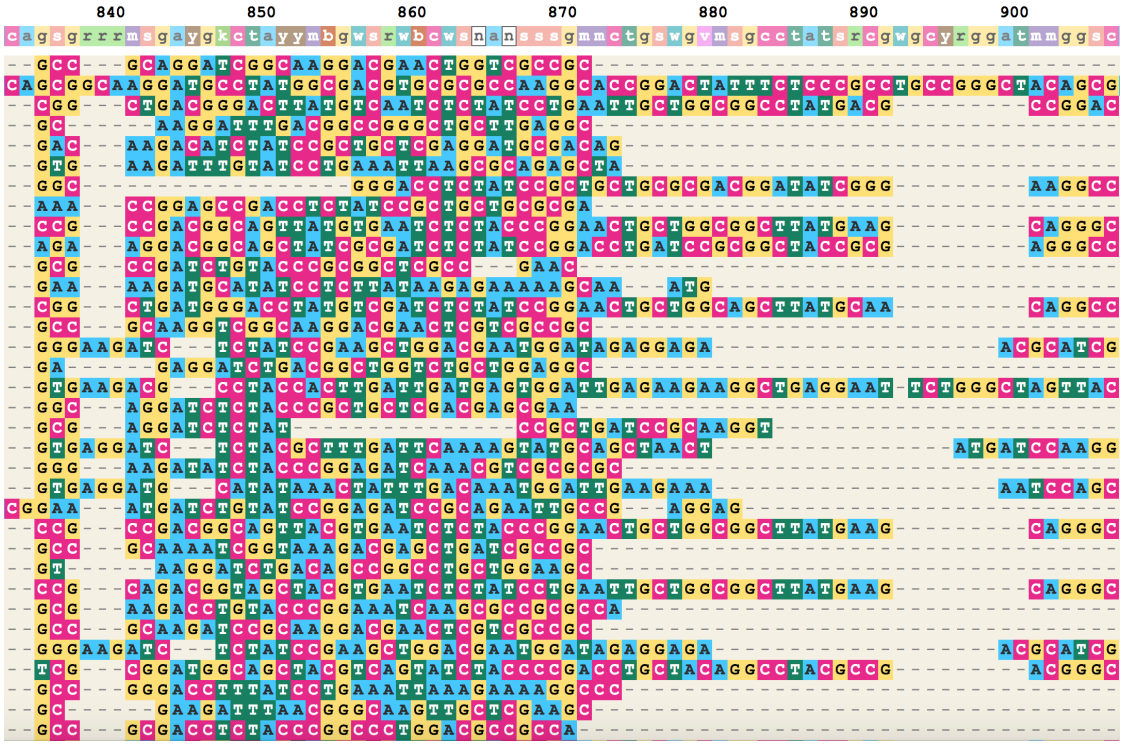

Figure 3.2. A "bad-quality" excerpt from the alignment for family K07406. A non-conserved region is in fact shown here for illustrative purposes only.

incorrectly-working heuristic results in many short, almost consecutive conserved regions, making the generated HMM profiles fragmented and hard to align against, as shown in figure 3.4.

The last kind of quality control plot is aimed at assessing the level of conservation within a conserved region. These so-called sequence logos show the letters occurring at each position of a conserved region of an MSA, their height being proportional to their ratio in that position. In addition, the total height of a stack of letters quantifies the amount of conservation in that column: the higher the stack, the higher the conservation (i.e. the lower the entropy). A well-conserved region can be seen in figure 3.5, while a not very well-conserved one is shown in figure 3.6.

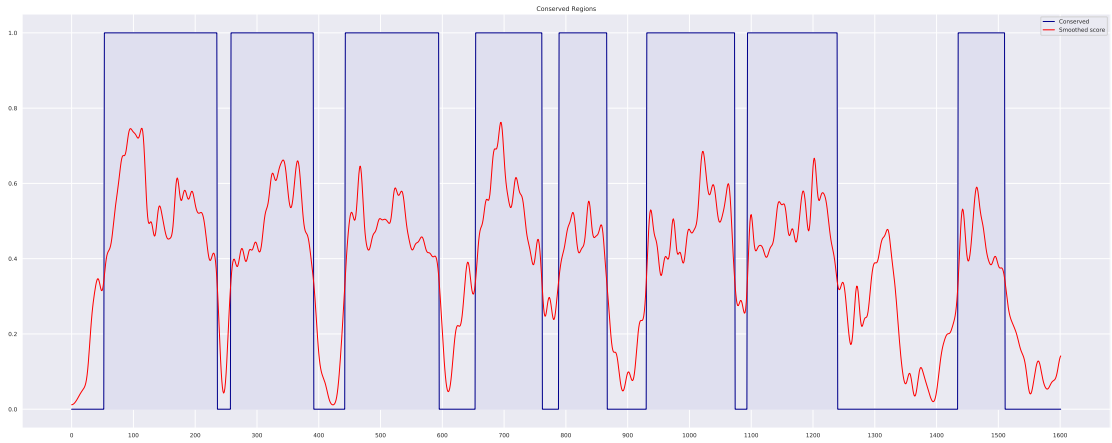

Figure 3.3. Good-quality conserved regions from family K03311.

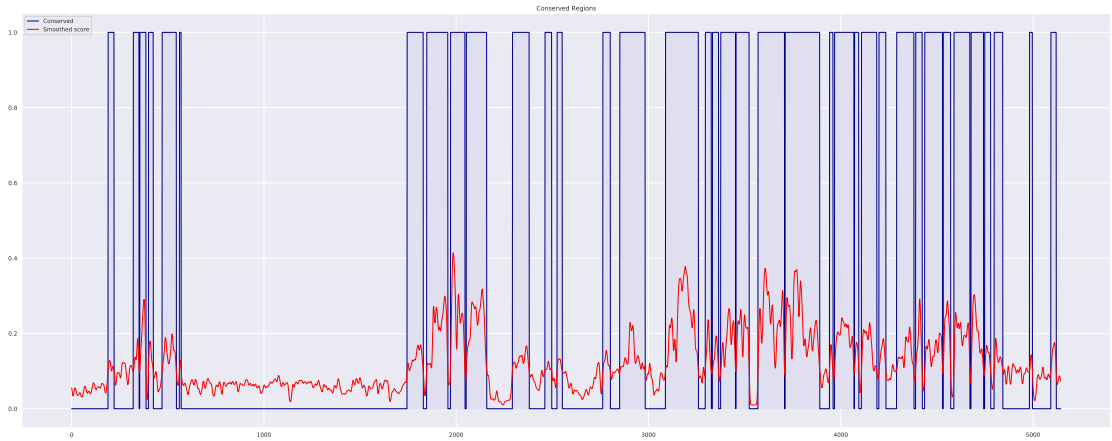

Figure 3.4. Low-quality conserved regions from family K01206.

Figure 3.5. The beginning of a well-conserved region from family K18591.

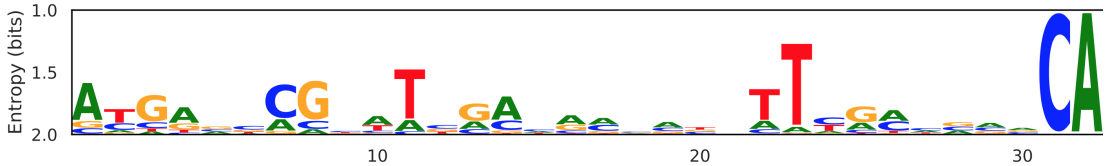

Figure 3.6. Excerpt from a not well-conserved region in family K01695.

After inspecting the 121 generated entries, we concluded that most of them are good-quality and thus we proceeded to use them as-is. The database takes up a little more than 2 Gigabytes in total, including all the quality control plots that our

software output alongside the useful data (sequences and HMM profiles). Table 3.2 summarizes the size distribution of these entries, while table 3.3 details the size of the components of the database used by the SHOGUN toolkit. This shows that using fewer and shorter sequences does obviously pay off in terms of disk space: **our software requires less than $\frac{1}{35}$ of the size of the SHOGUN database.**

Metric	**Size**	
Minimum	224.00	kB
Median	18.00	MB
Mean	19.86	MB
Maximum	82.00	MB
Total	**2.35**	**GB**

Table 3.2. Sizes of database entries generated for 121 functional gene families.

File	**Size**	
humanD252.acx	17.00	GB
humanD252.edx	3.40	GB
ko-enzyme-annotations.txt	975.00	kB
ko-module-annotations.txt	576.00	kB
ko-pathway-annotations.txt	2.30	MB
ko-species2ko.80pct.txt	12.00	MB
ko-species2ko.txt	12.00	MB
ko-strain2ko.txt	38.00	MB
rep82.1.bt2l	6.40	GB
rep82.2.bt2l	9.50	GB
rep82.3.bt2l	6.60	MB
rep82.4.bt2l	4.80	GB
rep82.fna	19.00	GB
rep82.gg.ctr	8.30	GB
rep82.gg.log	2.00	MB
rep82.rev.1.bt2l	6.40	GB
rep82.rev.2.bt2l	9.50	GB
rep82.tax	1.90	MB
sheared_bayes.txt	2.00	MB
Total	**84.00**	**GB**

Table 3.3. Detailed space requirements of SHOGUN.

3.2.3. Quantifying Relative Abundances

Once the database is built, the second stage of the software, abundance counting, can be invoked. Here two distinct methods are applied.

First, "traditional" read alignment is performed using minimap2 [20]. This tool is an evolution of the widely-used BWA-MEM [21] aligner from the same author, and while it is primarily designed for aligning long and ultra-long reads, it is claimed to be approximately 3 times faster than BWA-MEM even on short reads, while achieving comparable accuracy. In our experience, this claim seems to be accurate for the most part.

The output of minimap2 is a SAM file containing information such as which aligned read mapped to which reference sequence, the substitutions and indels in the alignment, and whether the pair of a paired-end read was mapped as well. This SAM file is then parsed by our software, and some useful statistics (such as percent identity of the alignment) are extracted from it. These will be used later for deciding whether a mapped read should indeed be counted towards the abundance of the corresponding gene.

The second method is profile hidden Markov model search, again using the HMMER3 package. Specifically, the `nhmmer` program is invoked for comparing all of the 121 profiles, each generated from multiple alignments of the corresponding homologous family, to the sequenced reads. While HMMER wasn't specifically designed for read alignment, it is a more sophisticated model than simple one-vs-one alignment. Furthermore, there is precedent for relying on HMM profiles in the context of metagenome analysis. For instance, Skewes-Cox et al [22] use them for detecting viruses in metagenomic data, and the SAT-Assembler [23] also performs a profile HMM search in order to create overlap graphs between reads mapped to the same profile.

The only downside of profile HMM searching is that it is orders of magnitude slower than direct sequence alignment. This makes it necessary to reduce the number of reads aligned to the reference profiles. Similarly to how we dealt with an excessively high number of sequences in the database building step, we implemented random sampling for the reads to be aligned to HMM profiles as well, mainly for reasons of simplicity.

The output of `nhmmer` is in a whitespace-separated, tabular, and to some extent ad-hoc format, for which there is no parser in the Biopython [24] package (that we otherwise rely heavily on for parsing various file formats used extensively in bioinformatics). In addition, since HMMER3 isn't a read aligner, it doesn't take paired-end reads into consideration. Therefore, further post-processing is required.

Pairing of the reads is achieved with the following algorithm:

1. First, we note that `nhmmer` handles individual HMM profiles independently from one another. This means that a single read can be mapped to more than one profile.

2. Next, for each fragment (i.e., pair of reads), we take the Cartesian product of the mappings of both reads. This effectively produces all possible combinations of HMM profiles that matched the reads.

3. We make the (somewhat naive) assumption of independence of composition at the two ends of the fragment, which means that the P-value of a pair is the product of the P-values of its two reads. We then note that for small E-values, the E-value is approximately proportional to the P-value, which in turn means that the pair P-value is approximately proportional to the product of its E-values. Therefore, we choose the mapping with the smallest product (or, equivalently, the smallest geometric mean) of E-values. (`nhmmer` only outputs E-values.)

4. Finally, we check if the given mapping results in a "proper pair", and we augment the mapping with this piece of information. A proper pair is one such that:

   a) both of its reads are mapped, and

   b) they are mapped to the same profile, and

   c) they are mapped to different strands.

   The latter two are correct criteria, since `nhmmer` checks the reverse complement of each read as well, and it also outputs the direction of the match.

Two examples for the post-processed read mappings are provided in figures 3.7 (for `minimap2`) and 3.8 (for `nhmmer`). Only the most important columns are shown for brevity and readability.
Once appropriately-postprocessed read mappings are obtained, they can be converted into relative abundances. This is done in two steps.

First, the reads are filtered according to certain criteria. For both mappings derived by minimap2 and HMMER, we implemented 6 levels of decreasing strictness, called **Draconian, Strict, Moderate, Lenient, Loose,** and **Placebo,** respectively. Without diving into the specifics of these criteria, their importance is that they impose different requirements on each read in order to be counted towards

read	reference	match	sub	in	del	align_ident	quality	is_proper_pair
SRR5935741.12712	bvu:BVU_3836	101	0	0	0	1.0	60	0
SRR5935741.3492	bfb:VU15_14680	88	8	1	1	0.8979591836734694	0	0
SRR5935741.299	bcae:A4V03_12425	63	3	0	0	0.9545454545454546	6	0
SRR5935741.18769	dys:G7050_11625	55	3	0	0	0.9482758620689656	23	0
SRR5935741.8004604	bdo:EL88_24125	71	3	0	0	0.9594594594594594	50	1
SRR5935741.8006327	bdo:EL88_24125	60	2	0	0	0.967741935483871	17	1
SRR5935741.2041	bdo:EL88_24125	61	2	0	0	0.9682539682539684	6	0
SRR5935741.8005150	bcac:CGC64_02250	55	0	0	0	1.0	44	1
SRR5935741.8005150	bcac:CGC64_02250	55	0	0	0	1.0	37	1
SRR5935741.8006094	bdh:GV66_03955	93	8	0	0	0.9207920792079208	21	0

Figure 3.7. Excerpt from the post-processed minimap2 mapping data for patient SRR5935741 from the study of Lloyd-Price et al [6].

read	read_index	reference	strand	bitscore	bias	bias_ratio	evalue	pair_mean_evalue	is_proper_pair
SRR5935741.4656891	1	tryptophane_K01695_trpA|allseqs|full	+	7.0	2.9	0.2929292929292929	8.4	8.4	1
SRR5935741.4656891	2	tryptophane_K01695_trpA|allseqs|full	-	7.0	2.9	0.2929292929292929	8.4	8.4	1
SRR5935741.4060729	1	tryptophane_K01695_trpA|allseqs|full	-	6.9	8.7	0.5576923076923077	8.8	8.8	0
SRR5935741.4546806	2	tryptophane_K01695_trpA|allseqs|full	-	6.9	2.3	0.25	8.9	8.9	0
SRR5935741.7824682	2	tryptophane_K01695_trpA|allseqs|full	-	6.8	4.1	0.3761467889908257	9.3	9.3	0
SRR5935741.2038520	1	tryptophane_K01695_trpA|allseqs|full	-	6.8	3.1	0.31313131313131315	9.7	9.699999999999998	0
SRR5935741.1226331	1	tryptophane_K01696_trpB|allseqs|full	+	84.1	6.2	0.06866002214839424	1.6000000000000002e-23	2.529822128134704e-22	1
SRR5935741.1226331	2	tryptophane_K01696_trpB|allseqs|full	-	76.1	7.0	0.08423586040914563	4e-21	2.529822128134704e-22	1
SRR5935741.13954842	1	tryptophane_K01696_trpB|allseqs|full	-	80.7	8.6	0.09630459126539752	1.7e-22	4.324349662087933e-22	1
SRR5935741.13954842	2	tryptophane_K01696_trpB|allseqs|full	+	78.0	10.3	0.11664779161947908	1.1e-21	4.324349662087933e-22	1
SRR5935741.7082493	1	tryptophane_K01696_trpB|allseqs|full	-	75.6	3.1	0.03939008894536214	5.7e-21	5.700000000000019e-21	0

Figure 3.8. Excerpt from the post-processed HMM mapping data for patient SRR5935741 from the study of Lloyd-Price et al [6].

the abundance of the matching reference. For instance, a filter might require any or all of the following:

1. In paired-end mode, the read must be mapped together with its pair.
2. At least a certain proportion (e.g. 90%) of the read must be aligned to the reference.
3. At least a certain proportion of the aligned positions must be matches (as opposed to substitutions or indels).
4. The read must pass the internal probabilistic quality check of the read aligner.
5. The read must have an E-value below, and/or a bit score above, a specific threshold.
6. In the case of HMM profile searches, the so-called "bias correction" of the bit score for the read must not exceed a certain ratio of the raw (uncorrected) bit score. This essentially ensures that the composition of the read is sufficiently random and meets the assumptions of the underlying hidden Markov model

– otherwise, reported E-values might be too optimistically low and therefore unsuitable for assessing the significance of an alignment.

Once reads are counted for each reference (sequence or HMM profile), the counts are transformed into relative abundances. According to current best practice, we calculate the metric known as TPM (which essentially measures relative abundance in units of parts per million) and percents (which is just $\frac{1}{10000}$ of the TPM values). Briefly, this means normalizing for reference length first, yielding the so-called "RPK" or "Reads per Kilobase" metric, then dividing each RPK value by their sum and multiplying by 1 000 000 (for TPM) or 100 (for percents).

For HMM-based mappings, the process ends here, since each gene family corresponds to exactly one reference profile. In the case of alignments produced by `minimap2`, however, each gene family is associated with many sequences. Consequently, TPM values for sequences in the same homologous family need to be summed up. An example of the results of this computation is provided in figure 3.9.

gene	draconian	strict	moderate	lenient	loose	placebo
degradation: E5.1.1.21 (isoleucine 2-epimerase) [K20708]	0,000000	0,000000	0,000000	0,000000	0,000000	0,000000
BCAA degradation: IL4I1 (L-amino-acid oxidase) [K03334]	0,000000	0,000000	0,000000	0,000000	0,000000	0,000000
degradation: vdh (valine dehydrogenase (NAD+)) [K00271]	0,000000	0,000000	0,000000	0,000000	0,000000	0,000000
CAA synthesis: E1.4.1.9 (leucine dehydrogenase) [K00263]	0,000000	0,000000	0,000000	0,000000	0,000000	0,000000
, ilvI (acetolactate synthase I/II/III large subunit) [K01652]	0,000000	0,000000	0,000000	0,000000	0,000000	0,013417
H, ilvN (acetolactate synthase I/III small subunit) [K01653]	2,428941	1,833459	1,685240	1,768552	1,797771	1,768541
E (branched-chain amino acid aminotransferase) [K00826]	0,000000	0,000000	0,000000	0,000000	0,004499	0,008707
sis: E4.3.1.19, ilvA, tdcB (threonine dehydratase) [K01754]	0,000000	0,000000	0,009539	0,008388	0,008114	0,007702
BCAA synthesis: cimA ((R)-citramalate synthase) [K09011]	21,623923	16,359067	15,095247	14,171108	14,000343	13,868595
AA synthesis: ilvC (ketol-acid reductoisomerase) [K00053]	0,683479	0,593312	0,863296	1,174433	1,261185	1,736983
CAA synthesis: ilvD (dihydroxy-acid dehydratase) [K01687]	0,015978	0,028957	0,035615	0,048341	0,088464	0,344852
nthesis: leuA, IMS (2-isopropylmalate synthase) [K01649]	0,000000	0,000000	0,011909	0,013617	0,152975	0,425056
leuB, IMDH (3-isopropylmalate dehydrogenase) [K00052]	0,000000	0,000000	0,000000	0,000000	0,000000	0,000000
/(R)-2-methylmalate dehydratase large subunit) [K01703]	0,132868	0,089990	0,094421	0,134890	0,122819	0,204730
/(R)-2-methylmalate dehydratase small subunit) [K01704]	0,066652	0,067714	0,060574	0,053263	0,068357	0,079686
hain amino acid:cation transporter, LIVCS family) [K03311]	0,000000	0,000000	0,000000	0,000000	0,000000	0,000000
acid transport system substrate-binding protein) [K01999]	0,000000	0,000000	0,000000	0,000000	0,000000	0,000000
sbB (lauroyl-Kdo2-lipid IVA myristoyltransferase) [K02560]	0,020263	0,013724	0,012277	0,010795	0,010443	0,009912
dimethylamine/trimethylamine dehydrogenase) [K00317]	0,000000	0,000000	0,000000	0,000000	0,000000	0,000000
rrinoid protein]:coenzyme M methyltransferase) [K14082]	0,000000	0,000000	0,000000	0,000000	0,000000	0,000000
amine---corrinoid protein Co-methyltransferase) [K14083]	0,000000	0,000000	0,000000	0,000000	0,000000	0,000000
adation: mttC (trimethylamine corrinoid protein) [K14084]	0,000000	0,000000	0,000000	0,000000	0,000000	0,000000
nthesis: cntA (carnitine monooxygenase subunit) [K22443]	0,000000	0,000000	0,000000	0,000000	0,000000	0,000000
nthesis: cntB (carnitine monooxygenase subunit) [K22444]	0,000000	0,000000	0,000000	0,000000	0,000000	0,000000
A synthesis: cutC (choline trimethylamine-lyase) [K20038]	0,224601	0,175523	0,157014	0,142665	0,142460	0,135226
choline trimethylamine-lyase activating enzyme) [K20037]	0,000000	0,000000	0,000000	0,005534	0,000000	0,005082
roduction: ACSS1_2, acs (acetyl-CoA synthetase) [K01895]	0,000000	0,000000	0,000000	0,000000	0,000000	0,000000

Figure 3.9. Excerpt from the relative abundance table based on `minimap2` for patient SRR5935741 from the study of Lloyd-Price et al [6].

3.2.4. Computational Performance

The abundance analysis stage of our software has a clear computing performance advantage over SHOGUN both in terms of RAM usage and running time when considering only direct read mapping (and not HMM profile alignment). Using this metric is justified because — surprisingly — HMM profile searching turned out not to be very useful for classification. The following numbers have been obtained on a Scaleway GP1-L virtual machine, with 32 AMD EPYC CPU cores and 128 GB of RAM.

The reduction in operative memory footprint is genuine and is owed to our smaller database, since read aligners typically need to load the entire reference database into memory. In our case, minimap2 needed less than 0.8 GB of RAM, while the aligner we ran SHOGUN with, Bowtie2 [25], used 27.7 GB. **This signifies a more than 34-fold reduction in RAM usage.** The aligner that SHOGUN comes with by default, called BURST [26], is significantly faster than Bowtie2, but it consumes even more RAM, and we were in fact unable to get it running on the aforementioned machine, as it simply exhausted all available RAM then crashed.

The reduction in running time, however, is mostly due to the choice of read aligner, and is therefore not an advantage of our pipeline per se. Still, the moral of this comparison is that it is very well worth following new developments in high-performance computing, and authors should continuously monitor and update dependencies in order to find the best current alternative.

Table 3.5 illustrates the execution time of both programs, on a small subset of patient data from the study by Lloyd-Price ey al [6], while figure 3.10 visualizes the same data. Table 3.4 shows the results of testing the hypothesis that the running time of our own pipeline using minimap2 is significantly smaller than that of SHOGUN with Bowtie2, and gives the ratio between the mean running times.

Quantity	**Value**
Own runtime mean *(s)*	87.8
Own runtime stddev *(s)*	0.6476
SHOGUN runtime mean *(s)*	929.3
SHOGUN runtime stddev *(s)*	401.8
One-sided Wilcoxon test statistic	190.0
One-sided Wilcoxon test p-value	$\mathbf{7.15 \cdot 10^{-5}}$
Speedup (fold)	**10.6**

Table 3.4. Summary of the comparison between running times.

Patient ID	Runtime, own pipeline *(s)*	Runtime, SHOGUN *(s)*
SRR5935776	87.36	860.95
SRR5935783	87.20	528.00
SRR5935852	87.52	1577.05
SRR5935978	87.28	1775.76
SRR5936116	87.13	493.46
SRR5936139	88.72	578.24
SRR5936180	87.52	720.11
SRR5946618	87.66	767.10
SRR5946647	87.84	631.35
SRR5947014	88.13	903.32
SRR5947067	88.93	474.33
SRR5950519	87.08	1403.51
SRR5950634	88.16	1387.42
SRR5950645	87.77	1452.19
SRR5950695	87.99	562.38
SRR5950697	87.99	943.93
SRR5950714	87.69	1045.43
SRR5950720	87.59	673.22
SRR5950746	89.57	878.67

Table 3.5. Running time of our software and that of SHOGUN.

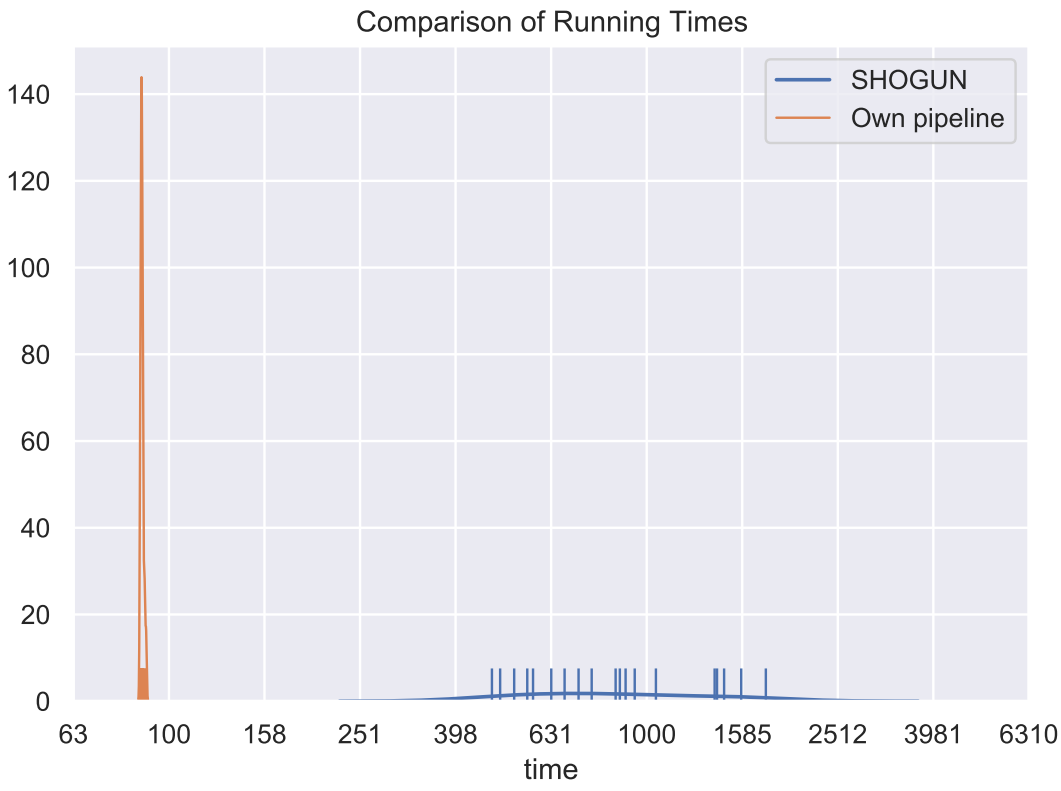

Figure 3.10. Visualization of the different running times.

3.3. Packaging for Reproducibility

The software was developed and tested under macOS 10.12.6 “Sierra” and Debian GNU/Linux 10 “Buster”. Many nontrivial programming tasks (such as interfacing with the filesystem, dealing with character encodings, calling out to the shell, or compiling external tools from source) are significantly easier to accomplish if a Unix-like environment can be assumed. However, since future users of the software are expected to work with a variety of other operating systems, including Microsoft Windows, the impedance mismatch between user and programmer convenience had to be bridged somehow. Furthermore, the pipeline relies on several dependencies: Python libraries as well as stand-alone bioinformatic programs. These should preferably be pinned at exact versions for improved reliability.

For these reasons, the software and all of its dependencies have been packaged as a Docker [27] image based on Ubuntu GNU/Linux 20.04 “Focal Fossa”, using Docker 2.2.0.5 (43884). Finally, the image was exported as a gzipped tar archive, from which it has been successfully installed on two other computers to date. Containers instantiated from this image can then communicate with the host machine via Docker volumes and a dedicated mount point inside the container. This is necessary for uploading input (e.g. metagenomic read files) and downloading results (e.g. tables containing abundance profiles) from/to the host computer.

4. Predicting Disease Status with Functional Genes

4.1. Data Sources

Raw metagenomic sequencing data have been obtained from four sources:

1. A set of 15 healthy clients of Medipredict, Ltd. (proprietary data). For reasons of data protection, directly identifiable information such as age, sex, and any data relating to the microbial composition, has been omitted from tables, plots, and the Supplementary material for these persons.
2. A study about Inflammatory Bowel Diseases by Lloyd-Price et al [6].
3. A de novo genome assembly article by Nielsen et al [28], henceforth referred to as "PRJEB1220".
4. The American Gut Project [29], referred to as "PRJEB11419" in this Thesis.

The last two studies have been selected by searching GMrepo [30] and filtering for samples from large, manually-curated metagenomic analysis projects, with the following criteria: `QCStatus = 1 AND experiment_type = 'Metagenomics' AND nr_reads_sequenced >= 1000000`. We then performed basic, automated exploratory analysis on the metadata for each study, plotting the distribution of patients' age, BMI, country of origin, disease status, and that of the sequencing depth of each stool sample. We preferred data sets where age and sex were specified, the countries of origin were diverse, detected diseases were only or mostly IBS and IBD, and the number of healthy controls did not exceed the number of ill patients disproportionately.

Naturally, no data set met all of these requirements simultaneously, therefore we picked the ones that still seemed the most decent based on at least the existence of metadata, geographical diversity, or healthy-ill balance. Severely unbalanced data sets (e.g. those with thousands of healthy controls) were again handled by (pseudo-) randomly choosing as many healthy samples as there were ill ones. This was also

necessary because processing the raw sequencing data and obtaining abundance information is time-consuming, so with our computing capabilities, it wouldn't have been possible to complete several thousands of samples within a time frame reasonable for this Thesis.

Furthermore, some of the selected samples ultimately failed to download or became corrupted for mostly unclear technical reasons that caused frequent intermittent lack of web service availability. These very few samples have been ignored and were subsequently excluded from further analysis.

Finally, both our pipeline and the SHOGUN have been run on each patient's data, resulting in 581 data points for each kind of result (functional gene composition in the case of our pipeline; and taxon, KEGG module, and pathway composition in the case of SHOGUN). All-zero rows have been omitted, because these are impossible to handle with the imputation method we used later. The smallest dataset thus cleaned was 536 rows long, which means that overall, not much data was lost. In addition, patient metadata was joined to each dataset redundantly, with the sole purpose of facilitating the rest of the work.

4.2. Handling Compositional Data

It is a well-known fact that relative abundances are a form of compositional data. Namely, they always sum to a constant, which can be normalized to 1 without loss of generality. This missing degree of freedom causes problems if we naively attempt to compute statistics (e.g. correlation coefficients) directly from the relative abundances. For example, the absolute abundance of a single species might increase while the rest is unchanged. This causes all the unchanged species to decrease in relative abundance, which in turn leads to spurious correlations.

Consequently, the data have to be transformed before statistical analysis. The most popular and simplest approaches are based on logarithms of relative abundances or ratios thereof. This means that zero values have to be imputed with small nonzero values. After researching available options, we decided to use multiplicative imputation [31], because it is easy to implement and has appealing statistical properties – namely, it is one of the methods that distorts data the least. The small imputed value was chosen to be $\delta = \frac{1}{D^2}$ where D is the dimensionality of the space (i.e. the number of genes or taxa), following the default in scikit-bio [32].

Therefore, values are transformed during imputation like so:

$$X_{ij}^{(i)} = \begin{cases} \frac{1}{D^2} & \text{if } X_{ij}^{(r)} = 0 \\ \left(1 - \frac{1}{D^2} \cdot \sum_{j=1}^{D} \mathbb{I}(X_{ij}^{(r)} = 0)\right) \cdot X_{ij}^{(r)} & \text{otherwise} \end{cases} \tag{4.1}$$

where $X^{(i)}$ is the imputed matrix, $X^{(r)}$ is the original (raw) but normalized matrix of relative abundances, that is, $\forall i \sum_{j=1}^{D} X_{ij}^{(r)} = 1$, and $\mathbb{I}(\cdot)$ is the indicator function. As usual, rows of the X matrices represent patients while their columns are genes or taxa.

Following imputation, three well-known transforms have been applied to the data: centered log-ratio (CLR), isometric log-ratio (ILR), and relative log-expression (RLE). Since the literature doesn't reach consensus as to what exactly these terms should mean, below we reproduce the exact formulae that we used:

$$\mathrm{CLR}(X)_{ij} = \log(X_{ij}) - \mathrm{IQM}(\log(X_{i\cdot})) \tag{4.2}$$

$$\mathrm{ILR}(X) = -\mathrm{CLR}(X) \cdot H_D^T \tag{4.3}$$

$$\mathrm{RLE}(X)_{ij} = \log(X_{ij}) - \mathrm{IQM}(\log(X_{\cdot j})) \tag{4.4}$$

where $\mathrm{IQM}(\cdot)$ denotes the interquartile mean of its argument vector, and H_D is the incomplete Helmert matrix of order D, $H_D \in \mathbb{R}^{(D-1)\times D}$.

Each of these methods has advantages and drawbacks. CLR and RLE are conceptually simple and easy to interpret, since transformed values are computed by comparing them to the central tendency along the axis of taxa (in the case of CLR) or patients (in the case of RLE). However, while improving statistical properties, they do not completely get rid of the constant-sum constraint: the sum along the appropriate dimension will still be close to zero, since measures of central tendency, the IQM included, are expected to be "close" to the mean.

In contrast, IRL does in fact get rid of this constraint by normalizing away the spurious dimension, therefore outputing only as many dimensions as there are actual degrees of freedom. However, the results will not be easy to interpret, nor biologically meaningful, since transformed values are a somewhat complicated linear combination of input values.

4.3. Exploratory Data Analysis

After transforming the data, we can start analyzing it. The first step in this is of course exploratory analysis. The common theme of exploring microbiome data is **dimensionality reduction**: the number of dimensions is in the hundreds or thousands, which cannot be visualized directly. For the same reason, plotting the distribution of all input dimensions is not useful: even if we did it, we could not look at hundreds or thousands of plots. We know that there is an inherent problem with the data anyway: relative abundances are usually heavily zero-inflated, so a sizeable portion of the data will come from imputation, resulting in a disproportionate frequency of a single constant value. Therefore, we know that statistical models relying on the assumption of normality cannot be used.

Instead, we are interested in simplifying the structure of the data in order to be able to plot it. We take two approaches:

1. Computing low-dimensional summary statistics and correlating them, and
2. Using unsupervised models for visualizing the distance between points, then inferring how they cluster.

As for the first approach, a convenient and biologically relevant metric is that of alpha diversity. This may be computed from relative abundances P_i in several different ways, although the results tend to be qualitatively similar, except in extreme cases. We computed three indexes of diversity:

1. Shannon diversity (entropy):

$$D_{Shannon} = -\sum_i P_i \cdot \log(P_i) \tag{4.5}$$

2. Simpson diversity, inverse (effective number of taxa/genes):

$$D_{Simpson} = \frac{1}{\sum_i P_i^2} \tag{4.6}$$

3. Richness (Simple count of detected taxa/genes):

$$D_{Richness} = \sum_i \mathbb{I}(P_i > 0) \tag{4.7}$$

We then plotted these together and examined whether any one of them clearly separates healthy and ill patients. We observed that the correlation between Shannon

and Simpson diversities is almost perfect (although we do need to take the logarithm of the Simpson and richness indexes, since entropy measures bits). The lowest Pearson correlation between them was 0.889 ($P = 1.82 \cdot 10^{-198}$ in the `shogun_taxatable.strain.kegg` dataset), and in 14 out of the 19 data sets, it exceeded 0.95 with similar, very low P-values. Correlation between Shannon entropy and richness wasn't so clear-cut, and neither was correlation between Simpson diversity and richness. Often, these were not significant at all, one of them was negative (which is clearly nonsensical), and only in 4 cases did they exceed 0.9. The highest-correlated case is shown in figure 4.1.

Neither diversity metric can be used for confidently distinguishing between ill and healthy samples, as data points do not cluster based on health status. They do however cluster based on their study of origin, suggesting that **the experimental setup impacts results so strongly that it suppresses the biological signal, if any.** Figures 4.2 and 4.3 demonstrate this phenomenon.

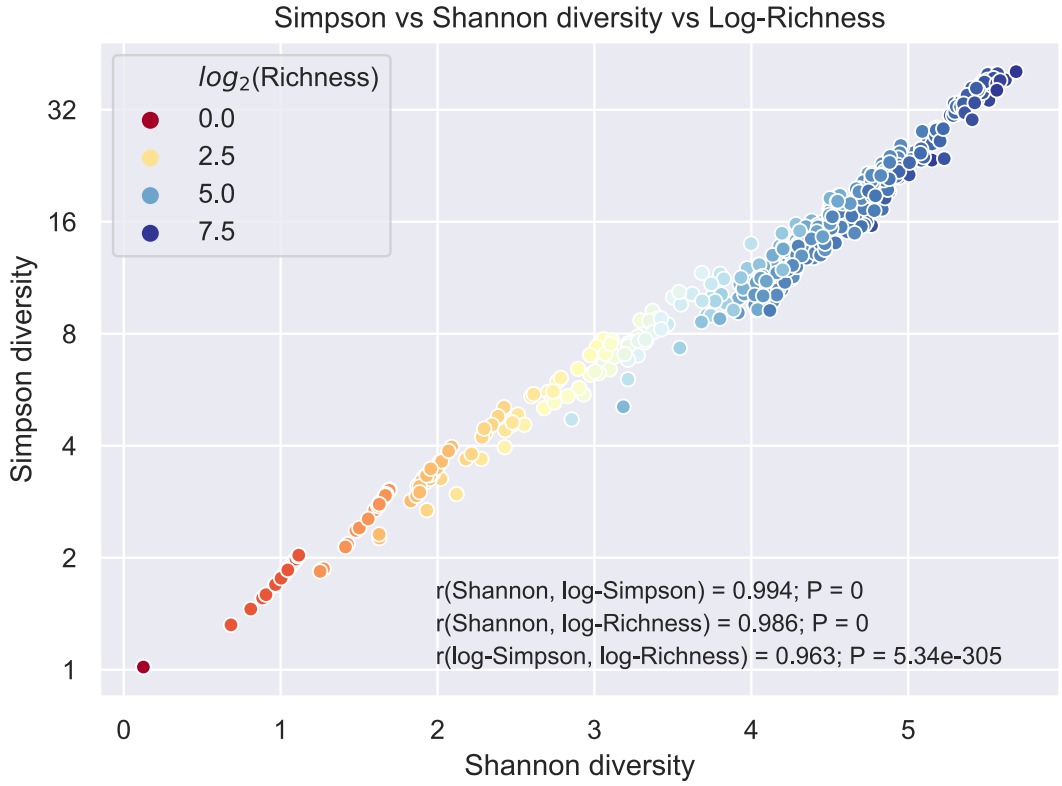

Figure 4.1. In the dataset `mp-pipeline_stats_hmm_pct_abundance_draconian`, different diversity metrics correlate strongly.

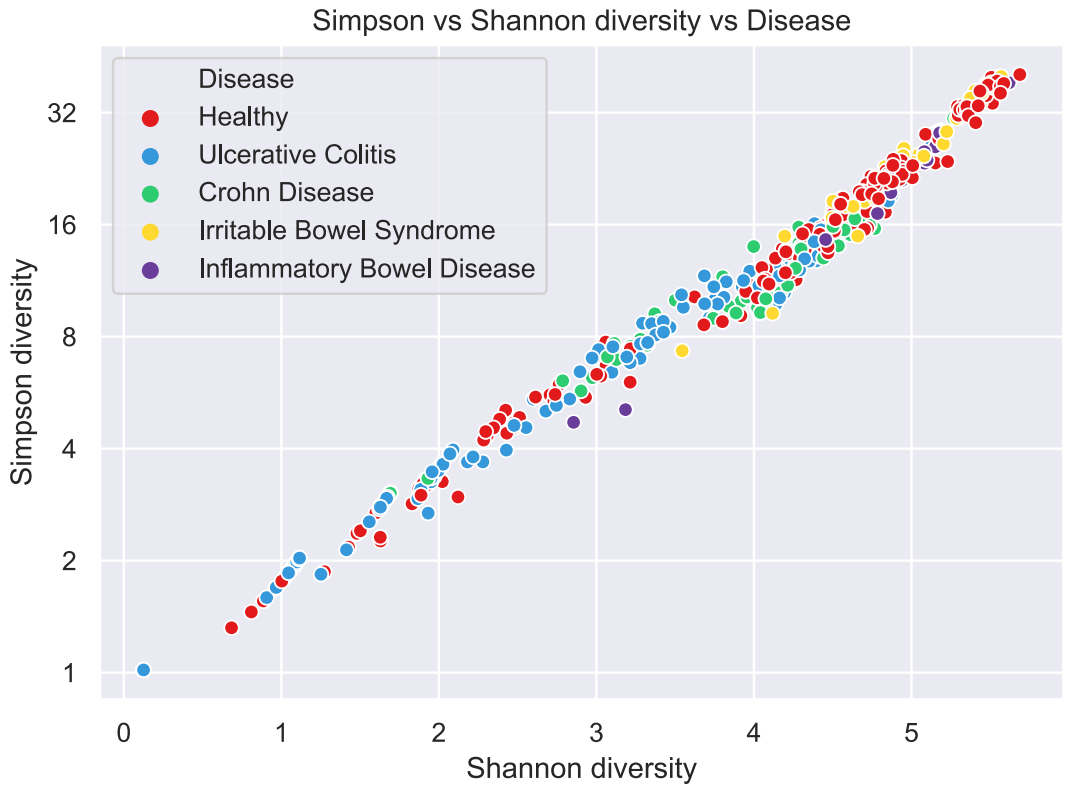

Figure 4.2. Data do not cluster strongly based on health and illness.

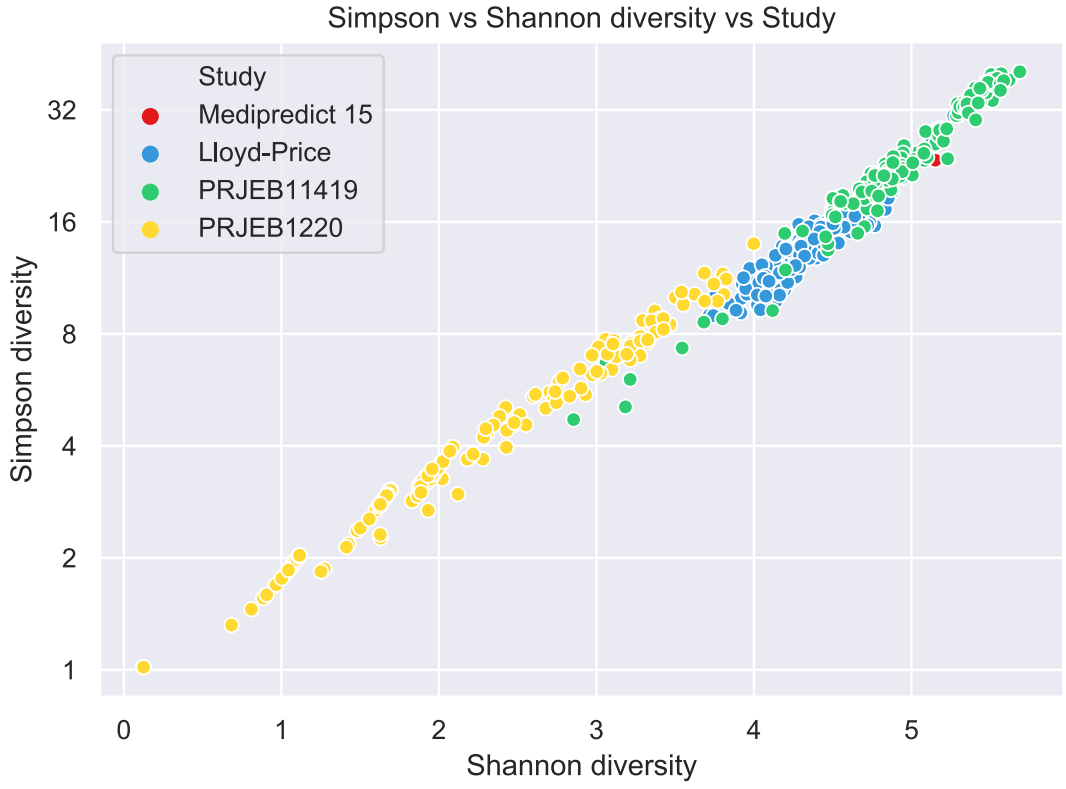

Figure 4.3. Clustering based on study (experimental setup) is well-visible.

For the second part, we used two standard dimensionality reduction models particularly suitable for visualization: Principal Coordinate Analysis and TSNE. Because TSNE does not work well with a very high number (several hundreds or thousands) of dimensions, Independent Component Analysis was used for lowering the dimensionality of the TSNE input space to 32. The scikit-learn library [33] was used for fitting the models to 19 different kinds of data sets as output by our pipeline as well as SHOGUN, each subject to the aforementioned 3 kinds of log transform. Qualitatively, both approaches resulted in largely the same projections, although sometimes PCoA plots looked nicer than TSNE plots and vice versa.

Again, we colored data points based on health status and study, as well as other potentially influential factors such as age and sex. Unfortunately, the outcome was once more uninteresting: data are well-separated along the dimension of different studies, but not with respect to health versus illness. Age and sex do not seem to be clustering factors either. In addition, this behavior appears consistent across data sets: the clarity of clustering does not depend much on whether taxa, KEGG modules, gene families, or metabolic pathways are considered, or whether abundances are transformed using CLR, ILR, or RLE.

The best-looking results were obtained from the `shogun_taxatable.bowtie2.pdf` dataset, transformed with centered log-ratio and relative log-expression. These are depicted in figures 4.4, 4.5, 4.6, and 4.7.

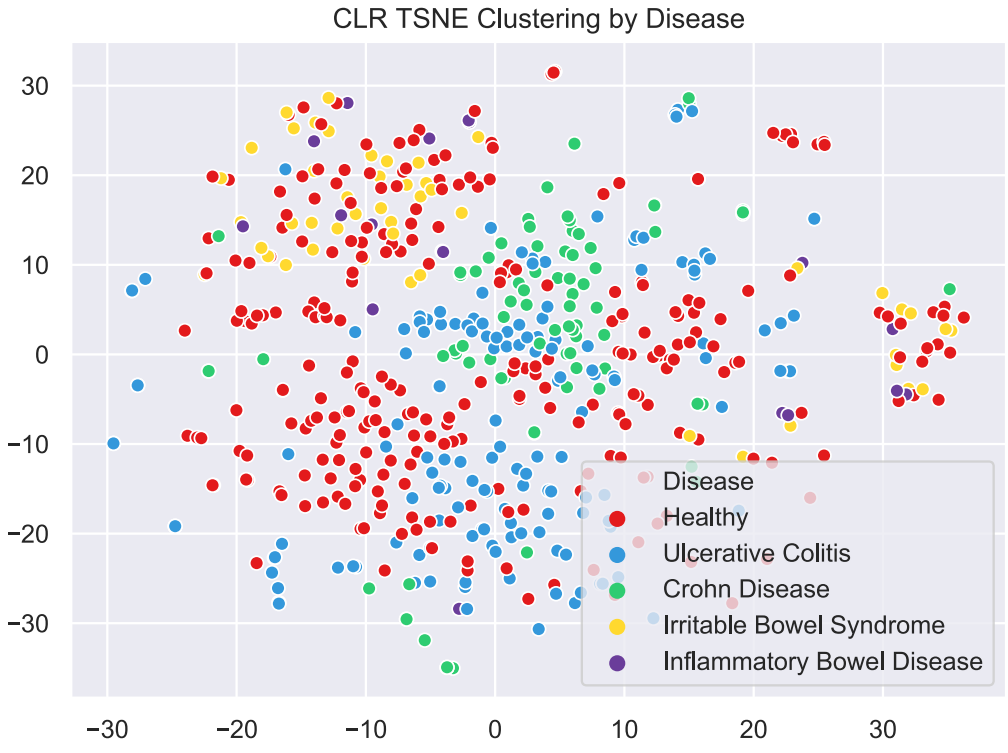

Figure 4.4. Clustering by disease as seen by TSNE after centered log-ratio transform.

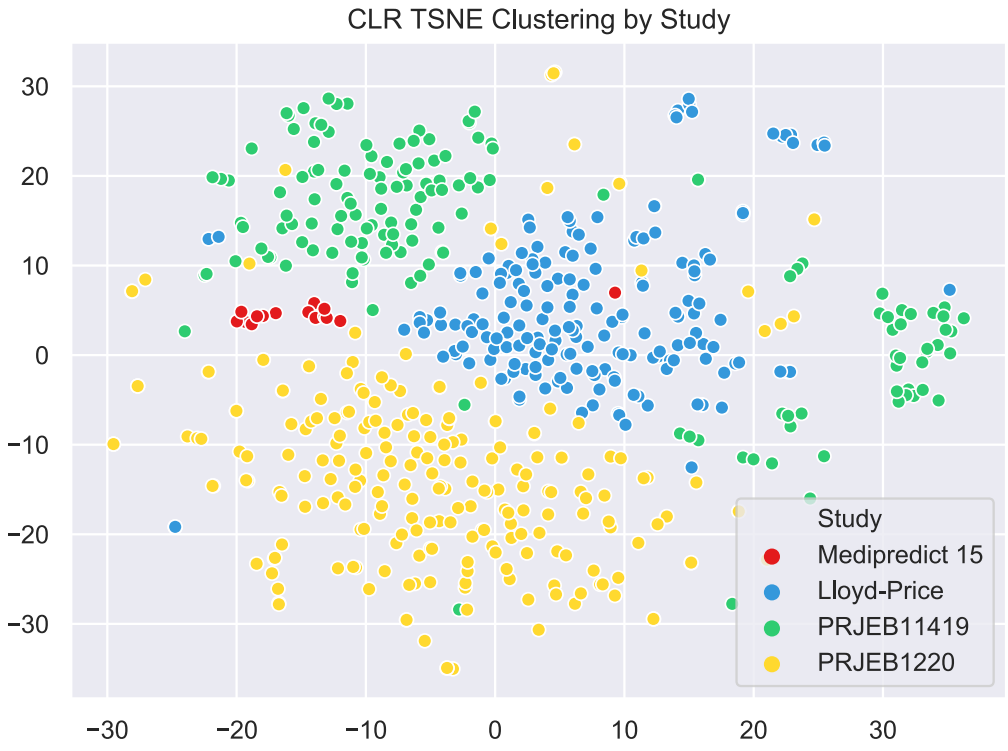

Figure 4.5. Clustering by study as seen by TSNE after centered log-ratio transform.

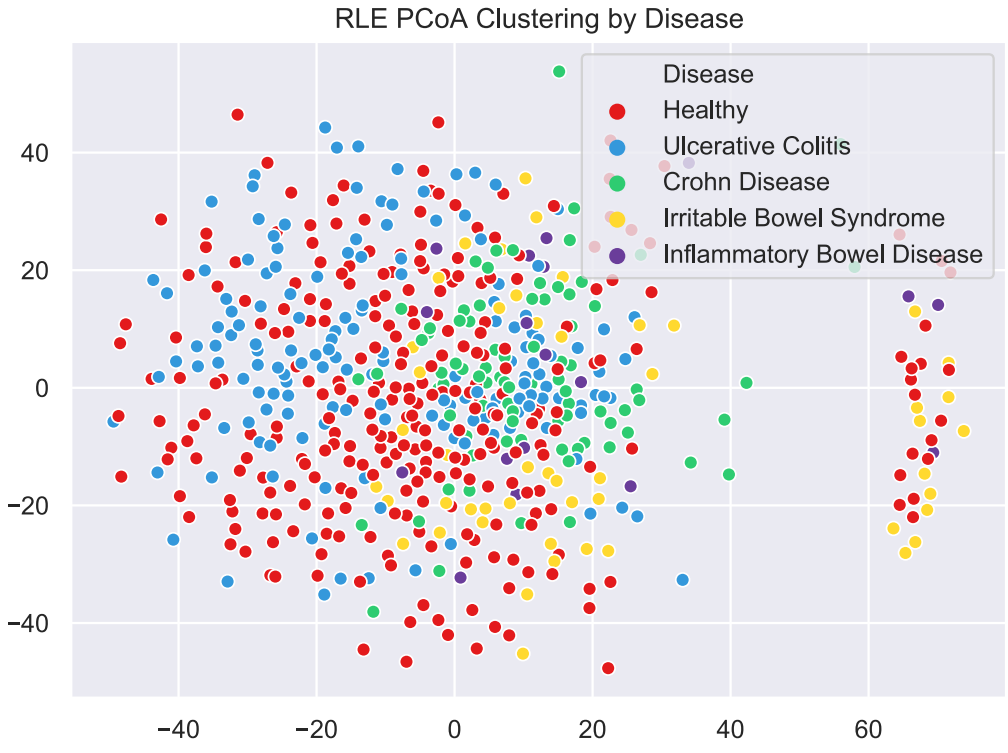

Figure 4.6. Clustering by disease as seen by PCoA after relative log-expression transform.

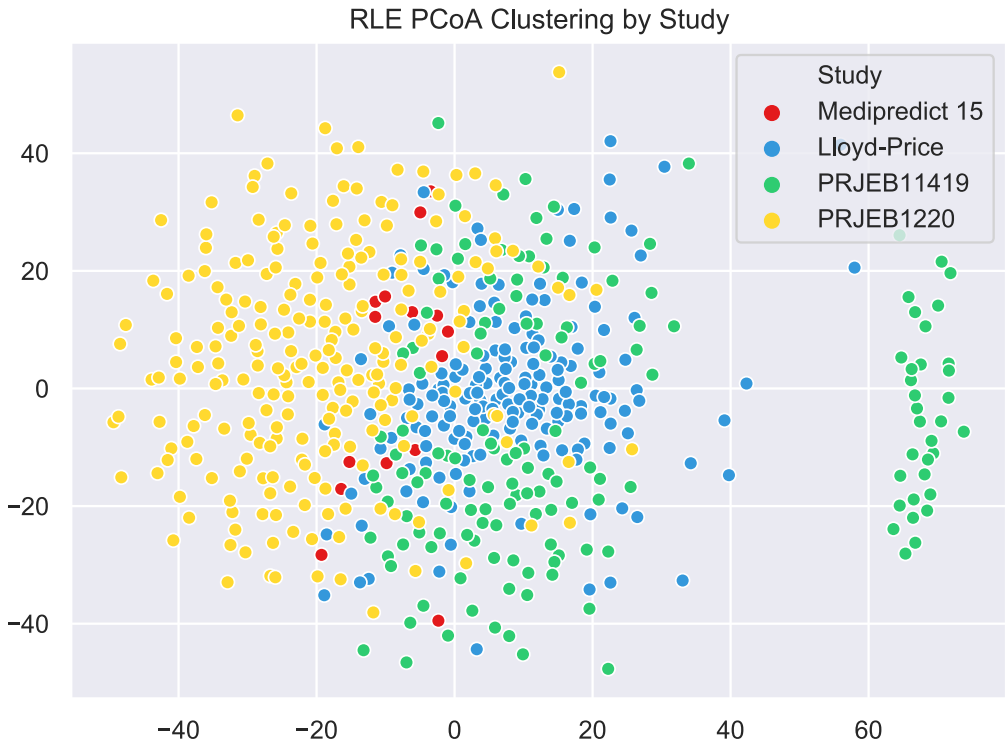

Figure 4.7. Clustering by study as seen by PCoA after relative log-expression transform.

In addition, we have performed some of the usual, but less advanced exploratory analyses, and plotted distributions of metadata, namely:

- Age, in order to ensure that most age groups are well-represented (figure 4.8).
- Diseases, in order to check how balanced or imbalanced the datasets are (figure 4.9);
- Countries, because geographical location is known to affect gut microbiome composition (figure 4.10);

Furthermore, for `mp-pipeline_stats_ref_seq_pct_abundance_placebo_CLR`, a smaller-dimensional dataset generated by our pipeline, Pearson correlationss between features were computed, as well as point-biserial correlations between each feature and the target label, encoding health as 0 and disease as 1. These are shown in figures 4.11 and 4.12), respectively. In the former, approximately square-shaped blocks of strong correlation can be observed. This is expected because the gene families are grouped by their biological function. In the latter, we can see that most correlation coefficients are almost zero, the largest ones being about ± 0.2. This is also expected, given the lack of disease-based clustering.

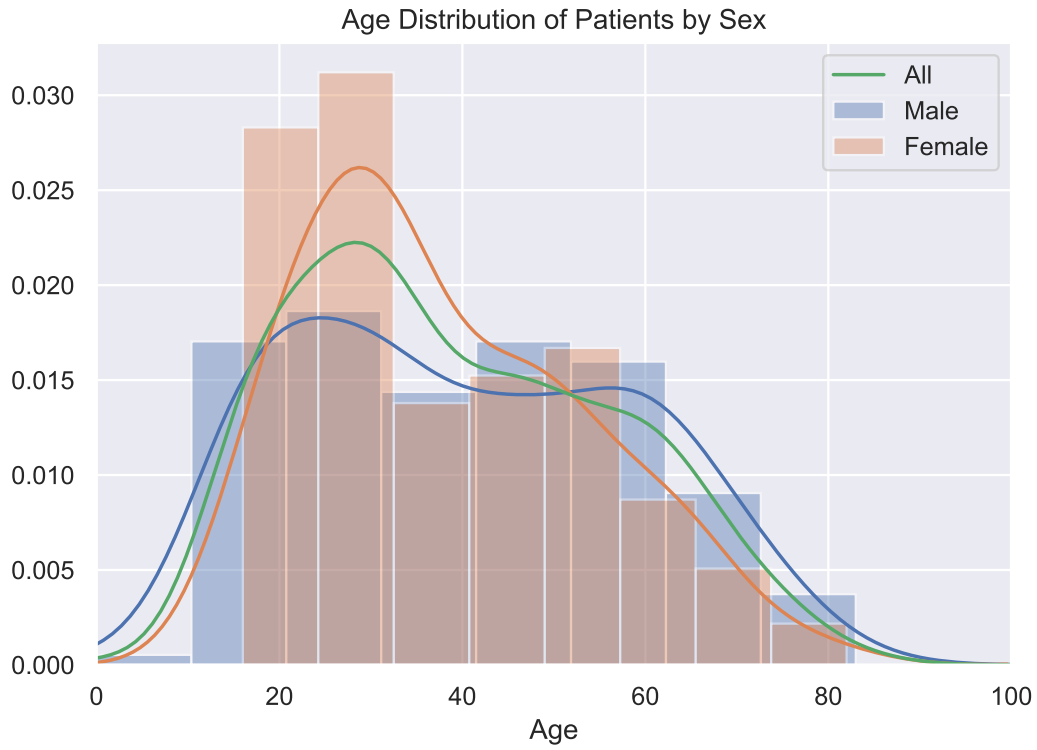

Figure 4.8. Distribution of age in the raw data sets, split by sex.

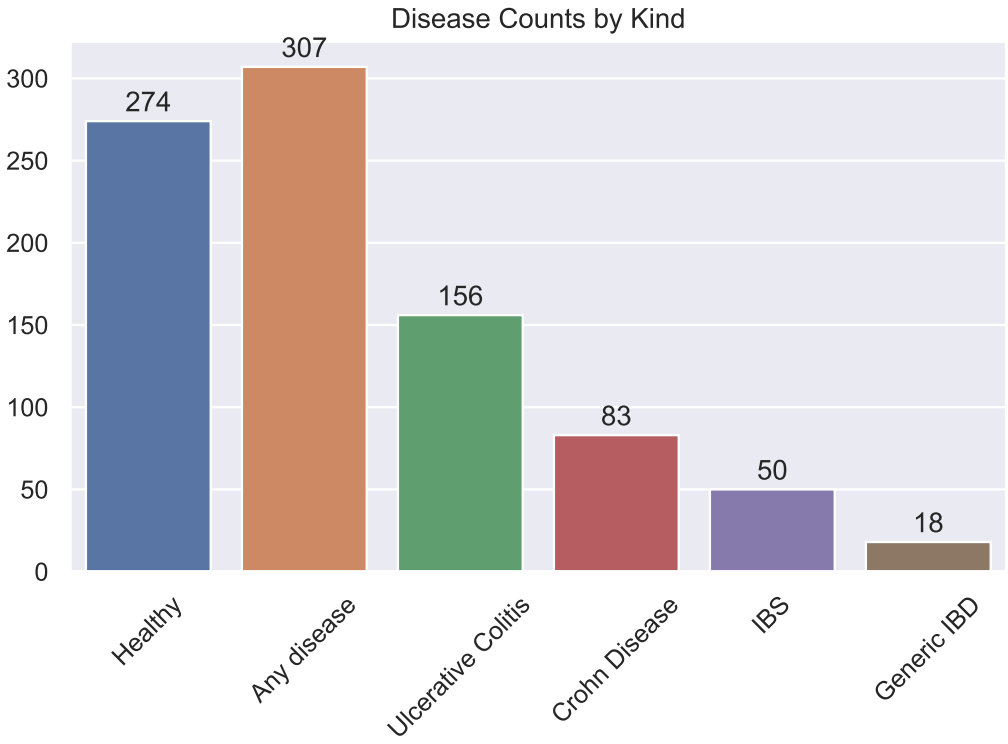

Figure 4.9. Distribution of diseases (and health) in the raw data sets.

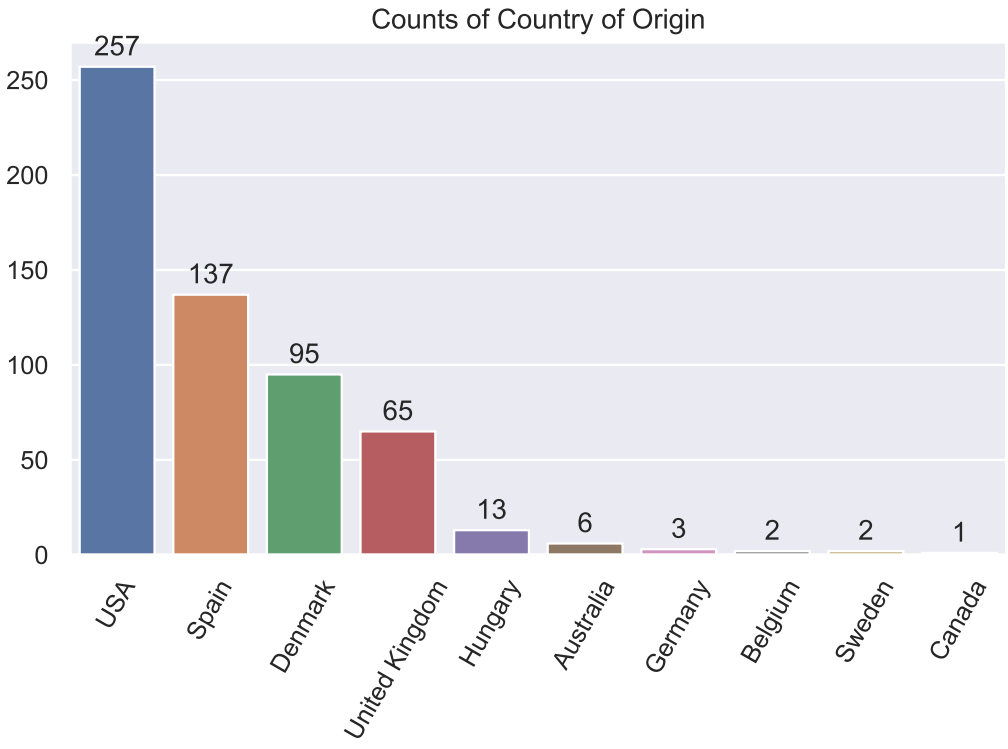

Figure 4.10. Distribution of countries in the raw data sets.

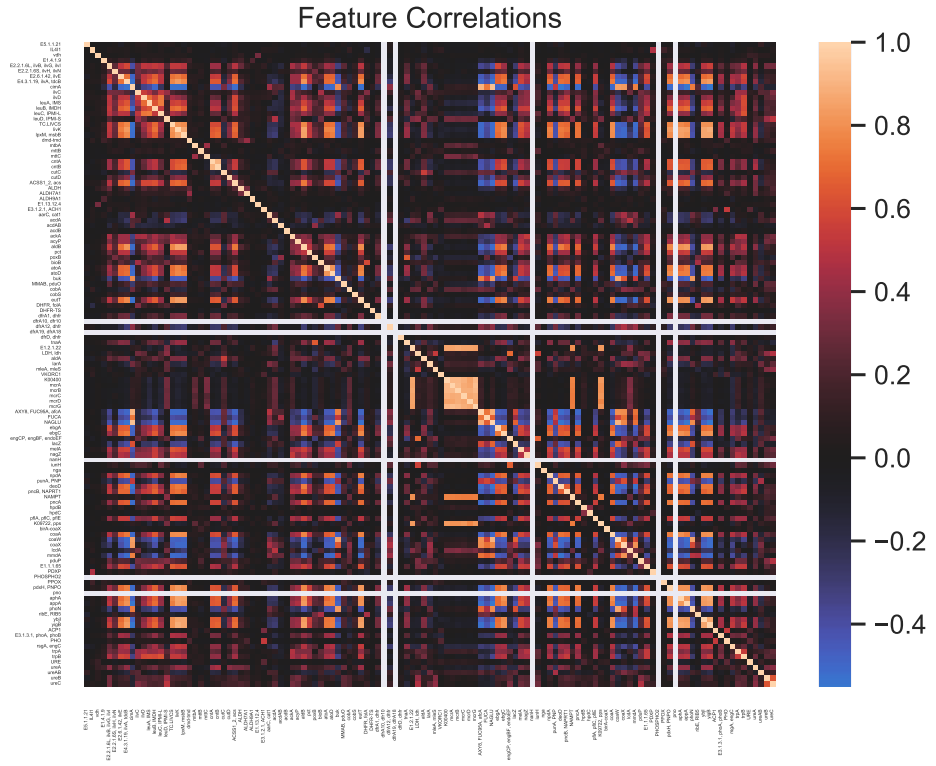

Figure 4.11. Pearson correlation coefficients between each pair of features.

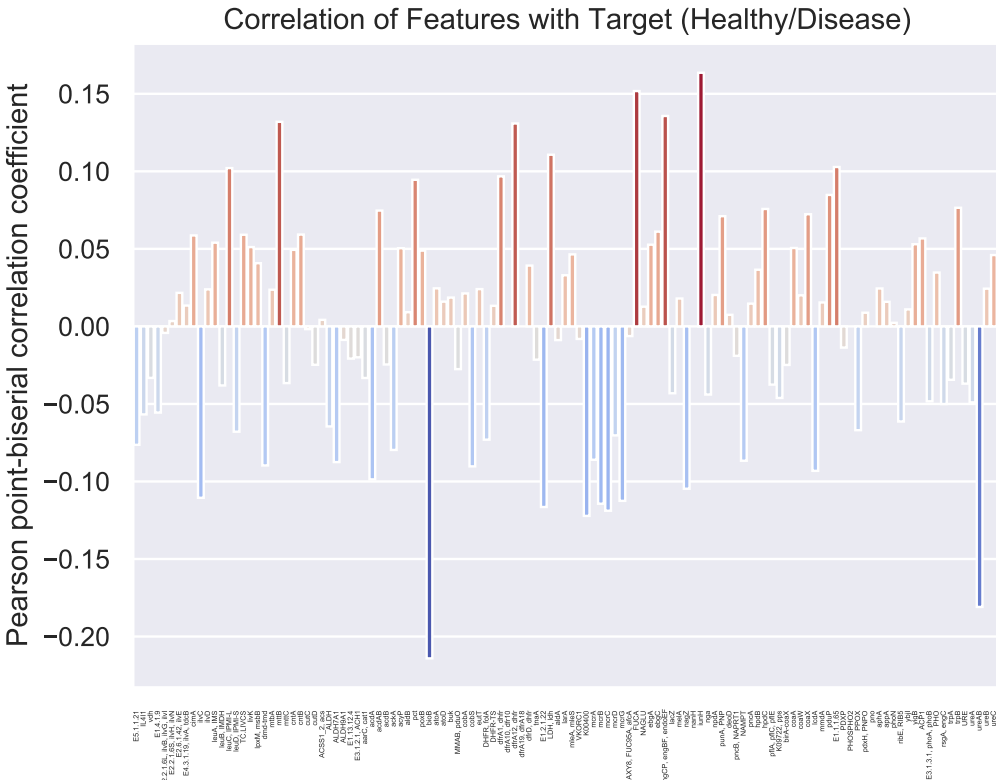

Figure 4.12. Point-biserial correlation coefficients between each feature and the target.

4.4. Building and Evaluating Classifier Models

There is a number of considerations when deciding on the particular statistical model:

- Given the relatively small number of data points (around 500), we need to fit simple, basic models on them.
- Because clustering based on the variable to be predicted (i.e. disease) is not at all strong, we will need to apply feature engineering methods in order to stand a chance of classifying samples.
- Given the high-dimensional nature of the data, we must once again use regularization.
- Another consequence of poor separation and high dimensionality is the tendency to overfit, so we need to be particularly careful about that, too.
- There is a relatively high number of data sets, so the model should be reasonably fast to train.

Considering all of these requirements, we have built a scikit-learn `Pipeline` with the following stages:

1. `RobustScaler`: Normalization to 0 median and unit interquartile range
2. `FastICA`: dimensionality reduction as pre-processing with Independent Component Analysis. We need this as a separate step because interaction features are added subsequently, which would be otherwise unfeasible, as it would result in tens of thousands or millions of features.
3. `PolynomialFeatures`: adding at most 2nd-order (interaction and quadratic) combinations of the columns produced by ICA.
4. `LogisticRegression`: logistic regression with ElasticNet (simultaneous L1 and L2) regularization. This is a very simple, relatively robust, and quite fast-to-fit model for binary classification. Another model we initially considered was a random forest, since in our experience, it is particularly effective when applied to biological data. However, it turned out to be unacceptably slow to fit to our many data sets, while performing slightly worse than logistic regression in terms of AUC.

We then split each of the 57 transformed data sets into training and test sets, where the test set was always 50 rows long. This strikes a balance between how representative it is for assessing the performance of a model and how much training data is lost. We fit the model to each training set, optimizing hyperparameters using 16-fold stratified cross-validation and the `BayesSearchCV` class of scikit-optimize [34]. The performance of the optimized models was evaluated on the corresponding test set using ROC curves. Confusion matrices were also built at the optimal decision threshold that maximized the difference between TPR and FPR, i.e. with decision function $y(x) = \mathbb{I}(x > \operatorname{argmax}_t \{TPR(t) - FPR(t)\})$.

The best-performing models were associated with the data sets in table 4.1. We note that most of the top-scoring models were fit to datasets generated by SHOGUN, and only one dataset generated by our own pipeline exceeded an AUC of 0.8.

We observed a tendency: there is a trade-off between AUC score and sensitivity to the decision threshold. The best, AUC = 0.93 model (figures 4.13 and 4.14) can't distinguish between the classes with high confidence. Sweeping with the decision boundary over the range $0.485 \ldots 0.514$, the TPR and FPR go from 1 to 0, which means that most predictions lie near 0.5, signaling that the model is unsure (table 4.2). Meanwhile, the worst model in the above table (figures 4.15 and 4.16), with AUC = 0.84, needs to have its decision boundary swept over the range of $0.11 \ldots 0.85$ for the same effect, indicating a much more robust model (table 4.3).

Overall, this seems very much like a bias-variance trade-off: high AUC scores mean seemingly more accurate but fragile models, suggesting that they overfit, whereas low AUC scores correspond to less accurate (thus, underfit) but more robust models. Figure 4.17 contrasts the ROC AUC score and the dispersion of the necessary decision boundaries of each aforementioned model.

Dataset of relative abundances	**Transform**	**AUC**
shogun_taxatable.strain	CLR	0.93
shogun_taxatable.strain.normalized	CLR	0.92
shogun_taxatable.strain.normalized	RLE	0.92
shogun_taxatable.strain.kegg	ILR	0.91
shogun_taxatable.bowtie2	RLE	0.90
shogun_taxatable.strain.kegg	RLE	0.87
mp-pipeline_stats_ref_seq_pct_abundance_placebo	CLR	0.84

Table 4.1. Data sets for best-performing models.

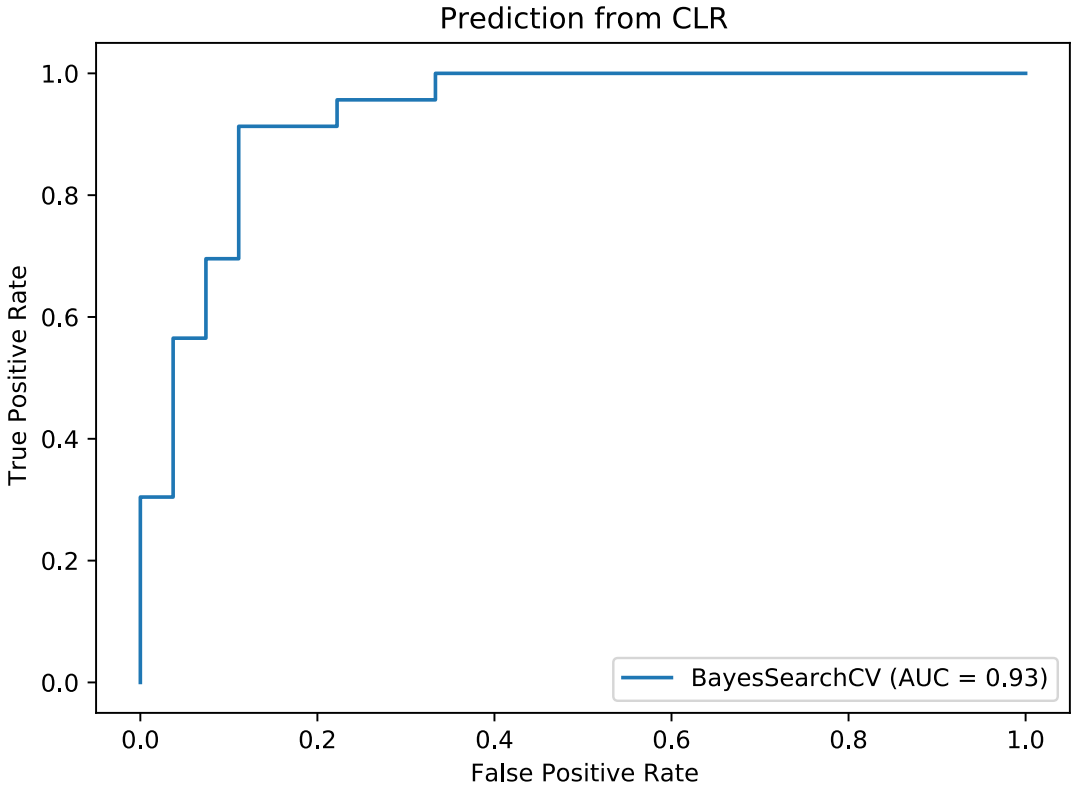

Figure 4.13. ROC curve of the best-performing model.

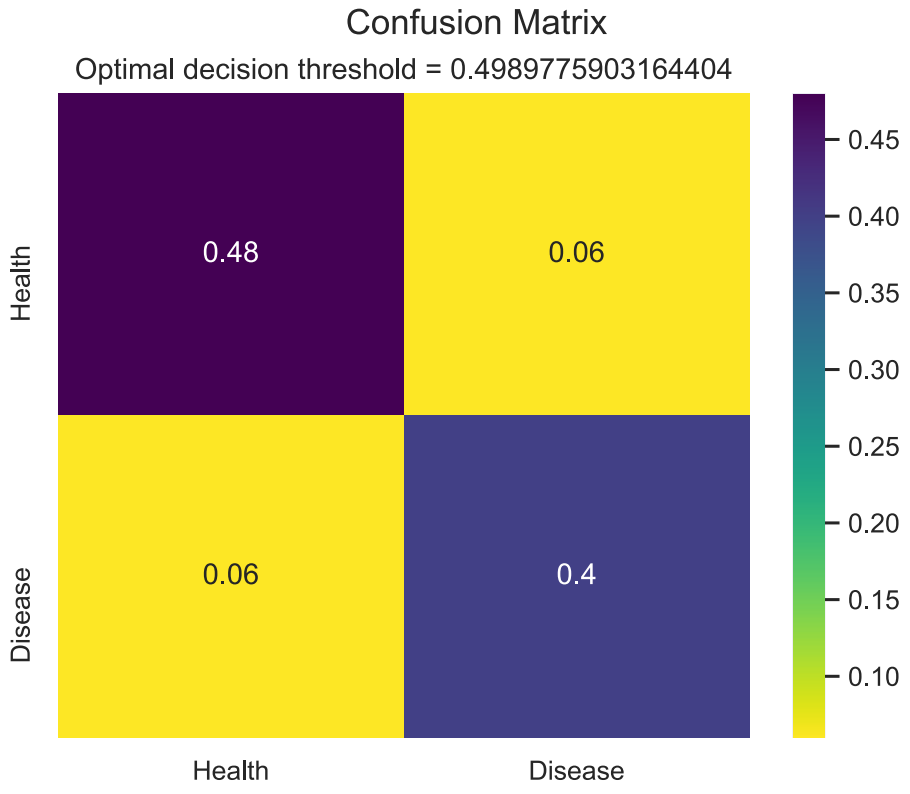

Figure 4.14. Confusion matrix of the best-performing model.

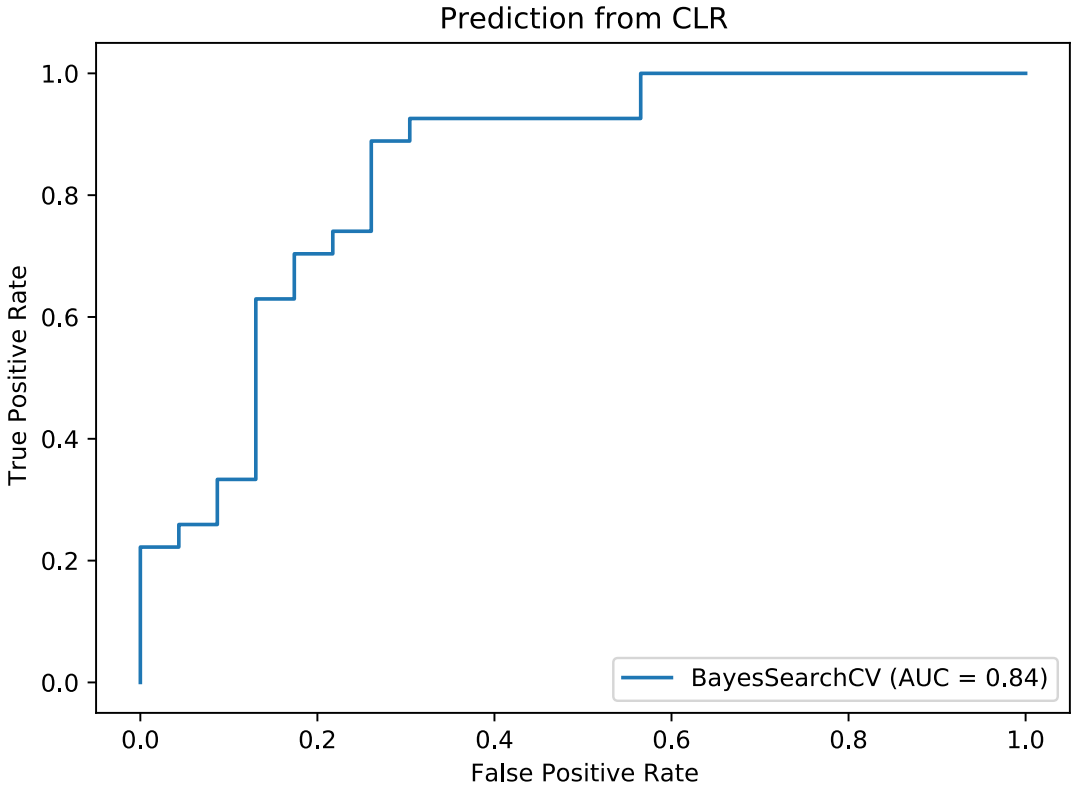

Figure 4.15. ROC curve of the least well-performing model among the top 7.

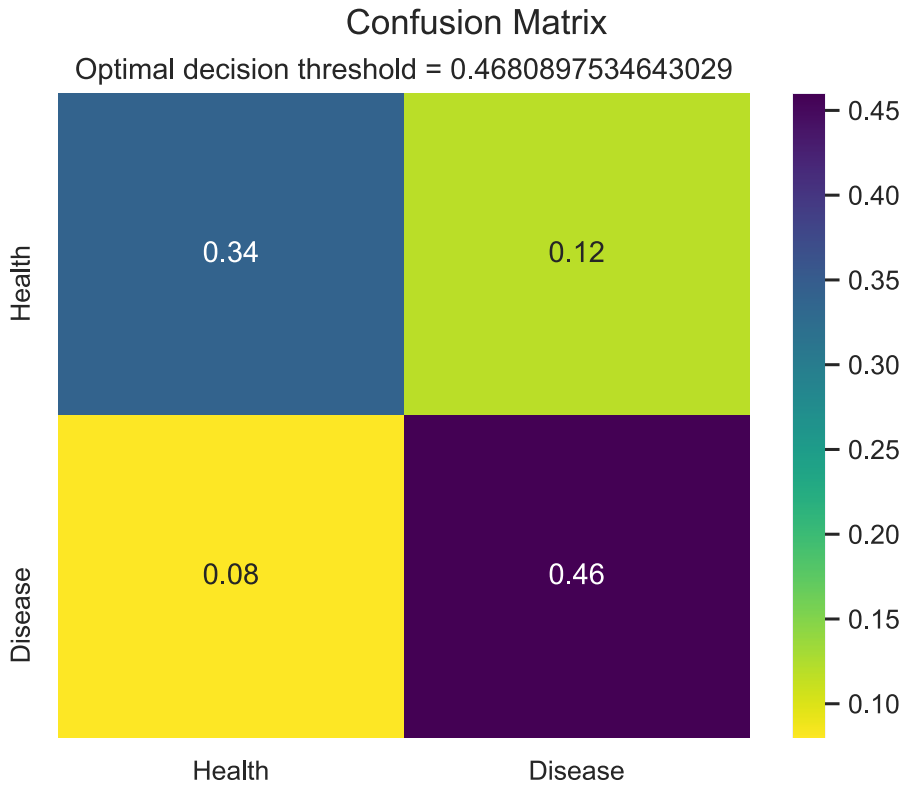

Figure 4.16. Confusion matrix of the least well-performing model among the top 7.

Threshold	TPR	FPR	TPR − FPR
1.513653	0.000	0.000	0.000
0.513653	0.043	0.000	0.043
0.505026	0.304	0.000	0.304
0.504775	0.304	0.037	0.267
0.503001	0.565	0.037	0.528
0.502691	0.565	0.074	0.491
0.501522	0.696	0.074	0.622
0.501088	0.696	0.111	0.585
0.498978	**0.913**	**0.111**	**0.802**
0.498499	0.913	0.222	0.691
0.498363	0.957	0.222	0.734
0.497484	0.957	0.333	0.623
0.497244	1.000	0.333	0.667
0.485408	1.000	1.000	0.000

Table 4.2. Decision Boundaries vs. Positive Rates of the best-performing model.

Threshold	TPR	FPR	TPR − FPR
1.848351	0.000	0.000	0.000
0.848351	0.037	0.000	0.037
0.752065	0.222	0.000	0.222
0.739061	0.222	0.043	0.179
0.723762	0.259	0.043	0.216
0.702240	0.259	0.087	0.172
0.664717	0.333	0.087	0.246
0.660855	0.333	0.130	0.203
0.557606	0.630	0.130	0.499
0.553100	0.630	0.174	0.456
0.518168	0.704	0.174	0.530
0.511729	0.704	0.217	0.486
0.507905	0.741	0.217	0.523
0.496029	0.741	0.261	0.480
0.468090	**0.889**	**0.261**	**0.628**
0.451621	0.889	0.304	0.585
0.450763	0.926	0.304	0.622
0.387802	0.926	0.565	0.361
0.385484	1.000	0.565	0.435
0.114481	1.000	1.000	0.000

Table 4.3. Decision Boundaries vs. Positive Rates of the less well-performing model.

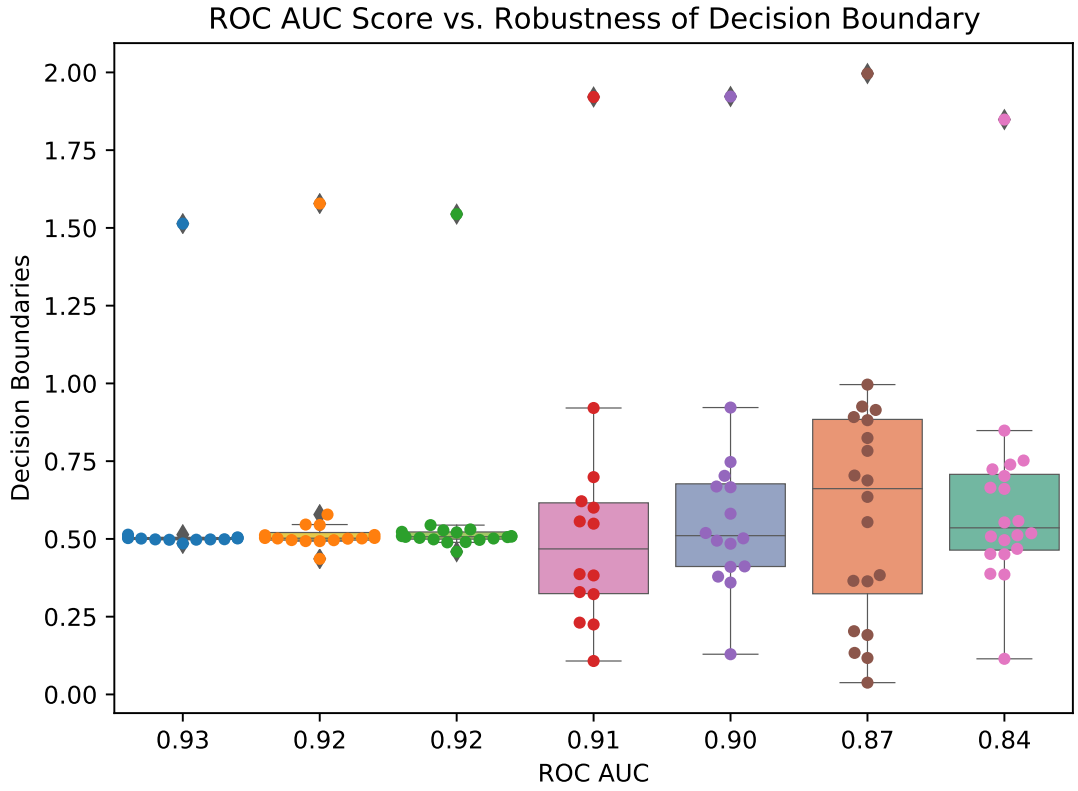

Figure 4.17. Dispersion of decision boundaries, and thus, robustness of the model, decreases with increasing ROC AUC.

Lastly, we examined feature importance. Since the models consist of several transformation steps, it is not easy to directly compute exact importance indices from the model. For example, we could take the coefficients of the logistic regression, but those would relate to the interaction terms, which in turn are also computed from a linear combination of the inputs (after ICA). This means that a straightforward biological interpretation would practically be impossible.

Therefore, we turned to approximations. One such metric is permutation importance, which measures how much the performance of the model drops if a given feature is randomly shuffled across samples. This method is theoretically appealing, although not without flaws. For example, highly-correlated features might not be assigned a high enough importance score, and indeed, our dataset has some highly-correlated features. We decided to ignore this fact for the sake of simplicity – we are only looking for a quick at-a-glance measure of feature importance here.

Permutation importances of the top-scoring features for a lower-dimensional dataset are plotted in figure 4.18. We can observe that only a few of them are statistically significant.

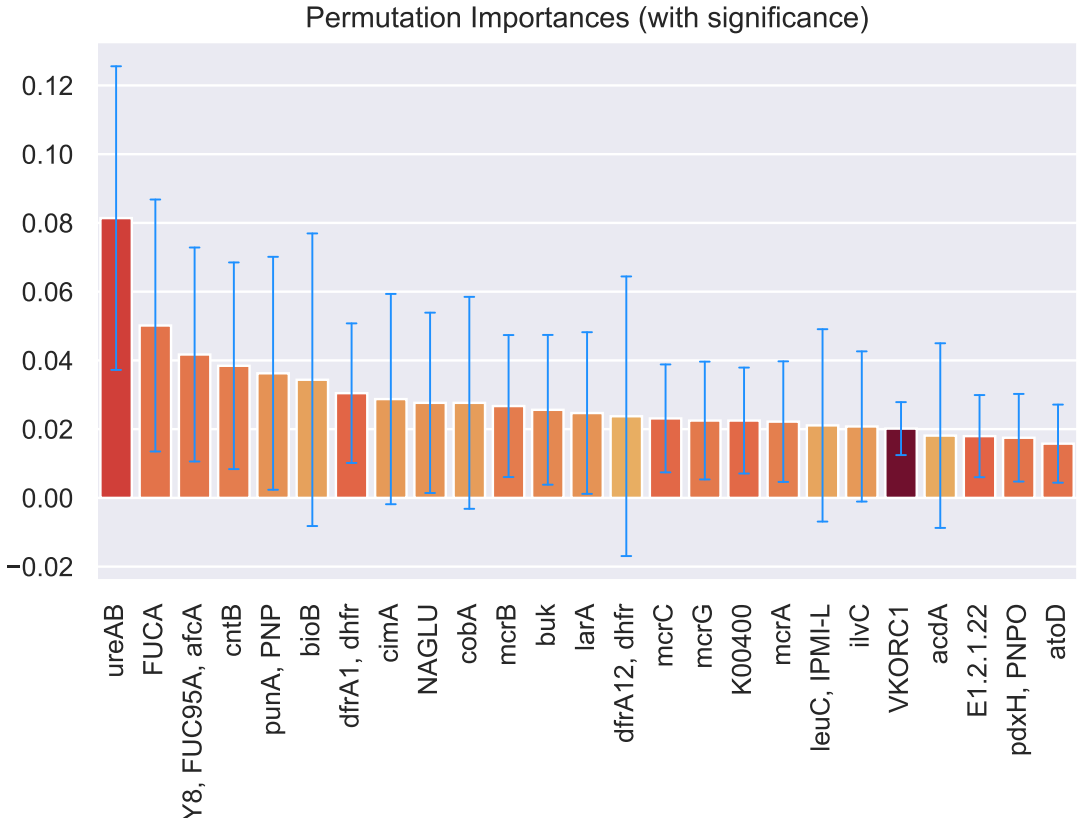

Figure 4.18. The 25 features with the highest permutation importance scores, averaged over 128 random permutations of each feature, in the dataset `mp-pipeline_stats_ref_seq_pct_abundance_placebo_CLR`. Bars are colored according to statistical significance (logit of the P-value), the darker the more significant. Error bars show ±1 stddev.

5. Microbial Correlation Networks

In this chapter we describe how abundance data can be used for discovering the interaction structure of the gut microbiome.

5.1. Generating Network Data

The most obvious way of describing potential interactions between microbiota is the correlation network. A correlation network is a graph such that two vertices are connected with an edge if and only if they are correlated with one another. In addition, a weighted correlation network contains edges with different weights, according to the strength of the correlation between the relevant vertices.

We wanted to run a number of different analyses in an automated manner, so we decided to generate unweighted networks. These are the simplest and most special kind of graphs, and most of the graph-theoretic models work well on them, a fact that makes them a good default for general preliminary or exploratory analysis. However, while this is an important advantage, there are also drawbacks to this decision, which are discussed in Chapter 6.

For reasons already explained, it is not sound to compute correlations directly from relative abundances. It is not useful to correlate columns of the IRL-transformed data either, because the IRL transform mixes dimensions with one another, so the resulting correlations and graphs will not have a clear interpretation. Therefore, we only computed correlations between columns of data after applying CLR and RLE transforms – this is similar to the procedure recommended in [35].

The exact procedure we carried out for each data set was the following. We computed Spearman's rank-order correlation between each pair of variables. We used this instead of Pearson's because there is no basis for assuming an exact linear relationship. We then added an edge between the variables as vertices if the correlation exceeded $\rho > 0.7$ and it was simultaneously significant at the $P < 0.001$ level. This is a pretty conservative approach, because we wanted to filter out as many artifactual correlations as possible. We have a good reason to believe these

exist: raw abundance data are sparse, and sparsity inflates correlations. (More on this problem in chapter 6.)

Finally, self-edges were removed and only the largest connected component of the graph was retained, because some of the analyses rely on connectedness of the subject graph. Node names (for example, KEGG IDs) were obtained and stored separately, so that visualizing a network (or part of it) would result in human-readable, easy-to-interpret plots.

It should also be noted that the above procedure ignores negative correlations altogether. This decision was made because negative correlations (and signed edges) are more difficult to interpret.

The resulting set of correlation networks was shrunk further. We only retained large enough adjacency matrices which were at the same time sufficiently sparse. The reasons for this are threefold:

1. Results of network science are usually stated in terms of asymptotics as the number of vertices and/or edges tends to infinity. Therefore a large adjacency matrix is needed to meet the assumptions behind a given calculation. Accordingly, the smallest generated correlation network was on the order of 400 vertices.

2. Real networks are usually sparse, so a dense adjacency matrix likely indicates that the dataset does not reflect reality, especially because we know our data are prone to generating excess correlations.

3. A network with tens of millions of edges is not computationally tractable within a reasonable amount of time with the small-scale computing resources we had access to.

This process was then repeated three times for each CLR and RLE-transformed dataset: once for the complete dataset, once for the subset of ill subjects, and once for the subset of healthy controls. This allows us to make inferences (albeit not in the strict, statisticaly exact sense) about the differences between the ill and healthy subgroups. Included in the final analysis were all three variants (combined, disease, healthy) of each of the following datasets:

- `shogun_taxatable.strain.kegg_CLR`
- `shogun_taxatable.strain.kegg.modules_CLR`

- `shogun_taxatable.strain.normalized_CLR`

We can see that only CLR-transformed data met the above inclusion criteria. The reason for this was that the RLE-transformed datasets exhibited an excessively high number of correlations, resulting in unreasonably dense networks.

5.2. Results and Interpretation of the Analyses

The dataset `shogun_taxatable.strain.normalized_CLR_combined` yielded the most illustrative, representative results. Therefore in the next paragraphs, we exclusively describe this dataset, unless otherwise noted. We also note that in all three groups of datasets mentioned above, **results obtained from the healthy control subset are virtually identical to those obtained from the corresponding subset of ill patients.** We therefore conclude that the structure of correlation networks is largely independent of whether they have been generated from data of healthy or IBD patients.

5.2.1. Degree and Distance Distribution

The distance distribution is close to expected (figure 5.1): its mode and median is 8, while the 90th percent effective diameter is 11, which is comparable to the metrics of typical small-world networks (cf. the popular idea of "six degrees of separation"). The logarithm of the complementary CDF of the degree distribution above $k \geq 10$ fits a line almost perfectly (figure 5.2), with a corresponding power-law coefficient (degree exponent) of $\gamma = 3.785$. According to Barabási [36], in the range $\gamma \geq 3$, the network essentially behaves like a random graph, making it harder to prove its scale-free nature, if any.

5.2.2. Assortativity

The network shows a strong and significant assortative tendency, with a linear regression coefficient of $\beta = 0.502$, $P = 1.442 \cdot 10^{-7}$ (figure 5.3). This disappears (figure 5.4) under degree-preserving randomization ($\beta_{rand} = -0.0356$, $P = 0.122$). However, typical biological networks are expected to exhibit disassortativity, so either this correlation network is atypical, or, as suggested by the degree exponent, it is merely random.

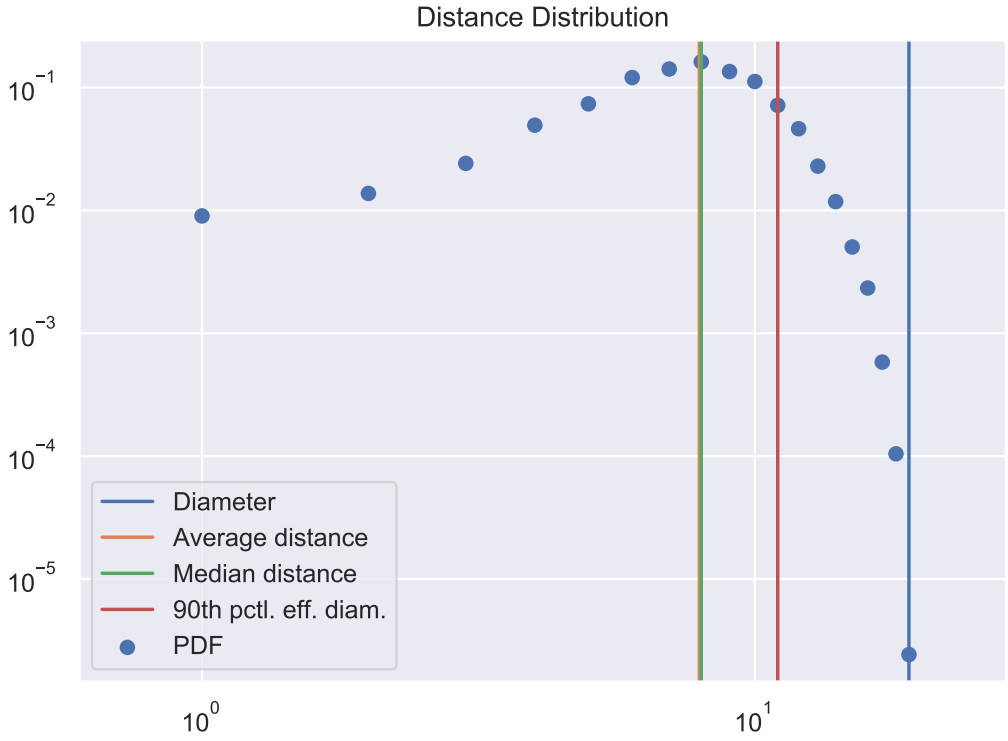

Figure 5.1. Distance distribution of the examined network, i.e. the relative frequency of shortest paths of length $1 \leq d \leq 19$.

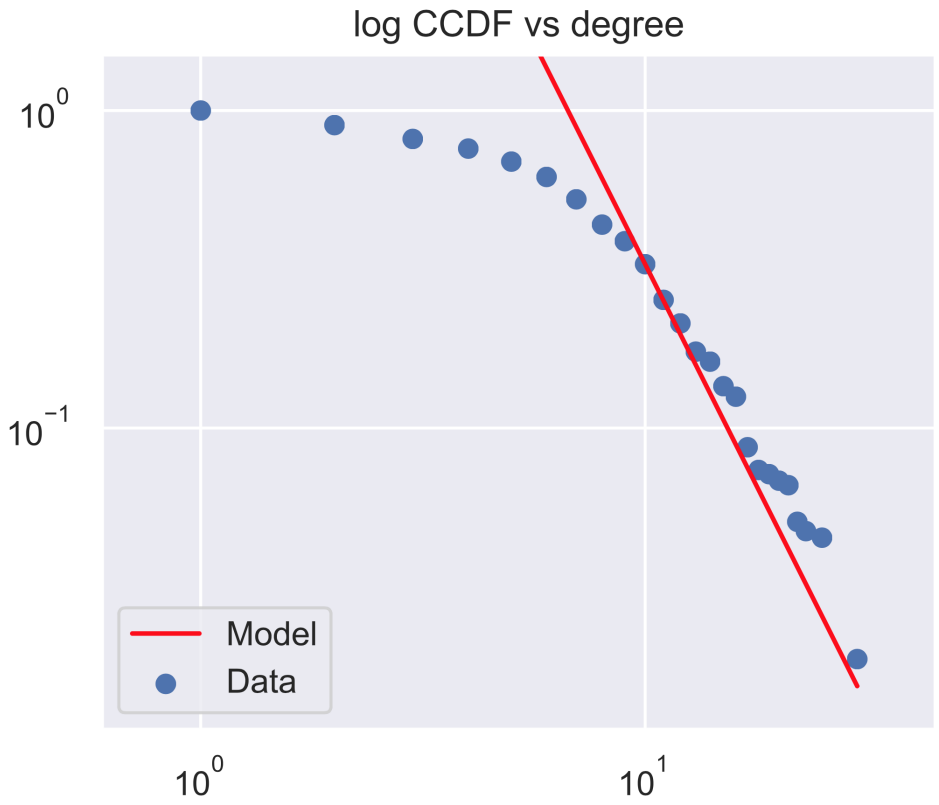

Figure 5.2. Degree distribution of the examined network. The tail of the complementary cumulative distribution function decays almost perfectly exponentially.

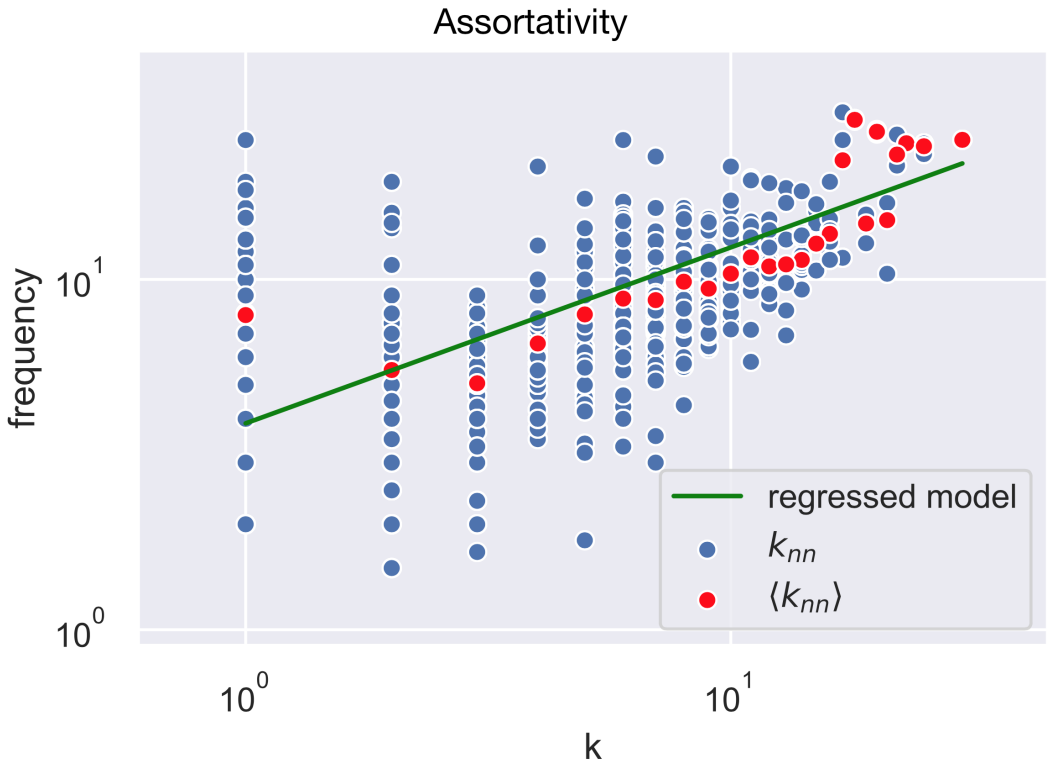

Figure 5.3. Assortativity plot of the network: average degree of neighbors $\langle k_{nn} \rangle$ as a function of k.

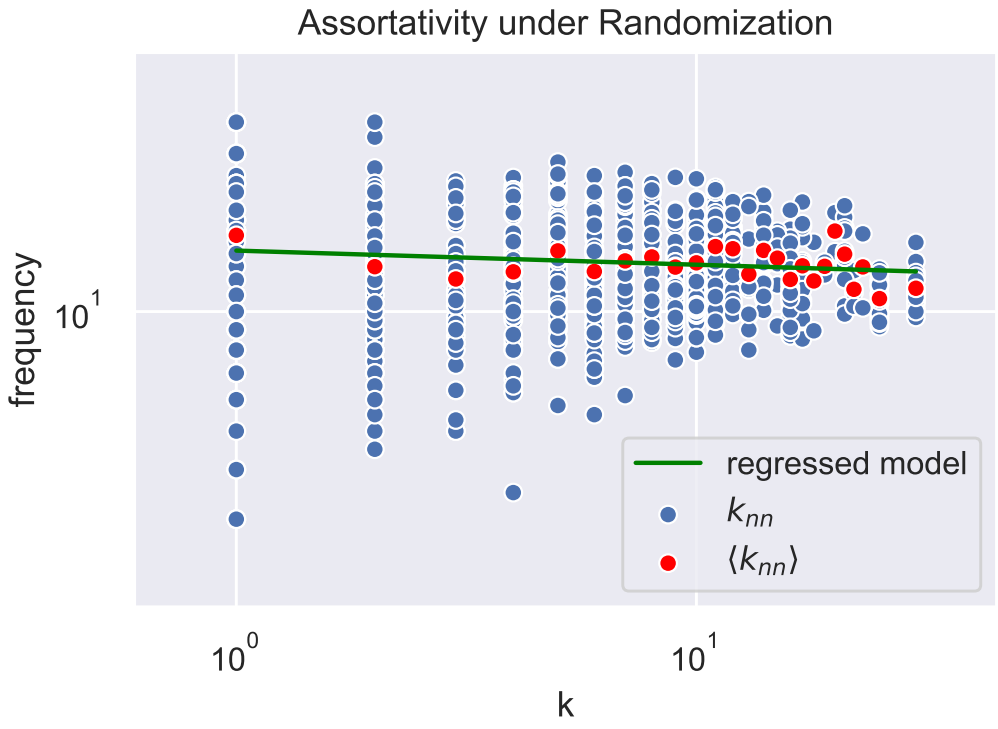

Figure 5.4. Assortativity disappears under randomization.

5.2.3. Robustness under Attacks and Random Failure

The robustness of the network under attacks was assessed by systematically removing the highest-degree node, then the second highest-degree node, etc. Meanwhile, robustness under the removal of random nodes ("random failure") was evaluated on 8 independent realizations of the randomly-modified graph, for each node removal rate $f = 0.0, 0.04, 0.08, \ldots, 0.96, 1.0$. Four quantities were then plotted against f:

- Absolute number of connected components
- Absolute size of (number of nodes in) the Giant Component, i.e. the largest remaining connected component ($|GC_f|$)
- Relative size of the GC with respect to the total number of nodes (N_f)
- Relative size of the GC with respect to the size of the GC in the original network ($|GC_0|$)

A sudden decrease in the size of the giant component and its subsequent shrinkage towards 0 signifies the so-called breaking point f_0 of the network, whereby its structure falls apart and it becomes a loose collection of very small sets of vertices. It is apparent from figures 5.5 and 5.6 that the network is of course somewhat more resilient against random failures ($f_0^{(rand)} \approx 0.4$) than it is against attacks ($f_0^{(atk)} \approx 0.8$), although it is not very robust in eiher case. In particular, the size of the giant component starts to drop quickly and immediately, even at $f = 0$. That is, the graph of the GC size is approximately convex in both cases, whereas it would appear concave in the presence of random failures if the network were truly scale-free.

The biological interpretation of "attacks" and "random failures" in the context of a network of microbial taxa or genes is actually pretty straightforward. If we assume that (positive) correlations arise out of true interactions (e.g. mutualism or commensalism) or due to a common third (e.g. environmental) factor affecting both variables, then the removal of nodes may be interpreted as the corresponding species or gene ceasing to function. For example, taking antibiotics might be considered an attack if it kills key species rich in interactions, meanwhile random fluctuations in the availability of diverse and healthy nutrients might cause decreased activity or extinction of any species, constituting a random failure.

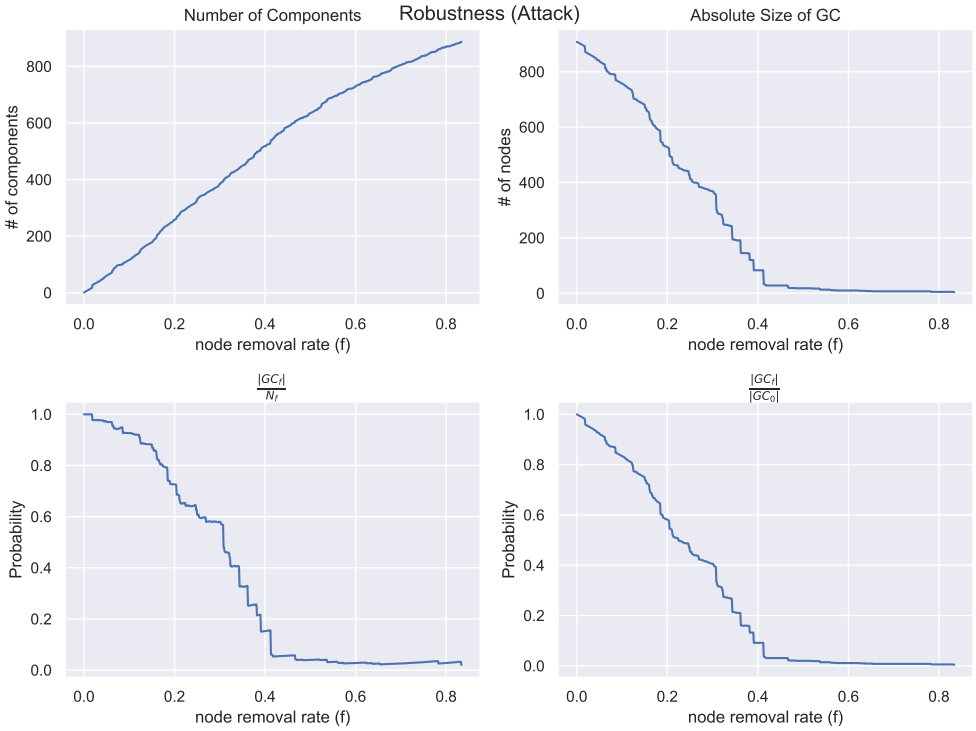

Figure 5.5. Robustness under Attacks.

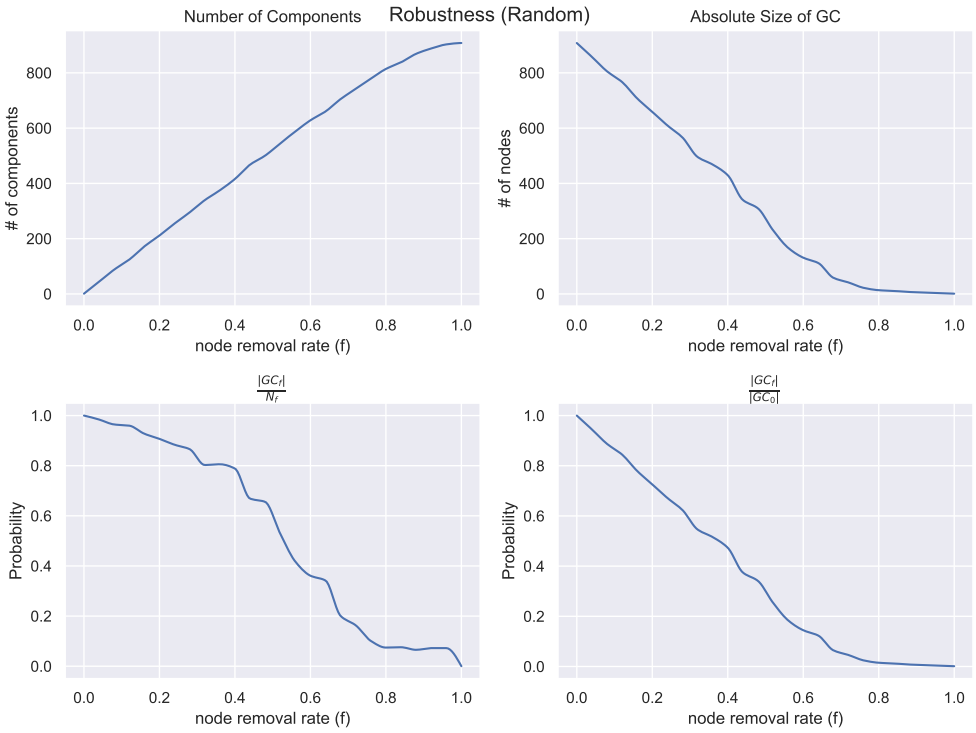

Figure 5.6. Robustness under Random Failure.

5.2.4. Vertex Importance using PageRank

The most central or "influential" species were defined with the help of PageRank. The usual damping factor of $\beta = 0.85$ was used. Among the top-scoring vertices are mostly bacteria, but some of them correspond to bacteriophages, as shown in figure 5.7. Some bacteria (such as *Candidatus Arthromitus*) are commensalists, while others (such as *Chlamydia avium*) are pathogenic. Altogether, there is no obvious pattern to be observed, nor any strong evidence for the dominance of a particular taxonomic or functional group. This also suggests that the network being analyzed wasn't very evocative of the real interaction structure of the microbiome to begin with.

5.2.5. Community Detection with Spectral Clustering

Communities, i.e. subgraphs of high internal connection density, were identified using spectral clustering. The optimal number of clusters was determined by maximizing modularity. The graph was repeatedly clustered into $N = 1, 2, \ldots, 50$ communities, and the $N = 49$ partition maximized modularity, with $Q_{max} = 0.857$.

However, plotting the modularity score against the number of communities (figure 5.8) reveals that modularity does not increase significantly after $N = 24$, while plotting the matrix of intra-community and inter-community connection densities (figure 5.9) also makes it clear that some communities are quite densely connected to many others. For instance, community #39 is connected to the majority of other clusters. This suggests that perhaps the mathematically-optimal number of clusters, 49, is too high, and we are in fact overfitting. We are thus inclined to instead declare $N = 24$ as the most realistic community count, as it is the lowest number of clusters with which modularity is still almost maximal.

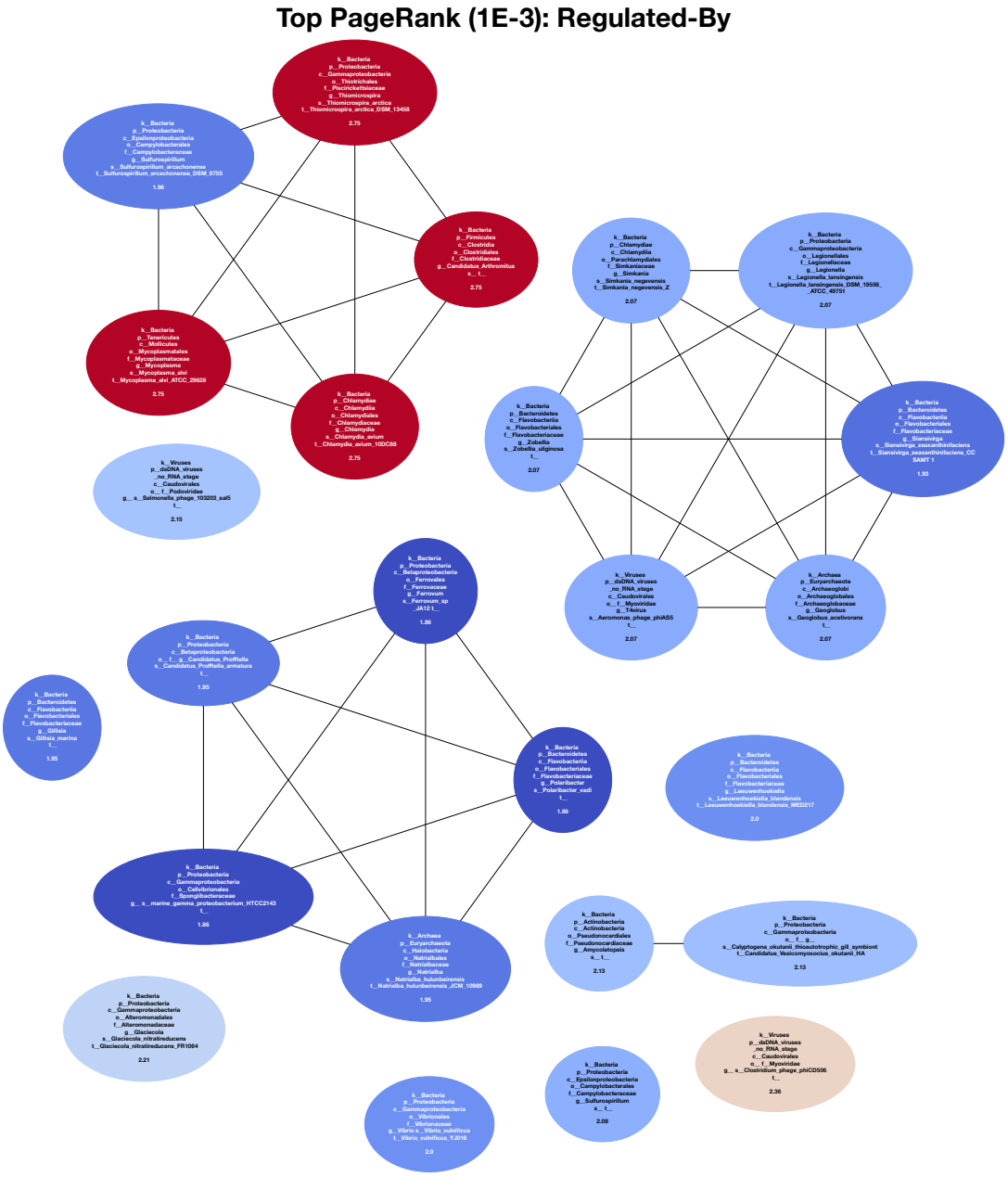

Figure 5.7. Top-scoring vertices according to PageRank.

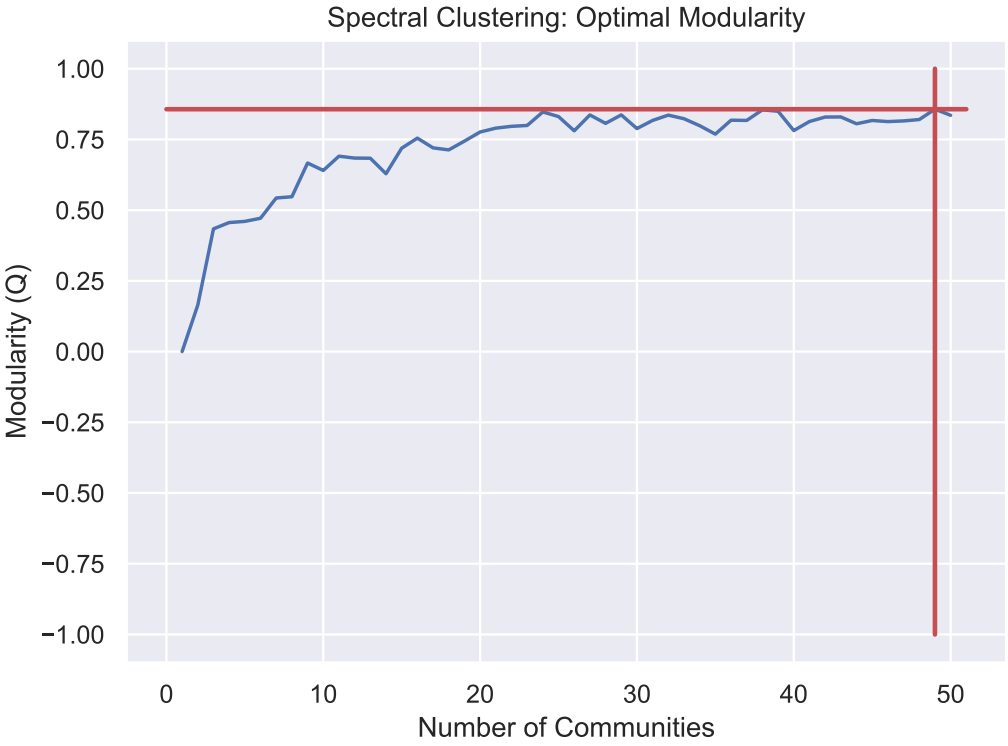

Figure 5.8. Modularity coefficient Q as a function of community count.

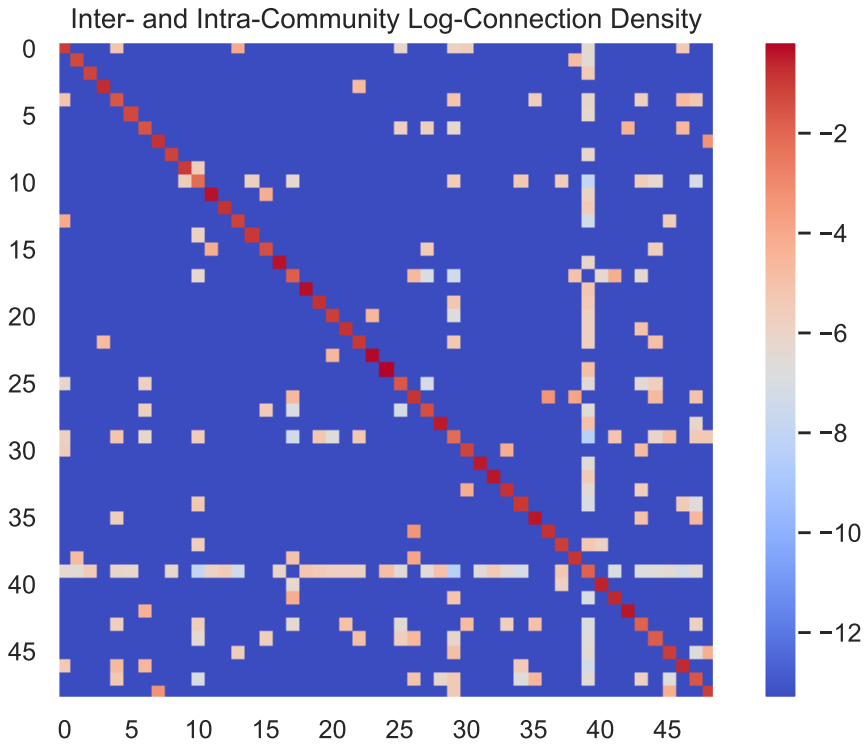

Figure 5.9. Intra- and inter-community connection densities. The diagonal of the matrix contains intra-cluster connection densities, every other matrix element is an inter-cluster density. Densities are computed as the ratio of existing and possible edges between each pair of vertices in the two clusters being considered. Coloring is based on the natural log scale.

6. Discussion and Future Work

6.1. Summary of the Results

Altogether, we can confirm that we have come up with an approach to metagenome analysis that is faster and more computationally lightweight than methods based on whole-genome reference databases, at the cost of reduced predictive performance.

Since classification of patients with IBD and IBS — diseases directly related to the microbial content of the lower gastrointestinal tract — already appears to be a hard problem, we expect that correctly detecting other kinds of diseases from metagenomic data will be even more challenging. Therefore, both the metagenome analysis pipeline and the subsequent machine learning models will need to be improved.

6.2. Improvements to the Software

Potential improvements to the metagenome pipeline are therefore threefold:

1. Fine-tuning heuristics by optimizing their parameters,
2. Speeding up manipulation of large data sets, and
3. Making use of better methodology in places where the current solution takes a shortcut.

The first group mainly entails running the pipeline, perhaps with the classification models attached, with various parametrizations of the underlying bioinformatic tools, such as `minimap2` or `HMMER3`, then choosing the parameters that correspond to the most biologically relevant outcome and/or the best classification performance. This has to be automated eventually, given the large search space spanned by all possible parameters.

It would also be valuable to develop a metric so as to precisely define what is

meant by "biologically relevant". An obvious but not necessarily optimal approach here would be to quantify the difference between the output of our model and that of an existing "gold standard" tool, when applied to a representative reference data set.

The most time-consuming part of metagenome analysis is usually read alignment. However, in our software, post-processing the output of the read aligners takes a comparable amount of time. In fact, when only running the quicker `minimap2` on read files of around 12–30 million pairs (1-2 GB gzipped) and skipping the profile HMM search step, post-processing usually takes noticeably *longer* than read mapping.

This can be owed to the fact that for reasons of velocity of development, simplicity, and correctness of the software, the developed algorithms are often naïve and sometimes a lot slower than what would likely be possible. In addition, we relied on the Pandas [37] library, which is heavily-optimized for the most part, but certain operations are still slow when applied to longer datasets. For instance, `groupby` and `join` function often ran for several minutes on our `DataFrame`s of approximately 500 rows and 6000 columns. However, such data manipulation could be replaced by less convenient but more performant manually-coded equivalents.

Lastly, the third opportunity for improvement is increasing the sensitivity of read alignment and abundance counting. A recurring theme during our entire work was the obvious sparsity of abundance data. This was apparently a real issue, since we have seen in the previous chapters that the best-performing models among the ones fitted on the output of our software were those that declared the least stringent criteria for read inclusion and counting, that is, the `placebo` datasets. We can infer from this fact that our functional genes pipeline has a high false negative rate when deciding if a read is mapped to a reference.

The origin of this problem is complex, but some of the design and methodological decisions are likely contributing factors. Concretely:

- Pseudorandom sampling is extensively used for making otherwise very large datasets tractable. As a reminder, the software draws samples from reference genes when building the database, and it also subsamples read files when aligning to HMM profiles. The latter is particularly dangerous practice – processing only 1, 5, or even 10% of all available reads can (and usually does) completely eliminate very low-abundance taxa or genes from the output. Therefore, more advanced alternatives to subsampling should be investigated.

Incidentally, the related technique of **rarefaction** is already deemed to be statistically unsound, as demonstrated in McMurdie and Holmes [38].

- Some of the conditions for accepting mapped reads might be unfounded and prohibitively strict, and that is inherent in our approach. To elaborate, reference sequences and HMM profiles are very often only a couple of hundreds or at most a couple of thousands of base pairs long. Plugging in some typical numbers for Illumina paired-end short-read sequencing, we might try to align reads of length $2 \cdot 150$ base pairs with an insert size of say, $50 \ldots 250$ to a reference gene of size 300.

  In this case, even if we succeed in mapping one of the reads to the reference, its pair is virtually guaranteed not to map or to map only partially, simply because there is physically not enough space for it to fit. Therefore, requiring that in paired-end mode all reads be aligned in proper pairs is likely unwarranted and results in many reads being thrown away incorrectly.

In addition to increasing the sensitivity, another methodological improvement would be to filter out host (human) DNA reads, as these can contribute noise if they contain sequences similar to microbial genes, either for evolutionary reasons or just by chance. The SHOGUN pipeline does perform this pre-filtering step automatically. We, however, omitted it, because originally, it wasn't going to be necessary.

Initially, our pipeline was to be run only on the proprietary data from the clients of Medipredict, and the sequencing vendor the company is currently working with filters out any non-microbial DNA fragments before prividing the final FASTQ files. However, the software was later executed on other, open-access datasets as well, as previously described. We unfortunately lacked the time to check how much human DNA, if any, these 3 open datasets contain.

6.3. Improvements to the Classification Models

We also identified aspects of the statistical methods applied that could be fruitful to explore further. These are more on the refinement side, rather than requiring fundamental, structural changes.

A first step would be to get to know the data and the behavior of the models in even more detail. This could be done by explaining the worst instances of misclassification, row-by-row, for example using the ELI5 [39] library. This would

in turn enable us to discover why the model is making its biggest mistakes, and perhaps point out problems with imputation or data preparation in general. Bad imputation is one of the primary causes of severe, outlying prediction errors, so this would also be an opportunity to try other imputation methods.

We could make use of a more principled metric of feature importance, too. The SHAP [40] package offers a game-theoretic index of feature importance, which can be computed exactly for tree-based models. It would be interesting to try tree-based models anyway, primarily the more advanced boosting algorithms, such as XGBoost [41] or CatBoost [42]. These are expected to perform better than simple logistic regression on high-dimensional data, provided that one manages to avoid overfitting, which they are more prone to.

Another actionable suggestion for dealing with overfitting is manual feature selection. We could use the all-vs-all correlation matrix of features for eliminating multicollinearity, while the correlations between features and the response would help us remove redundant columns that only contain noise.

After having improved the data, we could try and gain more insight into the inner workings of the models. Obviously, while area under the ROC curve and the confusion matrix are useful metrics, they don't provide a detailed description of the advantages and disadvantages of the model. In order to fill this gap, we should evaluate the models using a variety of metrics, such as:

- Matthews Correlation Coefficient, of which the strength is that corrects for class imbalance (a priori probabilities), so it is a good default for describing the goodness of a model if positives and negatives are deemed equally important.
- F1 score, which is a very popular metric, and thus other statisticians and machine learning experts will likely find it familiar and easy to compare against other models.
- Area under the Precision-Recall curve. This is yet another AUC index, however, in some contexts, precision and recall can be more informative than false positive rate and false negative rate.

Another obvious-sounding but in practice nontrivial thing to do is simply getting more computing power, and increasing the size of the data sets by one or two orders of magnitude. Going from 500 to 5000 or even 50 000 instances does sound feasible given enough time and sufficient hardware. This would aid a lot with training as

well as evaluation – in particular, if there is more data, we can use more instances for the hold-out set, and consequently, the generalization power of the models can be assessed more accurately.

6.4. Enhancements to Network Analysis Methods

Studying correlation networks is probably the hardest task to carry out correctly within the context of the present work. It then follows that it also presents the highest number of concrete opportunities for using more advanced techniques and mathematical models.

First of all, networks generated according to the procedure described in Chapter 5 may have many spurious edges not bearing any real biological meaning, however statistically significant the underlying correlations might be. Again, the reason is sparsity: the many zeros in the abundance tables appear, after imputation and log-transformation, as several repeated large numbers. By repeated we mean that they are equal within a row or column (depending on the exact transform applied). Rare co-occurring nonzero values can then drive correlation values disproportionately towards ± 1, even though the co-occurrence is only an artifact of noise (and thus heavily depends on the quality and parametrization of read alignment). This phenomenon is demonstrated in listing 6.1. It could be ameliorated using the same approaches that address the problem of sparsity in general, as previously mentioned.

Listing 6.1. Log-transformation of imputed sparse data results in overly confident estimates of correlation

```
>>> from scipy.stats import pearsonr, spearmanr
>>>
>>> x = np.log([0.010, 0.010, 0.010, 0.010, 2.000, 0.100])
>>> y = np.log([0.001, 0.001, 0.001, 0.001, 1.500, 0.010])
>>>
>>> 'Pearson R = {:.3f}, P = {:.6f}'.format(*pearsonr(x, y))
'Pearson R = 0.993, P = 0.000073'
>>> 'Spearman R = {:.3f}, P = {:.6f}'.format(*spearmanr(x, y))
'Spearman R = 1.000, P = 0.000000'
```

Another issue with our network datasets is discretization. We generated unweighted graphs by thresholding correlation coefficients and P-values. This was necessary because our goal was to apply several models and to run many different

kinds of statistics on the same dataset. And while almost all models work on undirected, unweighted graphs with positive edges, more general (weighted, directed, or signed) graphs present various degrees of challenges.

However, discretization of continuous variables inherently loses information, and so does the act of completely ignoring negative correlations. Consequently, we should aim for retaining the original correlation coefficients and P-values without thresholding, and rely on more advanced models which are able to deal with such more general networks.

A third aspect is about being more principled when comparing networks based on data of healthy and ill persons. It is of course useful to visualize and inspect the resulting graphs, but for an objective comparison, proper statistical testing and other quantitative methods should be applied. Some of the recently-developed metrics that could help us achieve this goal are DeltaCon [43], NetDifM [44], and Portrait Divergence [45].

7. Acknowledgments

I would like to thank my advisors, Prof. Silvio Tosatto and Dr. Damiano Piovesan, for accepting me as their student, for their professional help, kind patience, and availability.

Thanks to all the excellent Professors at the University for their often-needed flexibility, and for the humane, equal, collegial treatment of students.

A big Thank You to my girlfriend, Noémi Szebenszki, for her love and understanding, for all the meticulous spell checking, and for believing in me even when I didn't believe in myself.

I would like to express my utmost gratitude towards my entire family for their moral, financial, and spiritual support, and for their continued encouragement during my studies. Mom, Dad, and Sis: I love you all.

Many thanks to my friends, Kristóf Kőhalmy and Eszter Horák, for always inviting me over and consoling me when I was feeling overwhelmed and hopeless.

I would like to thank my friends: Martino De Nardi and Dávid Feiner, for the Italian translation of the abstract and for the editing of the French translation thereof, respectively.

I would also like to thank my newfound Italian friends: Francesco Grimaldi, Laura Iacovissi, Stefano Pascali, Vittoria Ricciuti, and many others, for helping me learn the wonderful Italian language and lifestyle in the best possible way.

Thanks to the insanely talented singers of the Corollario, and its director, Nunzio Borra, for accepting me into the Choir and regarding me as a friend from the very beginning. I will never forget singing with you.

I would like to thank my friend András Tóth who let me know about Medipredict, and for his never-ending valuable advice about biology and bioinformatics tools.

Last but not least, I would also like to thank the entire Medipredict team for creating a lighthearted, encouraging, yet highly professional environment. I warmly thank my mentor, Eszter Tóth, for her great ideas, helpful inputs, and the long, interesting discussions and brainstorming sessions.

Bibliography

[1] D. Knights, L. W. Parfrey, J. Zaneveld, C. Lozupone, and R. Knight, "Human-associated microbial signatures: examining their predictive value", *Cell host & microbe*, vol. 10, pp. 292–296, Oct. 2011. DOI: 10.1016/j.chom.2011.09.003. [Online]. Available: https://www.cell.com/action/showPdf?pii=S1931-3128(11)00287-3.

[2] R. Knight, A. Vrbanac, B. C. Taylor, A. Aksenov, C. Callewaert, J. Debelius, A. Gonzalez, T. Kosciolek, L.-I. McCall, D. McDonald, A. V. Melnik, J. T. Morton, J. Navas, R. A. Quinn, J. G. Sanders, A. D. Swafford, L. R. Thompson, A. Tripathi, Z. Z. Xu, J. R. Zaneveld, Q. Zhu, J. G. Caporaso, and P. C. Dorrestein, "Best practices for analysing microbiomes", *Microbiome*, vol. 16, pp. 410–422, Jul. 2018. DOI: 10.1038/s41579-018-0029-9.

[3] M. K. Gibson, K. J. Forsberg, and G. Dantas, "Improved annotation of antibiotic resistance functions reveals microbial resistomes cluster by ecology", *The ISME journal*, 2014. DOI: ISMEJ.2014.106. [Online]. Available: http://dantaslab.wustl.edu/Dantas_Pubs/2014_Gibson_Resfams_ISMEJ.pdf.

[4] S. Nayfach and K. S. Pollard, "Toward accurate and quantitative comparative metagenomics", *Cell*, vol. 166, no. 5, pp. 1103–1116, Aug. 2016. DOI: 0.1016/j.cell.2016.08.007.

[5] J. Gilbert, M. J. Blaser, J. G. Caporaso, J. Jansson, S. V. Lynch, and R. Knight, "Current understanding of the human microbiome", *Nat med.*, vol. 24, no. 4, pp. 392–400, Apr. 2018. DOI: 10.1038/nm.4517.

[6] J. Lloyd-Price, C. Arze, A. N. Ananthakrishnan, M. Schirmer, J. Avila-Pacheco, T. W. Poon, E. Andrews, N. J. Ajami, K. S. Bonham, C. J. Brislawn, D. Casero, H. Courtney, A. Gonzalez, T. G. Graeber, A. B. Hall, K. Lake, C. J. Landers, H. Mallick, D. R. Plichta, M. Prasad, G. Rahnavard, J. Sauk, D. Shungin, Y. Vázquez-Baeza, R. A. White, iBDMDB investigators, J. Braun, L. A. Denson, J. K. Jansson, R. Knight, S. Kugathasan, D. P. B. McGovern, J. F. Petrosino, T. S. Stappenbeck, H. S. Winter, C. B. Clish, E. A. Franzosa, H. Vlamakis, R. J. Xavier, and C. Huttenhower, "Multi-omics of the gut microbial ecosystem in inflammatory bowel diseases", *Nature*, vol. 569, pp. 655–662, May 2019. DOI: 10.1038/s41586-019-1237-9.

[7] E. A. Franzosa, A. Sirota-Madi, J. Avila-Pacheco, N. Fornelos, H. J. Haiser, S. Reinker, T. Vatanen, A. B. Hall, H. Mallick, L. J. McIver, J. S. Sauk, R. G. Wilson, B. W. Stevens, J. M. Scott, K. Pierce, A. A. Deik, K. Bullock1, F. Imhann, J. Porter, A. Zhernakova, J. Fu, R. K. Weersma, C. Wijmenga, C. B. Clish, H. Vlamakis, C. Huttenhower, and R. J. Xavier, "Gut microbiome structure and metabolic activity in inflammatory bowel disease", *Nat microbiol.*, vol. 4, no. 2, pp. 293–305, Feb. 2019. DOI: 10.1038/s41564-018-0306-4.

[8] T. Sharpton, S. Lyalina, J. Luong, J. Pham, E. M. Deal, C. Armour, C. Gaulke, S. Sanjabi, and K. S. Pollard, "Development of inflammatory bowel disease is linked to a longitudinal restructuring of the gut metagenome in mice", *Msystems*, vol. 2, no. 5, Oct. 2017. DOI: 10.1128/mSystems.00036-17.

[9] M. Kanehisa and S. Goto, "KEGG: Kyoto encyclopedia of genes and genomes", *Nucleic acids res.*, vol. 28, pp. 27–30, 2000. DOI: 10.1093/nar/28.1.27.

[10] T. Preston-Werner and P. Gedam, "TOML: Tom's obvious, minimal language", [Online]. Available: https://toml.io/.

[11] "Prokaryotic RefSeq genomes", [Online]. Available: https://www.ncbi.nlm.nih.gov/refseq/about/prokaryotes/#representative_genomes.

[12] "BLAST+ databases FTP directory", [Online]. Available: ftp://ftp.ncbi.nlm.nih.gov/blast/db/.

[13] F. Sievers, A. Wilm, D. Dineen, T. J. Gibson, K. Karplus, W. Li, R. Lopez, H. McWilliam, M. Remmert, J. Söding, J. D. Thompson, and D. G. Higgins, "Fast, scalable generation of high-quality protein multiple sequence alignments using Clustal Omega", *Mol syst biol.*, vol. 7, no. 539, 2011. DOI: 10.1038/msb.2011.75. [Online]. Available: https://www.embopress.org/doi/epdf/10.1038/msb.2011.75.

[14] W. S. J. Valdar, "Scoring residue conservation", *Proteins*, vol. 48, pp. 227–241, 2002.

[15] J. Canny, "A computational approach to edge detection", *IEEE transactions on pattern analysis and machine intelligence*, vol. PAMI-8, no. 6, pp. 679–698, Nov. 1986. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.420.3300&rep=rep1&type=pdf.

[16] S. R. Eddy, "Accelerated profile HMM searches", *PLOS computational biology*, vol. 7, no. 10, Oct. 2011. DOI: 10.1371/journal.pcbi.1002195. [Online]. Available: https://journals.plos.org/ploscompbiol/article/file?id=10.1371/journal.pcbi.1002195&type=printable.

[17] A. Tareen and J. B. Kinney, "Logomaker: beautiful sequence logos in Python", *Bioinformatics*, vol. 36, no. 7, pp. 2272–2274, Apr. 2020. DOI: 10.1093/bioinformatics/btz921. [Online]. Available: https://academic.oup.com/bioinformatics/article/36/7/2272/5671693.

[18] S. Seabold and J. Perktold, "Statsmodels: econometric and statistical modeling with Python", in *9th Python in science conference*, 2010.

[19] B. Hillmann, D. Knights, Q. Zhu, N. Bokulich, and G. Caporaso, "SHOGUN: accurate, scalable metagenomic quantification with shallow shotgun sequencing", DOI: 10.5281/zenodo.3901811. [Online]. Available: https://github.com/knights-lab/SHOGUN/.

[20] H. Li, "Minimap2: pairwise alignment for nucleotide sequences", *Bioinformatics*, vol. 34, no. 18, pp. 3094–3100, Sep. 2018. DOI: 10.1093/bioinformatics/bty191. [Online]. Available: https://academic.oup.com/bioinformatics/article/34/18/3094/4994778.

[21] ——, "Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM", *ArXiv*, vol. 1303, Mar. 2013. [Online]. Available: https://arxiv.org/abs/1303.3997.

[22] P. Skewes-Cox, T. J. Sharpton, K. S. Pollard, and J. L. DeRisi, "Profile hidden markov models for the detection of viruses within metagenomic sequence data", *PLOS One*, vol. 9, no. 8, Aug. 2014. DOI: 10.1371/journal.pone.0105067. [Online]. Available: https://journals.plos.org/plosone/article/file?id=10.1371/journal.pone.0105067&type=printable.

[23] Y. Zhang, Y. Sun, and J. R. Cole, "A scalable and accurate targeted gene assembly tool (SAT-Assembler) for next-generation sequencing data", *PLOS computational biology*, vol. 10, no. 8, Aug. 2014. DOI: 10.1371/journal.pcbi.1003737. [Online]. Available: https://journals.plos.org/ploscompbiol/article/file?id=10.1371/journal.pcbi.1003737&type=printable.

[24] P. A. Cock, T. Antao, J. T. Chang, B. A. Chapman, C. J. Cox, A. Dalke, I. Friedberg, T. Hamelryck, F. Kauff, B. Wilczynski, and M. J. L. de Hoon, "Biopython: freely available python tools for computational molecular biology and bioinformatic", *Bioinformatics*, vol. 25, pp. 1422–1423, 2009. DOI: 10.1093/bioinformatics/btp163.

[25] B. Langmead and S. Salzberg, "Fast gapped-read alignment with Bowtie 2", *Nature methods*, vol. 9, pp. 357–359, Mar. 2012. DOI: 10.1038/nmeth.1923. [Online]. Available: https://www.nature.com/articles/nmeth.1923.

[26] G. Al-Ghalith and D. Knights, "Burst enables optimal exhaustive dna alignment for big data", DOI: 10.5281/zenodo.806850. [Online]. Available: https://zenodo.org/record/3779009.

[27] D. Merkel, "Docker: lightweight Linux containers for consistent development and deployment", *Linux journal*, p. 2, Mar. 2014. DOI: 10.5555/2600239.2600241. [Online]. Available: https://dl.acm.org/doi/10.5555/2600239.2600241.

[28] H. B. Nielsen, M. Almeida, A. S. Juncker, S. Rasmussen, J. Li, S. Sunagawa, D. R. Plichta, L. Gautier, A. G. Pedersen, E. L. Chatelier, E. Pelletier, I. Bonde, T. Nielsen, C. Manichanh, M. Arumugam, J.-M. Batto, M. B. Q. dos Santos, N. Blom, N. Borruel, K. S. Burgdorf, F. Boumezbeur, F. Casellas, J. Doré, P. Dworzynski, F. Guarner, T. Hansen, F. Hildebrand, R. S. Kaas, S. Kennedy, K. Kristiansen, J. R. Kultima, P. Léonard, F. Levenez, O. Lund, B. Moumen, D. L. Paslier, N. Pons, O. Pedersen, E. Prifti, J. Qin, J. Raes, S. Sørensen, J. Tap, S. Tims, D. W. Ussery, T. Yamada, M. Consortium, P. Renault, T. Sicheritz-Ponten, P. Bork, J. Wang, S. Brunak, and S. D. Ehrlich, "Identification and assembly of genomes and genetic elements in complex metagenomic samples without using reference genomes", *Nature biotechnology*, vol. 32, no. 8, pp. 822–828, Jul. 2014. DOI: 10.1038/nbt.2939. [Online]. Available: https://www.nature.com/articles/nbt.2939.

[29] D. McDonald, E. Hyde, J. W. Debelius, J. T. Morton, A. Gonzalez, G. Ackermann, A. A. Aksenov, B. Behsaz, C. Brennan, Y. Chen, L. D. Goldasich, P. C. Dorrestein, R. R. Dunn, A. K. Fahimipour, J. Gaffney, J. A. Gilbert, G. Gogul, J. L. Green, P. Hugenholtz, G. Humphrey, C. Huttenhower, M. A. Jackson, S. Janssen, D. V. Jeste, L. Jiang, S. T. Kelley, D. Knights, T. Kosciolek, J. Ladau, J. Leach, C. Marotz, D. Meleshko, A. V. Melnik, J. L. Metcalf, H. Mohimani, E. Montassier, J. Navas-Molina, T. T. Nguyen, S. Peddada, P. Pevzner, K. S. Pollard, G. Rahnavard, A. Robbins-Pianka, N. Sangwan, J. Shorenstein, L. Smarr, S. J. Song, T. Spector, A. D. Swafford, V. G. Thackray, L. R. Thompson, A. Tripathi, Y. Vázquez-Baeza, A. Vrbanac, P. Wischmeyer, E. Wolfe, Q. Zhu, T. A. G. Consortium, and R. Knight, "American gut: an open platform for citizen science microbiome research", *mSystems*, May 2018. DOI: 10.1128/mSystems.00031-18. [Online]. Available: https://msystems.asm.org/content/msys/3/3/e00031-18.full.pdf.

[30] S. Wu, C. Sun, Y. Li, T. Wang, L. Jia, S. Lai, Y. Yang, P. Luo, D. Dai, Y.-Q. Yang, Q. Luo, N. L. Gao, K. Ning, L.-j. He, X.-M. Zhao, and W.-H. Chen, "GMrepo: a database of curated and consistently annotated human gut metagenomes", *Nucleic acids research*, vol. 48, no. D1, pp. D545–D553, Jan.

2020. DOI: 10.1093/nar/gkz764. [Online]. Available: https://academic.oup.com/nar/article/48/D1/D545/5559685.

[31] J. A. Martín-Fernández, C. Barceló-Vidal, and V. Pawlowsky-Glahn, "Dealing with zeros and missing values in compositional data sets using nonparametric imputation", *Mathematical geology*, vol. 35, no. 3, pp. 253–278, Apr. 2003.

[32] The scikit-bio development team, *Scikit-bio: a bioinformatics library for data scientists, students, and developers*, version 0.5.6, 2020. [Online]. Available: http://scikit-bio.org.

[33] 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.

[34] T. Head, MechCoder, G. Louppe, I. Shcherbatyi, fcharras, Z. Vinícius, cmmalone, C. Schröder, nel215, N. Campos, T. Young, S. Cereda, T. Fan, rene-rex, K. Shi, J. Schwabedal, carlosdanielcsantos, Hvass-Labs, M. Pak, SoManyUsernamesTaken, F. Callaway, L. Estève, L. Besson, M. Cherti, K. Pfannschmidt, F. Linzberger, C. Cauet, A. Gut, A. Mueller, and A. Fabisch, *Scikit-optimize*, version 0.7.4. DOI: 10.5281/zenodo/1207017. [Online]. Available: https://zenodo.org/record/1207017.

[35] W. Long and Q. Wang, "Two methods of correlation coefficient on compositional data", *Sciencedirect*, vol. 18, pp. 1757–1763, 2013. DOI: 10.1016/j.procs.2013.05.344.

[36] A. L. Barabási and M. Pósfai, *Network science*. Cambridge University Press, 2016, ISBN: 9781107076266 1107076269. [Online]. Available: http://barabasi.com/networksciencebook/.

[37] The pandas development team, *Pandas-dev/pandas: Pandas*, version 1.0.3, Feb. 2020. DOI: 10.5281/zenodo.3509134. [Online]. Available: https://doi.org/10.5281/zenodo.3509134.

[38] P. J. McMurdie and S. Holmes, "Waste not, want not: why rarefying microbiome data is inadmissible", *Plos computational biology*, vol. 10, no. 4, e1003531, Apr. 2014. DOI: 10.1371/journal.pcbi.1003531.

[39] The ELI5 developers, *ELI5*. [Online]. Available: https://eli5.readthedocs.io/.

[40] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions", in *Advances in neural information processing systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., Curran Associates, Inc., 2017, pp. 4765–4774. [Online]. Available: http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf.

[41] T. Chen and C. Guestrin, "XGBoost: a scalable tree boosting system", in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, ser. KDD '16, San Francisco, California, USA: ACM, 2016, pp. 785–794, ISBN: 978-1-4503-4232-2. DOI: 10.1145/2939672.2939785. [Online]. Available: http://doi.acm.org/10.1145/2939672.2939785.

[42] A. V. Dorogush, V. Ershov, and A. Gulin, "CatBoost: gradient boosting with categorical features support", *Workshop on ML systems at NIPS 2017*, 2017. [Online]. Available: http://learningsys.org/nips17/assets/papers/paper_11.pdf.

[43] D. Koutra, N. Shah, J. T. Vogelstein, B. Gallagher, and C. Faloutsos, "DeltaCon: principled massive-graph similarity function with attribution", *ACM trans. knowl. discov. data*, vol. 10, no. 3, Feb. 2016. DOI: 10.1145/2824443.

[44] J. Ji, Z. Yuan, X. Zhang, and F. Xue, "A powerful score-based statistical test for group difference in weighted biological networks", *BMC bioinformatics*, vol. 17, no. 86, Feb. 2016. DOI: 10.1186/s12859-016-0916-x. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4751708/pdf/12859_2016_Article_916.pdf.

[45] J. P. Bagrow and E. M. Bollt, "An information-theoretic, all-scales approach to comparing networks", *ArXiv*, Jul. 2019.

Appendix A.

List of annexes

- LaTeX sources, supplementary files, and a compiled PDF in the `LaTeX` directory;