Improving Machine Learning Models for Microbiome Analysis and Democratizing Data Science Along the Way

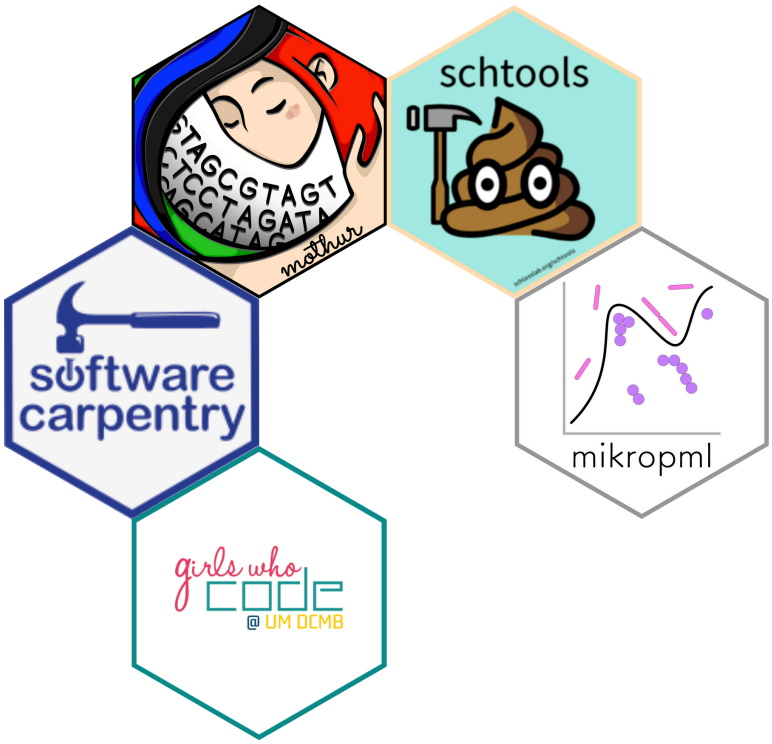
by

Kelly L. Sovacool

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Bioinformatics)
in the University of Michigan
2023

Doctoral Committee:

Professor Patrick D. Schloss, Chair
Associate Professor Gregory J. Dick
Associate Professor Peter L. Freddolino
Associate Professor Jenna Wiens
Professor Vincent B. Young


mothur
schtools
software carpentry
mikropml
girls who code
@ UM DCMB

Kelly L. Sovacool
sovacool@umich.edu
ORCID iD: 0000-0003-3283-829X

DEDICATION

To my parents. I am forever grateful for all you do for me.

ACKNOWLEDGEMENTS

Thank you to the Department of Computational Medicine & Bioinformatics for funding my research and training via the NIH Training Program in Bioinformatics (T32 GM070449).

Thank you to all the people who supported me in this journey and helped shape me to become who I am:

My family: Mom, Dad, Erika, and Katie, and my partner David. Your support means everything to me. My extended family for not disowning me for attending the University of Michigan despite your other allegiances. All my friends from this chapter of life and the chapters preceding it.

Kristi Janson, my high school teacher who taught the biomedical sciences courses that made me want to become a scientist. Rachelle Sells Galvin and the scientists at Eli Lilly & Company who graciously organized a day for me to shadow them, where they shared their passion for science and gave me great advice for pursuing research experiences in college.

My undergraduate research mentors at the University of Kentucky: Jerzy Jaromczyk, Neil Moore, Dave Weisrock, Justin Kratovil, and Hunter Moseley. Hunter Moseley spent many hours critiquing my code and cultivated my programming skills from crawling to running.

All those involved in The Carpentries instance at the University of Michigan who contributed to the curriculum, taught in the pilot workshops, and continue to empower others to learn how to code.

The Graduate Employees Organization at the University of Michigan (AFT Local-3550), for bargaining for affordability and dignity for all graduate workers.

Past and present members of the Executive Committee of our chapter of Girls Who Code at U-M DCMB, especially Brooke Wolford and Zena Lapp, who co-founded the chapter; Marlena Duda, who spearheaded the effort to develop our custom curriculum; and Audrey Drotos and Hayley Falk, who are passing on the torch to the next generation of graduate students to sustain this club for many years to come. All those who contributed to the curriculum and continue to support the club. DCMB, for unwaivering support of the club since the beginning, especially our faculty co-sponsors Maureen Sartor and Cristina Mitrea. Thanks to all the students who participated and tried something new.

My committee members Greg Dick, Peter Freddolino, Jenna Wiens, and Vince Young for your helpful insights. Krishna Rao for helping me understand clinical guidelines for diagnosing and treating *C. difficile* infections.

Members of the Schloss Lab past and present, with special thanks to my undergraduate mentee, Megan Coden. You are all wonderful people and you made graduate school a great experience.

My advisor and mentor, Pat. You are the best!

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE

LIST OF TABLES

TABLE

LIST OF ACRONYMS

ARR Absolute risk reduction

AUBPRC Area under the balanced precision-recall curve

AUPRC Area under the precision-recall curve

AUROC Area under the receiver-operator characteristic curve

CDI *Clostridioides difficile* infection

CI Confidence interval

EHR Electronic health record

ICU Intensive Care Unit

IDSA Infectious Diseases Society of America

MCC Matthews Correlation Coefficient

ML Machine learning

NNB Number needed to benefit

NNS Number needed to screen

NNT Number needed to treat

OTU Operational Taxonomic Unit

RAM Random access memory

rRNA ribosomal Ribonucleic Acid

RDP Ribosomal Database Project

STEM Science, Technology, Engineering, and Mathematics

ABSTRACT

The human microbiome plays an important role in maintaining health. Changes in the taxonomic and functional composition of the gut microbiota have been implicated in numerous diseases including colorectal cancer, *Clostridioides difficile* infection (CDI), and others. Thus, the gut microbiome is a promising source of biomarkers for disease diagnosis and prediction. Machine learning (ML) approaches can leverage large datasets to gain insights into associations between the microbiota and disease. Here, we present a new algorithm that improves microbiome analysis for ML applications, apply ML to predict severity of CDI, and introduce resources that empower data scientists to go from the basics of coding to applying ML for reproducible research.

Assigning amplicon sequences to operational taxonomic units (OTUs) is an important step in characterizing microbial communities across large datasets. However, a gap in existing OTU assignment methods inhibited the ability of researchers to incorporate new samples to previously clustered datasets, such as when deploying ML models. To provide an efficient method to fit sequences to existing OTUs while maintaining high OTU quality, we developed the OptiFit algorithm, an improved implementation of reference-based clustering. Our benchmarks revealed that OptiFit produces similar quality OTUs as a gold standard method yet at faster speeds. Thus, OptiFit provides a suitable option for users requiring consistent and high quality OTU assignments for ML applications and beyond.

CDI can lead to severe complications including death, with half a million cases annually in the United States. The composition of the gut microbiome plays an important role in determining colonization resistance and clearance upon exposure to *C. difficile*. We investigated whether ML models trained on OTUs from stool samples on the day of CDI diagnosis could predict which cases led to severe outcomes. We trained models to predict CDI severity for four different severity definitions. The models performed best when predicting pragmatic severity, a composite definition of complications due to any cause or confirmed as CDI-attributable via chart review when possible. Our results suggest that while chart review is valuable to verify the cause of complications, including as many samples as possible is indispensable for training performant models on imbalanced datasets. We evaluated the potential clinical value of these models and found similar performance compared to prior

models based on electronic health records, although further work is needed to determine the feasibility of deploying such models in clinical practice. These results represent a step toward the goal of deploying ML to inform clinical decisions and ultimately improve CDI outcomes.

Bioinformatics is a kind of data science, an interdisciplinary field integrating computer science, statistics, and domain knowledge. Novice researchers frequently have domain knowledge, but lack other skills necessary to apply data science to their datasets while adhering to best practices in reproducibility. We developed three resources to help democratize data science: a curriculum teaching the basics of Python for data science to young students, a curriculum teaching programming skills for reproducible research, and an R package implementing an ML framework to help novices apply ML responsibly while being customizable for advanced users. These contributions cover a breadth of audience skill levels to help fill gaps in existing resources for data science. In summary, this dissertation advances bioinformatics for microbiome research from the start of data analysis through application, and ultimately toward enabling others to reproduce and extend our work.

CHAPTER 1

Introduction

1.1 Microbial communities in human health

Microbial communities are assemblages of microorganisms – archaea, bacteria, fungi, protists, and viruses – that inhabit a local environment [1]. A microbiome consists of the microbial community in its environment together with the molecules they produce such as nucleic acids, proteins, lipids, metabolites, and more [2]. Microbiomes are thus tightly associated with the local environment they occupy such as waterways, soil layers, ocean floors, plants, insects, and animals. The human body hosts microbes that inhabit the skin, mouth, gut, vagina, airway, and other body regions [3]. The living members of a microbiome or referred to as the microbiota, and they interact with each other cooperatively and competitively [4] and can influence the health of the host directly or indirectly [5]. The composition of a microbiome can be characterized according to the taxonomy of its microbiota, the functions they carry out, and the metabolites they produce.

High throughput sequencing and other 'omics techniques can be used to characterize the metagenomes, metatranscriptomes, metaproteomes, and meta-metabolomes of microbiomes and describe how they change over time, in response to changing environments, or between healthy and diseased states of the host. A benefit of 'omics techniques is they do not require microbes to be cultured in a laboratory, making it possible to observe genes, gene products, and metabolites from microbes missed by culturing techniques and even discover new species [6]. These large, multivariate datasets present a challenge for bioinformatic analysis, as greater computational resources and more sophisticated statistical techniques are required to process and analyze big data [3]. An older but still widely-used method to profile the taxonomic composition of microbial communities is amplicon sequencing. In amplicon sequencing, a region of a marker gene is selected depending on which domain of microbial life is being targeted, and the DNA sequences matching that region are amplified and sequenced [7, 8]. Relative to shotgun metagenomics, amplicon sequencing significantly reduces the

costs and computational resources needed to characterize the taxonomic composition of microbiomes. Amplicon sequence data can typically be identified at the genus level, while shotgun metagenomics data provide a finer taxonomic resolution at the species or strain level [9]. When researchers need to extract sequences from many samples and only require taxonomic resolution at the genus level, amplicon sequencing is a practical choice.

1.1.1 *Clostridioides difficile* infection

The healthy gut microbiome is resistant to colonization and infection from pathogens due to competition from beneficial microbes. Medications such as antibiotics, proton-pump inhibitors, and osmotic laxatives can disrupt the taxonomic and functional composition of the gut microbiome, thereby allowing pathogens to gain a foothold [10, 11]. *C. difficile* is classically considered a hospital-acquired pathogen that infects patients who are taking antibiotics for other illnesses, and especially in elderly patients as the immune system weakens with age. However, community-acquired *C. difficile* infection (CDI) is increasing, even in patients with no recent history of antibiotic use [12]. One mechanism through which the healthy gut community protects against CDI is bile acid metabolism, which can inhibit the growth and alter toxin production of *C. difficile* [13, 14, 15]. Mouse studies have shown that the initial taxonomic composition of the gut microbiome can influence *C. difficile* clearance, host moribundity, and cecal tissue damage in infected mice [16, 17, 18]. Differences between resilient versus susceptible microbiomes could be used as biomarkers to identify patients at risk of being infected or developing severe complications.

Clinical outcomes of CDI can be severe, as a small portion of patients experience complications requiring ICU admission due to CDI such as ileus, toxic megacolon, or death in 8-9% of cases [19, 20]. Colectomy is used as a last-resort treatment to prevent death when other treatments fail, and the mortality rate in patients who undergo colectomy for CDI is approximately 35% [21]. CDI is of particular cause for concern due to the risk of recurrence, where a patient experiences another CDI within 2-8 weeks of a prior CDI [22]. It is thought that the antibiotics prescribed to treat CDI may also prevent the microbiome from recovering, thereby perpetuating a cycle of perturbance and *C. difficile* proliferation. Adjuvant therapies such as bezlotoxumab and fecal microbiota transplant have recently been introduced to help break the cycle of recurrence by targeting the toxins produced by *C. difficile* or restoring the microbiota to a healthy state, but they are not yet used widely and typically only in patients experiencing a first or second recurrence [23]. Furthermore, vancomycin is commonly used to treat first cases of CDI, but vancomycin-resistant *Enterococcus* is becoming more common and enterococci have been shown to exacerbate the pathogenesis of *C. difficile* [24, 25].

There is a great need for improved therapies to prevent recurrent and severe CDI, as well as for tools to predict which patients are at risk so that clinicians can adjust treatment plans to prevent adverse outcomes from occurring.

1.2 Machine learning for science and health care

Machine learning techniques applied to large datasets have transformed the quantitative sciences towards a data-driven paradigm. Supervised ML approaches can be used to classify samples or make predictions, and researchers often claim that the good discriminative performance of a model supports the veracity of an underlying scientific claim [26]. Alternatively, ML can be used pragmatically in fields like medicine in order to aid in diagnosing diseases or to make predictions about disease outcomes, with the goal of improving health care. Models based on clinical laboratory tests and electronic health record (EHR) data and have been trained to predict deterioration in COVID-19 patients [27], identify patients at risk of being infected with *C. difficile* in ICU wards [28], and predict outcomes in CDI patients [29]. ML models trained on microbiota have been used to improve detection of colorectal cancer [30], distinguish CDI patients from diarrheal controls [31], and identify which members of the microbiota contribute most to the performance of these models [32].

While ML techniques hold great promise to improve health care, caution must be taken to train, test, validate, and deploy ML models responsibly. Pervasive pitfalls have been identified in studies applying ML; these include data leakage between the training and test/validation set, failing to set a random seed, biased training data, inappropriate choice of performance metric, not reporting variation in performance, and neglecting to document methods in sufficient detail [26, 32]. Errors in ML can invalidate the conclusions of a study and erode trust in the scientific endeavor. Even worse, errors can be dangerous when ML is applied to health care, as diagnoses and prognoses assisted by ML can directly affect patients [33]. Before a model can ever be deployed in clinical practice, it must be rigorously evaluated to ensure it avoids these errors, and ultimately that it will improve rather than worsen patient outcomes [34]. ML practitioners must take care to avoid technical pitfalls and engage with experts from interdisciplinary fields to ensure their models will be useful and beneficial for clinical practice.

1.3 Democratizing reproducible data science

Data science is an interdisciplinary field that integrates computer science, statistics, and expertise from a problem domain. When the problem domain is biology, the field could

be referred to as biological data science, computational biology, or bioinformatics, although there is no consensus definition for any of these terms. As costs decrease for generating large datasets such as those from high-throughput sequencing experiments, there is an ever-growing need for data science practitioners with the skills and knowledge to process the data, make inferences, and communicate their findings. Democratizing data science means making the theory, methods, and tools, more accessible by creating educational resources, user-friendly software tools, or even simply making data publicly available. Accessibility is important for filling the growing demand for skilled data scientists across sectors as well as to improve diversity in the field [35, 36, 37, 38]. Non-profit organizations have been founded to help address the diversity gap in computer science, data science, and other STEM fields including Girls Who Code, Women in Science and Engineering, Association for Women in Science, Society for Advancement of Chicanos and Native Americans in Science, and many others.

An important attribute of any scientific finding is reproducibility, where others can repeat the same methods on the original dataset to obtain the same result [39]. Reproducibility does not guarantee correctness, replicability, nor generalizability, but it is a minimal achievable standard that helps others evaluate scientific claims [40]. A finding could be entirely unreproducible, where the data are not shared and the analysis methods are not described in sufficient detail. Achieving perfect reproduciblity is unlikely as eventually link rot, software bugs, and shuttering of organizations can occur, but "good enough" practices are attainable [41]. Aside from enabling others to validate or build upon one's work, reproducible practices make researchers more productive both collaboratively and individually [41]. As a reproducibility crisis has been identified in virtually all scientific fields including microbiology, bioinformatics, and ML for health research, promoting and encouraging reproducible practices is important to re-establish trust and trustworthiness in scientific findings [42, 39, 26, 43]. However, many early-career researchers lack the quantitative and computational skills and self-confidence necessary to perform reproducible computational science, in some cases due to prior demotivating experiences [44]. Toward the goal of disseminating reproducible research practices, Software Carpentry, Data Carpentry, and Library Carpentry (together under the umbrella term "The Carpentries") have developed extensive educational materials and taught them in hands-on workshops worldwide to researchers, scientists, librarians, and other data wranglers [45]. A particularly important contribution of The Carpentries is the instructor training course which promotes evidence-based pedagogical practices that motivate and empower learners, such as instruction via participatory live-coding [46, 47]. Improving access to educational resources for best practices in coding and data science will equip budding scientists with the skills necessary to conduct and communicate their work

reproducibly.

1.4 Dissertation outline and contributions

In the preambles of Chapters 2 through 4, I note my specific contributions to the work described in each chapter. Chapter 2 presents OptiFit, a new OTU clustering algorithm that enables researchers to fit new data to existing *de novo* OTUs while maintaining OTU quality. Chapter 3 presents findings from training ML models to predict the severity of CDI from OTUs and comparing model performance to prior approaches. Chapter 4 introduces two curricula and one software package which help democratize data science for a range of audiences. In Chapter 5, I discuss the impacts of the findings presented in Chapters 2 through 4 and propose future work to build upon this dissertation.

1.5 Datasets used in this dissertation

In Chapter 2, we re-used previously published 16S rRNA gene amplicon sequence data extracted from four different communities: soil, marine, mouse gut, and human gut. Using multiple datasets from disparate sources allowed us to demonstrate the suitability of OptiFit for microbiome researchers and microbial ecologists with diverse scientific interests. In Chapter 3, we used a dataset of 16S rRNA gene amplicon sequences extracted from 1,277 stool samples collected on the day of diagnosis from CDI patients at the University of Michigan. White blood cell counts and creatinine levels were also collected on the day of diagnosis in order to calculate IDSA severity scores. The occurrence of ICU admission, colectomy, or death within 30 days was recorded and in some cases, physicians conducted chart review to determine whether the complication was attributable to the CDI. In Chapter 4, we used results from surveys of learners who participated in the Girls Who Code club and Carpentries workshop where we piloted our new curricula, which allowed us to measure the success of our teaching approaches. The data are described in further detail in each of their respective chapters.

CHAPTER 2

OptiFit: an Improved Method for Fitting Amplicon Sequences to Existing OTUs

2.1 Preamble

This chapter introduces a novel algorithm, OptiFit, for performing reference-based clustering of amplicon sequences into Operational Taxonomic Units. We showed that OptiFit produces OTUs at a similar quality as other clustering methods while enabling new sequences to be clustered to existing *de novo* OTUs, which was not previously possible. OptiFit can be used with OTU-based machine learning models to make predictions on new data, which we later demonstrated in a follow-up analysis [48].

I performed all of the analysis and created the figures for this chapter. Other co-authors conceived of and implemented the OptiFit algorithm and contributed analysis code. This paper was originally published in 2022 in mSphere with the following co-authors: Kelly L. Sovacool, Sarah L. Westcott, M. Brodie Mumphrey, Gabrielle A. Doston, and Patrick D. Schloss [49].

2.1.1 Importance

Advancements in DNA sequencing technology have allowed researchers to affordably generate millions of sequence reads from microorganisms in diverse environments. Efficient and robust software tools are needed to assign microbial sequences into taxonomic groups for characterization and comparison of communities. The OptiClust algorithm produces high quality groups by comparing sequences to each other, but the assignments can change when new sequences are added to a dataset, making it difficult to compare different studies. Other approaches assign sequences to groups by comparing them to sequences in a reference database to produce consistent assignments, but the quality of the groups produced is reduced compared to OptiClust. We developed OptiFit, a new reference-based algorithm that

produces consistent yet high quality assignments like OptiClust. OptiFit allows researchers to compare microbial communities across different studies or add new data to existing studies without sacrificing the quality of the group assignments.

2.2 Introduction

Amplicon sequencing is a mainstay of microbial ecology. Researchers can affordably generate millions of sequences to characterize the composition of hundreds of samples from microbial communities without the need for culturing. In many analysis pipelines, 16S rRNA gene sequences are assigned to operational taxonomic units (OTUs) to facilitate comparison of taxonomic composition between communities to avoid the need for taxonomic classification. A distance threshold of 3% (or sequence similarity of 97%) is commonly used to cluster sequences into OTUs based on pairwise comparisons of the sequences within the dataset. The method chosen for clustering affects the quality of OTU assignments and thus may impact downstream analyses of community composition [50, 51]. OTU quality can be conceptualized as how well the OTU assignments match the definition set by the distance threshold, i.e. whether sequence pairs that are at least as similar as the distance threshold are assigned to the same OTU and sequence pairs that are more dissimilar than the distance threshold are assigned to different OTUs.

There are two main categories of OTU clustering algorithms: *de novo* and reference-based. OptiClust is a *de novo* clustering algorithm which uses the distance score between all pairs of sequences in the dataset to cluster them into OTUs by maximizing the Matthews Correlation Coefficient (MCC) [50]. This approach takes into account the distances between all pairs of sequences when assigning query sequences to OTUs, in contrast to other *de novo* methods such as the greedy clustering algorithms implemented in USEARCH and VSEARCH [52, 53]. In methods employing greedy clustering algorithms, only the distance between each sequence and a representative centroid sequence in the OTU is considered while clustering. As a result, distances between pairs of sequences in the same OTU are frequently larger than the specified threshold, i.e. they are false positives. In contrast, the OptiClust algorithm takes into account the distance between all pairs of sequences when considering how to cluster sequences into OTUs and is thus less willing to take on false positives.

A limitation of *de novo* clustering is that different OTU assignments will be produced when new sequences are added to a dataset, making it difficult to use *de novo* clustering to compare OTUs between different studies. Furthermore, since *de novo* clustering requires calculating and comparing distances between all sequences in a dataset, the execution time can be slow and memory requirements can be prohibitive for very large datasets. Reference

clustering attempts to overcome the limitations of *de novo* clustering methods by using a representative set of sequences from a database, with each reference sequence seeding an OTU. Commonly, the Greengenes set of representative full length sequences clustered at 97% similarity is used as the reference with VSEARCH [53, 54]. Query sequences are then clustered into OTUs based on their similarity to the reference sequences. Any query sequences that are not within the distance threshold to any of the reference sequences are either thrown out (closed reference clustering) or clustered *de novo* to create additional OTUs (open reference clustering). While reference-based clustering is generally fast, it is limited by the diversity of the reference database. Novel sequences in the sample will be lost in closed reference mode if they are not represented by a similar sequence in the database. We previously found that the OptiClust *de novo* clustering algorithm created the highest quality OTU assignments of all clustering methods [50].

To overcome the limitations of current reference-based and *de novo* clustering algorithms while maintaining OTU quality, we developed OptiFit, a reference-based clustering algorithm. While other tools represent reference OTUs with a single sequence, OptiFit uses all sequences in existing OTUs as the reference and fits new sequences to those reference OTUs. In contrast to other tools, OptiFit considers all pairwise distance scores between reference and query sequences when assigning sequences to OTUs in order to produce OTUs of the highest possible quality. Here, we tested the OptiFit algorithm with the reference as a public database (e.g. Greengenes) or *de novo* OTUs generated using a reference set from the full dataset and compared the performance to existing tools. To evaluate the OptiFit algorithm and compare to existing methods, we used four published datasets isolated from soil [55], marine [56], mouse gut [57], and human gut [30] samples. OptiFit is available within the mothur software program.

2.3 Results

2.3.1 The OptiFit algorithm

OptiFit leverages the method employed by OptiClust of iteratively assigning sequences to OTUs to produce the highest quality OTUs possible, and extends this method for reference-based clustering. OptiClust first seeds each sequence into its own OTU as a singleton. Then for each sequence, OptiClust considers whether the sequence should move to a different OTU or remain in its current OTU, choosing the option that results in a better MCC score [50]. The MCC uses all values from a confusion matrix and ranges from negative one to one, with a score of one occurring when all sequence pairs are true positives and true negatives, a score

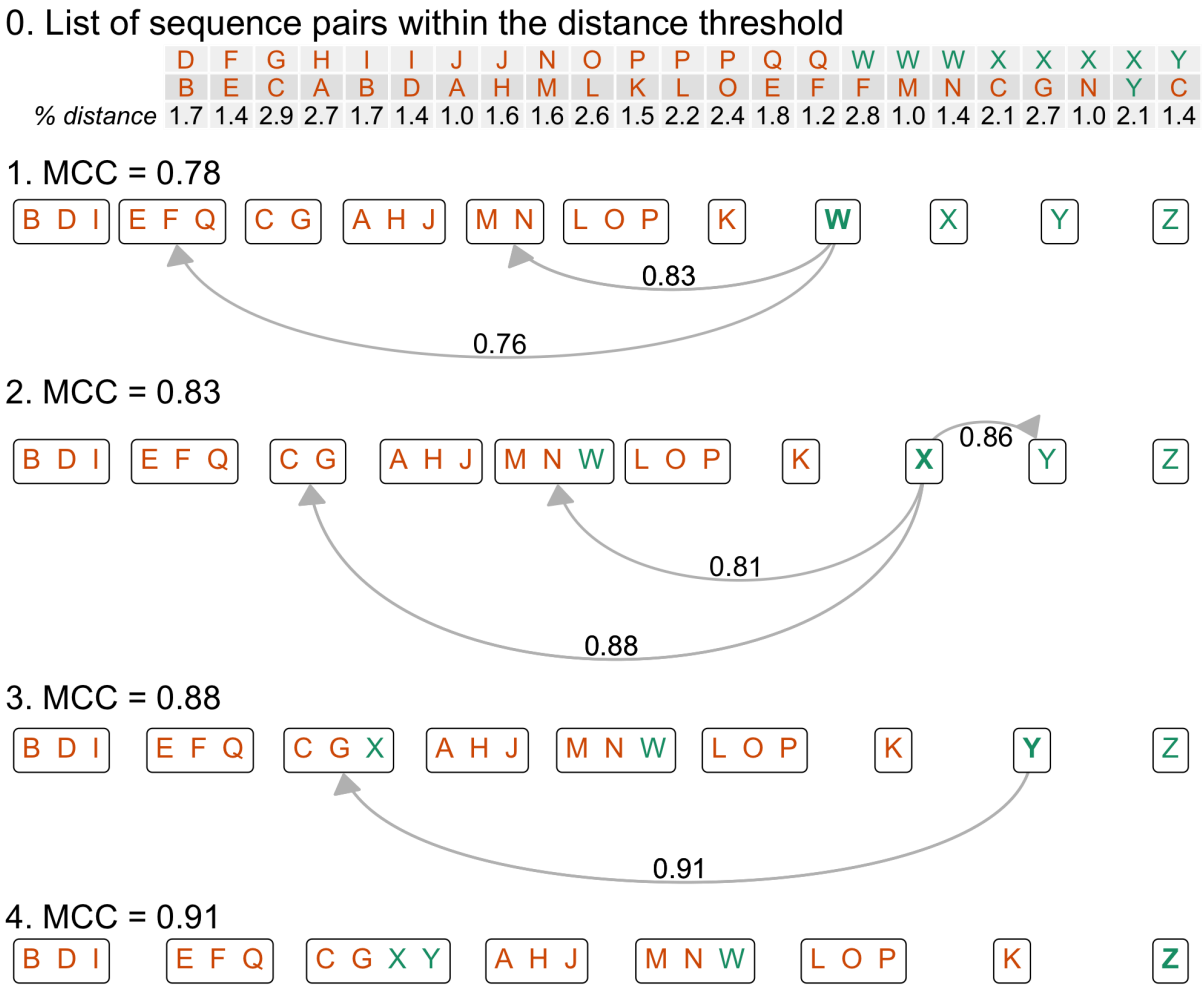
of negative one occurring when all pairs are false positives and false negatives, and a score of zero when there are equal numbers of true and false assignments (i.e. no better than random guessing). Sequence pairs that are similar to each other (i.e. within the distance threshold) are counted as true positives if they are clustered into the same OTU, and false negatives if they are not in the the same OTU. Sequence pairs that are not similar to each other are true negatives if they are not clustered into the same OTU, and false positives if they are in the same OTU. Thus, a pair of sequences is considered correctly assigned when their OTU assignment matches the OTU definition set by the distance threshold. OptiClust iterations continue until the MCC stabilizes or until a maximum number of iterations is reached. This process produces *de novo* OTU assignments with the most optimal MCC given the input sequences.

OptiFit begins where OptiClust ends, starting with a list of reference OTUs and their sequences, a list of query sequences to cluster to the reference OTUs, and the sequence pairs that are within the distance threshold (e.g. 0.03) (Figure 2.1). Initially, all query sequences are placed into separate OTUs. Then, the algorithm iteratively reassigns the query sequences to the reference OTUs to optimize the MCC. Alternatively, a sequence will remain unassigned if the MCC value is maximized when the sequence is a singleton rather than clustered into a reference OTU. All query and reference sequence pairs are considered when calculating the MCC. This process is repeated until the MCC changes by no more than 0.0001 (default) or until a maximum number of iterations is reached (default: 100). In the closed reference mode, any query sequences that cannot be clustered into reference OTUs are discarded, and the results only contain OTUs that exist in the original reference. In the open reference mode, unassigned query sequences are clustered *de novo* using OptiClust to generate new OTUs. The final MCC is reported with the best OTU assignments. There are two strategies for generating OTUs with OptiFit: 1) cluster the query sequences to reference OTUs generated by *de novo* clustering an independent database, or 2) split the dataset into a reference and query fraction, cluster the reference sequences *de novo*, then cluster the query sequences to the reference OTUs.

2.3.2 Reference clustering with public databases

To test how OptiFit performs for reference-based clustering, we clustered each dataset to three databases of reference OTUs: the Greengenes database v13_8_99 [58], the SILVA non-redundant database v132 [59], and the Ribosomal Database Project (RDP) v16 [60]. Reference OTUs for each database were created by performing *de novo* clustering with OptiClust at a distance threshold of 3% using the V4 region of each sequence (see Figure 2.2).

Figure 2.1: The OptiFit algorithm



Here we present a toy example of the OptiFit algorithm fitting query sequences to existing OTUs, given the list of all sequence pairs that are within the distance threshold of 3%. Previously, 50 reference sequences were clustered *de novo* with OptiClust (see the OptiClust supplemental text [50]). Reference sequences A through Q (colored) were within the distance threshold to at least one other reference sequence; the remaining reference sequences formed additional singleton OTUs (not shown). The goal of OptiFit is to assign the query sequences W through Z (colored) to the reference OTUs. Here, there are 50 reference sequences and 4 query sequences which make 1,431 sequence pairs, of which 23 pairs are within the 3% distance threshold. Initially (step 1), OptiFit places each query sequence in its own OTU, resulting in 14 true positives, 9 false negatives, 0 false positives, and 1,408 true negatives for an MCC score of 0.78. Then, for each query sequence (), OptiFit determines what the new MCC score would be if that sequence were moved to one of the OTUs containing at least one other similar sequence (steps 2-4). The sequence is then moved to the OTU which would result in the best MCC score. OptiFit stops iterating over sequences once the MCC score stabilizes. In this example, only one iteration over each sequence was needed. Note that sequence Z was dissimilar from all other sequences and thus it remained a singleton. The final MCC score is 0.91 with 20 true positives, 3 false negatives, 1 false positive, and 1407 true negatives.

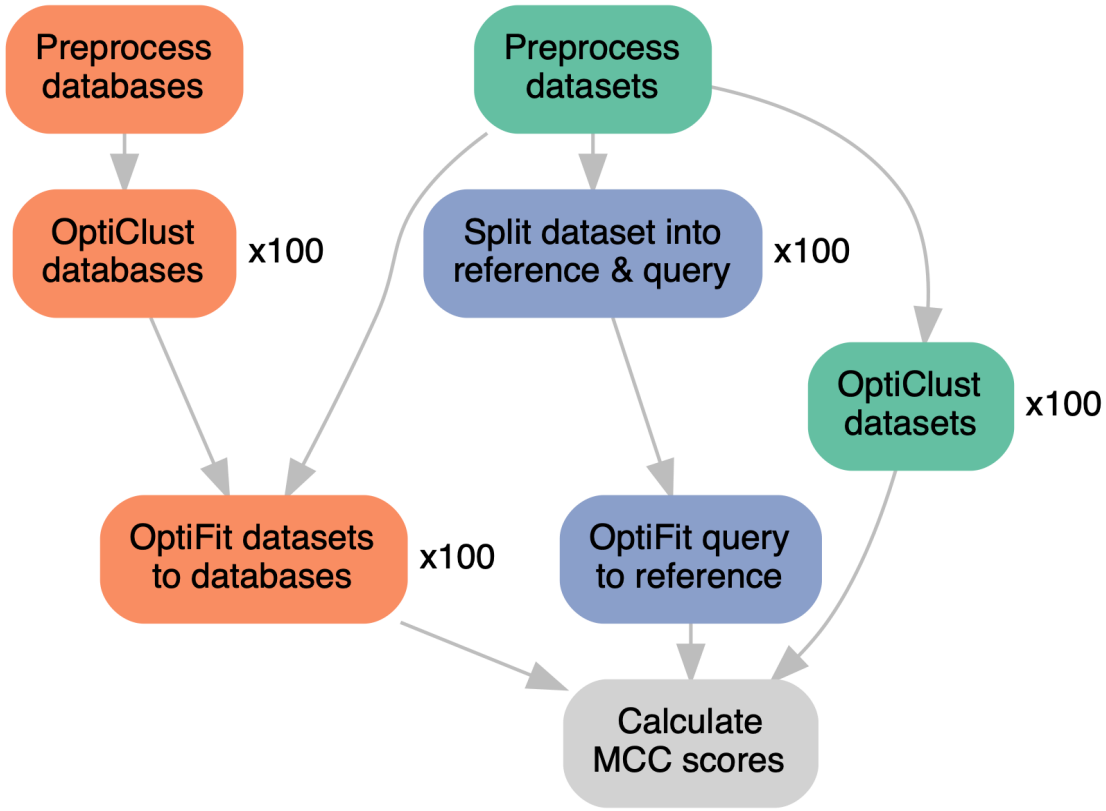
After trimming to the V4 region, the databases contained 174,979, 16,192, and 173,648 unique sequences and produced *de novo* MCC scores of 0.72, 0.74, and 0.73 for Greengenes, RDP, and SILVA, respectively. Clustering query sequences with OptiFit to Greengenes and SILVA in closed reference mode performed similarly, with median MCC scores of 0.85 and 0.77 respectively, while the median MCC was 0.35 when clustering to RDP (Figure 2.3; "db: Greengenes", "db: SILVA", and "db: RDP"). For comparison, clustering datasets with OptiClust produced an average MCC score of 0.86 (Figure 2.3; "*de novo*"). This gap in OTU quality mostly disappeared when clustering in open reference mode, which produced median MCCs of 0.86 with Greengenes, 0.86 with SILVA, and 0.86 with the RDP. Thus, open reference OptiFit produced OTUs of very similar quality as *de novo* clustering with OptiClust, and closed reference OptiFit followed closely behind as long as a suitable reference database was chosen.

Since closed reference clustering does not cluster query sequences that could not be clustered into reference OTUs, an additional measure of clustering performance to consider is the fraction of query sequences that were able to be clustered. On average, more sequences were clustered with Greengenes as the reference (59%) than with SILVA (50%) or with the RDP (9.7%) (Figure 2.3). This mirrored the result reported above that Greengenes produced better OTUs in terms of MCC score than either SILVA or RDP. Note that *de novo* and open reference clustering methods always cluster 100% of sequences into OTUs. The database chosen affects the final closed reference OTU assignments considerably in terms of both MCC score and fraction of query sequences that could be clustered into the reference OTUs.

Despite the drawbacks, closed reference methods have been used when fast execution speed is required, such as when using very large datasets [61]. To compare performance in terms of speed, we repeated each OptiFit and OptiClust run 100 times and measured the execution time. Across all dataset and database combinations, closed reference OptiFit outperformed both OptiClust and open reference OptiFit (Figure 2.3). For example, with the human dataset fit to SILVA reference OTUs, the average run times in seconds were 406.8 for closed reference OptiFit, 455.3 for *de novo* clustering the dataset, and 559.4 for open reference OptiFit. Thus, the OptiFit algorithm continues the precedent that closed reference clustering sacrifices OTU quality for execution speed.

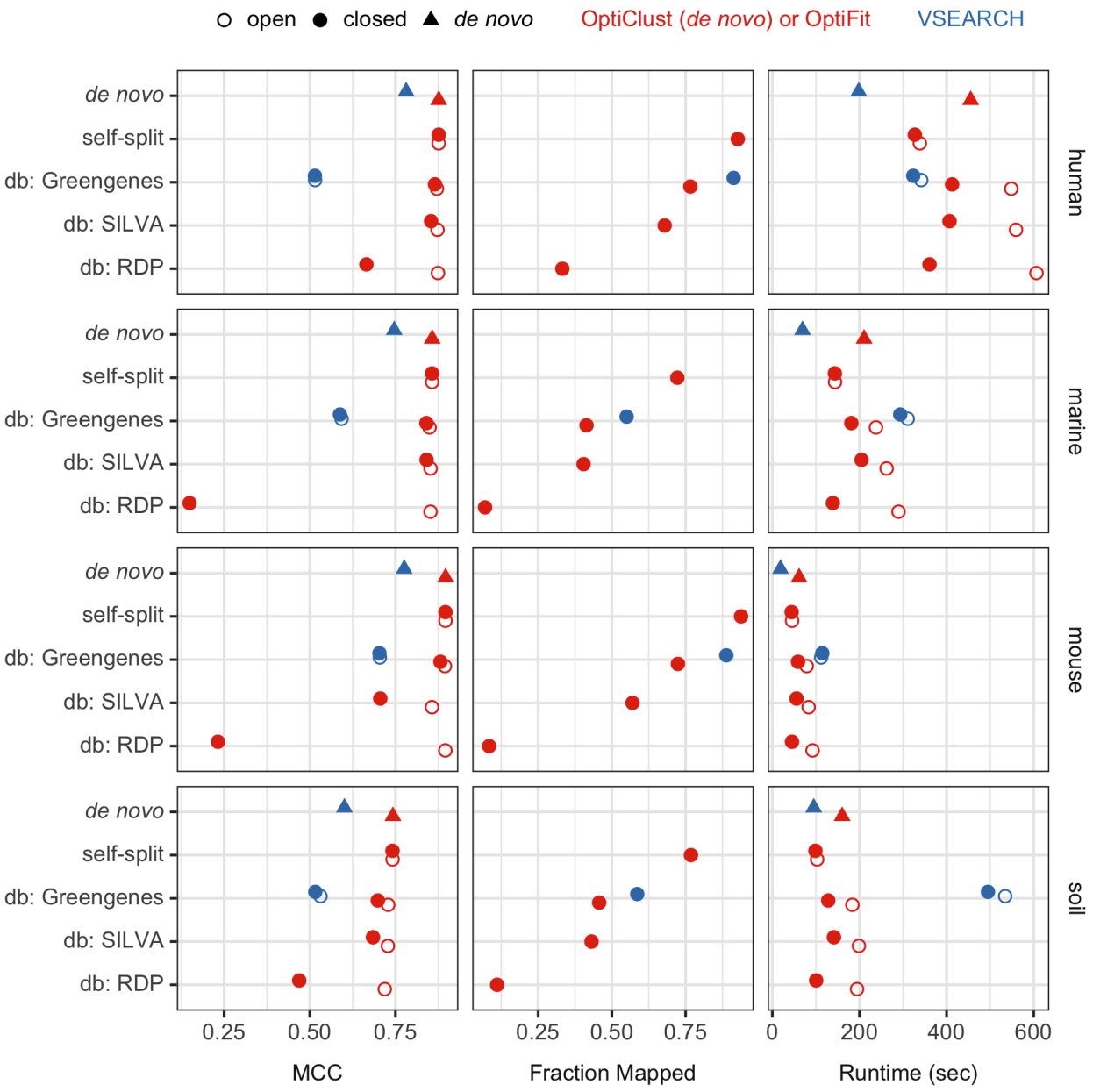
To compare to the reference clustering methods used by QIIME2, we clustered each dataset with VSEARCH against the Greengenes database of OTUs previously clustered at 97% sequence similarity. Each reference OTU from the Greengenes 97% database contains one reference sequence, and VSEARCH maps sequences to the reference based on each individual query sequence's similarity to the single reference sequence. In contrast, OptiFit

Figure 2.2: The OptiFit benchmarking workflow



Reference sequences from Greengenes, the RDP, and SILVA were downloaded, preprocessed with mothur by trimming to the V4 region, and clustered *de novo* with OptiClust for 100 repetitions. Datasets from human, marine, mouse, and soil microbiomes were downloaded, preprocessed with mothur by aligning to the SILVA V4 reference alignment, then clustered *de novo* with OptiClust for 100 repetitions. Individual datasets were fit to reference databases with OptiFit; OptiFit was repeated 100 times for each dataset and database combination. Datasets were also randomly split into a reference and query fraction, and the query sequences were fit to the reference sequences with OptiFit for 100 repetitions. The final MCC score was reported for all OptiClust and OptiFit repetitions.

Figure 2.3: OptiFit results with databases as references



The median MCC score, fraction of query sequences that mapped in closed-reference clustering, and runtime in seconds from repeating each clustering method 100 times. Each dataset underwent three clustering strategies; 1) *de novo* clustering the whole dataset using OptiClust, 2) splitting the dataset with 50% of the sequences as a reference set and the other 50% as a query set, clustering the references using OptiClust, then clustering the query sequences to the reference OTUs with OptiFit, and 3) clustering the dataset to a reference database (Greengenes, SILVA, or RDP). Reference-based clustering was repeated with open and closed mode. For additional comparison, VSEARCH was used for *de novo* and reference-based clustering against the Greengenes database.

accepts reference OTUs which each may contain multiple sequences, and the sequence similarity between all query and reference sequences is considered when assigning sequences to OTUs. In closed reference mode, OptiFit produced 27.2% higher quality OTUs than VSEARCH in terms of MCC score, but VSEARCH was able to cluster 24.9% more query sequences than OptiFit to the Greengenes reference database (Figure 2.3). This is because VSEARCH only considers the distances between each query sequence to the single reference sequence, while OptiFit considers the distances between all pairs of reference and query sequences in an OTU. When open reference clustering, OptiFit produced higher quality OTUs than VSEARCH against the Greengenes database, with median MCC scores of 0.86 and 0.56, respectively. In terms of run time, OptiFit outperformed VSEARCH in both closed and open reference mode by 53.6% and 44.0% on average, respectively. Thus, the more stringent OTU definition employed by OptiFit, which prefers the query sequence to be similar to all other sequences in the OTU rather than to only one sequence, resulted in fewer sequences being clustered to reference OTUs than when using VSEARCH, but caused OptiFit to outperform VSEARCH in terms of both OTU quality and execution time.

2.3.3 Reference clustering with split datasets

When performing reference clustering against public databases, the database chosen greatly affects the quality of OTUs produced. OTU quality may be poor when the reference database consists of sequences that are too unrelated to the samples of interest, such as when samples contain novel populations. While *de novo* clustering overcomes the quality limitations of reference clustering to databases, OTU assignments are not consistent when new sequences are added. Researchers may wish to cluster new sequences to existing OTUs or to compare OTUs across studies. To determine how well OptiFit performs for clustering new sequences to existing OTUs, we employed a split dataset strategy, where each dataset was randomly split into a reference fraction and a query fraction. Reference sequences were clustered *de novo* with OptiClust, then query sequences were clustered to the *de novo* OTUs with OptiFit.

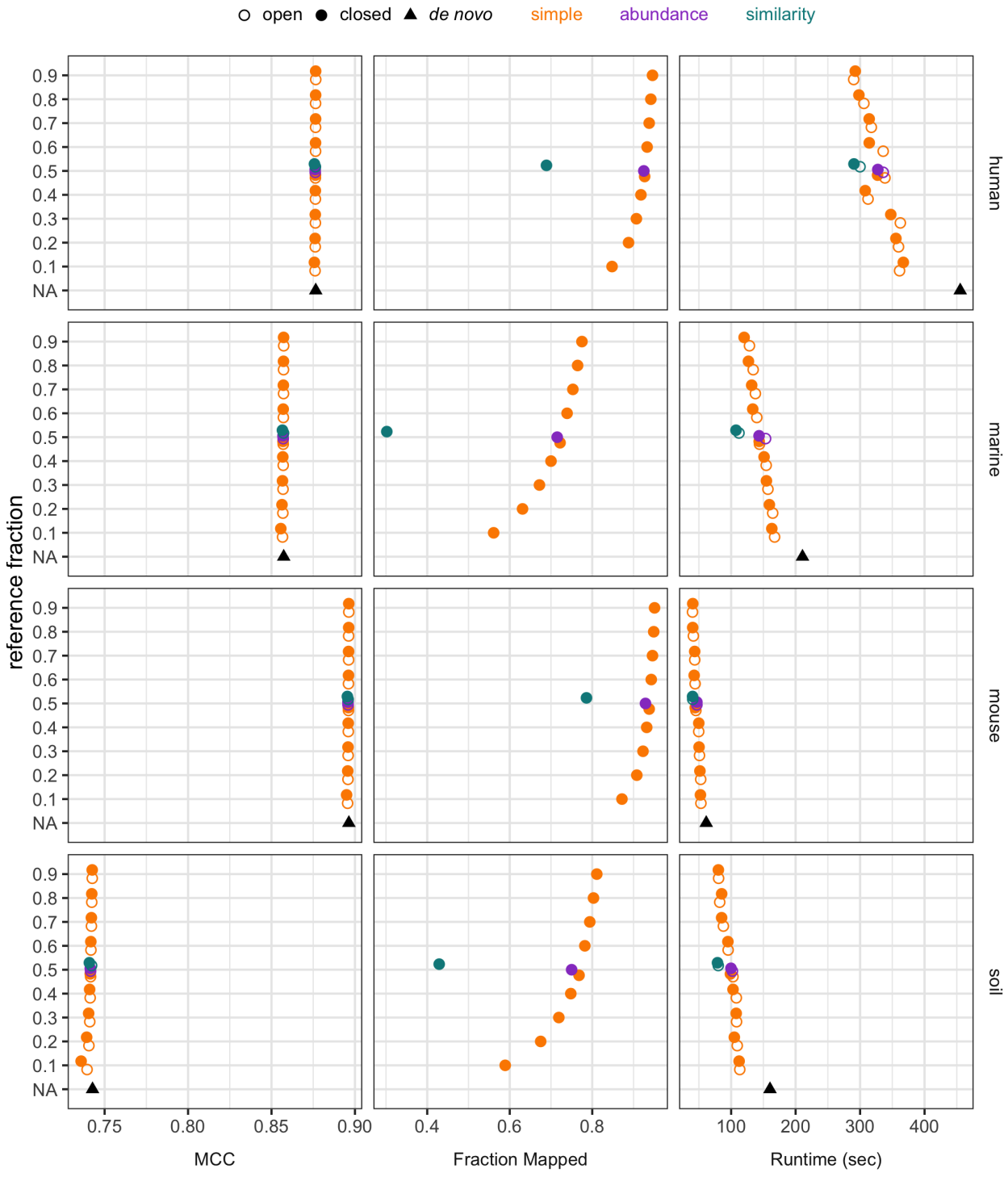
First, we tested whether OptiFit performed as well as *de novo* clustering when using the split dataset strategy with half of the sequences selected for the reference by a simple random sample (a 50% split) (Figure 2.3; "self-split"). OTU quality was similar to that from OptiClust regardless of mode (0.031% difference in median MCC). In closed reference mode, OptiFit was able to cluster 84.9% of query sequences to reference OTUs with the split strategy, a great improvement over the average 59% of sequences clustered to the Greengenes database. In terms of run time, closed and open reference OptiFit performed faster than OptiClust on whole datasets by 39.6% and 36.8%, respectively. Random access memory

(RAM) usage was similar, with OptiFit requiring slightly more RAM in gigabytes than OptiClust. Open and closed reference OptiFit required 1.8% and 1.2% more RAM than OptiClust, respectively (data not shown). The split dataset strategy also performed 6.7% faster than the database strategy in closed reference mode and 65.5% faster in open reference mode. Thus, reference clustering with the split dataset strategy creates as high quality OTUs as *de novo* clustering yet at a faster run time, and fits far more query sequences than the database strategy.

While we initially tested this strategy using a 50% split of the data into reference and query fractions, we next investigated whether there was an optimal reference fraction size. To identify the best reference size, reference sets with 10% to 90% of the sequences were created, with the remaining sequences used for the query (Figure 2.4). OTU quality was remarkably consistent across reference fraction sizes. For example, splitting the human dataset 100 times yielded a coefficient of variation (i.e. the standard deviation divided by the mean) of 0.0018 for the MCC score across all fractions. Run time generally decreased as the reference fraction increased; for the human dataset, the median run time was 364.0 seconds with 10% of sequences in the reference and 290.8 seconds with 90% of sequences in the reference. The RAM usage was virtually the same across reference fraction sizes, with a coefficient of variation of 0.00089 for the human dataset (data not shown). In closed reference mode, the fraction of sequences that mapped increased as the reference size increased; for the human dataset, the median fraction mapped was 0.85 with 10% of sequences in the reference and 0.95 with 90% of sequences in the reference. These trends held for the other datasets as well. Thus, the reference fraction did not affect OTU quality in terms of MCC score nor the memory usage, but did affect the run time and the fraction of sequences that mapped during the closed reference clustering.

After testing the split strategy using a simple random sample to select the reference sequences, we then investigated other methods of splitting the data. We tested three methods for selecting the fraction of sequences to be used as the reference at a size of 50%: a simple random sample, weighting sequences by relative abundance, and weighting by similarity to other sequences in the dataset (Figure 2.4). OTU quality in terms of MCC was similar across all three sampling methods (median MCC of 0.86). In closed-reference clustering mode, the fraction of sequences that mapped were similar for simple and abundance-weighted sampling (median fraction mapped of 0.85 and 0.84, respectively), but worse for similarity-weighted sampling (median fraction mapped of 0.56). While simple and abundance-weighted sampling produced better quality OTUs than similarity-weighted sampling, OptiFit performed faster on similarity-weighted samples with a median runtime of 103.9 seconds compared to 135.4 and 134.8 seconds for simple and abundance-weighted sampling, respectively. Thus,

Figure 2.4: OptiFit results with datasets as self-references



The median MCC score, fraction of query sequences that mapped in closed-reference clustering, and runtime in seconds from repeating each clustering method 100 times. Each dataset was split into a reference and query fraction. Reference sequences were selected via a simple random sample, weighting sequences by relative abundance, or weighting by similarity to other sequences in the dataset. With the simple random sample method, dataset splitting was repeated with reference fractions ranging from 10% to 90% of the dataset and for 100 random seeds. *De novo* clustering each dataset with OptiClust is also shown for comparison.

employing more complicated sampling strategies such as abundance-weighted and similarity-weighted sampling did not confer any advantages over selecting the reference via a simple random sample, and in fact decreased OTU quality in the case of similarity-weighted sampling.

2.4 Discussion

We developed a new algorithm for clustering sequences to existing OTUs and have demonstrated its suitability for reference-based clustering. OptiFit makes the iterative method employed by OptiClust available for tasks where reference-based clustering is required. We have shown that OTU quality is similar between OptiClust and OptiFit in open reference mode, regardless of strategy employed. Open reference OptiFit performs slower than OptiClust due to the additional *de novo* clustering step, so users may prefer OptiClust for tasks that do not require reference OTUs.

When clustering to public databases, OTU quality dropped in closed reference mode to different degrees depending on the database and dataset source, and no more than half of query sequences were able to be clustered into OTUs across any dataset/database combination. This may reflect limitations of reference databases, which are unlikely to contain sequences from novel microbes. This drop in quality was most notable with the RDP reference, which contained only 16,192 sequences compared to 173,648 sequences in SILVA and 174,979 in Greengenes. Note that Greengenes has not been updated since 2013 at the time of this writing, while SILVA and the RDP are updated regularly. We recommend that users who require an independent reference database opt for large databases with regular updates and good coverage of microbial diversity for their environment. Since OptiClust still performs faster than open reference OptiFit and creates higher quality OTUs than closed reference OptiFit with the database strategy, we recommend using OptiClust rather than clustering to a database whenever consistent OTUs are not required.

The OptiClust and OptiFit algorithms produced higher quality OTUs than VSEARCH in open reference, closed reference, or *de novo* modes. However, VSEARCH was able to cluster more sequences to OTUs than OptiFit in closed reference mode. While both OptiFit and VSEARCH use a distance or similarity threshold for determining how to cluster sequences into OTUs, VSEARCH is more permissive than OptiFit regardless of mode. The OptiFit and OptiClust algorithms use all of the sequences to define an OTU, preferring that all pairs of sequences (including reference and query sequences) in an OTU are within the distance threshold in order to maximize the MCC. In contrast, VSEARCH only requires each query sequence to be similar to the single centroid sequence that seeded the OTU, thus

allowing pairs of query sequences to be less similar to each other than the threshold specified. Because of this, VSEARCH sacrifices OTU quality by allowing more dissimilar sequences to be clustered into the same OTUs.

When clustering with the split dataset strategy, OTU quality was remarkably similar when reference sequences were selected by a simple random sample or weighted by abundance, but quality was slightly worse when sequences were weighted by similarity. We recommend using a simple random sample since the more sophisticated reference selection methods do not offer any benefit. The similarity in OTU quality between OptiClust and OptiFit with this strategy demonstrates the suitability of using OptiFit to cluster sequences to existing OTUs, such as when comparing OTUs across studies. However, when consistent OTUs are not required, we recommend using OptiClust for *de novo* clustering over the split strategy with OptiFit since OptiClust is simpler to execute but performs similarly in terms of both run time and OTU quality.

Unlike existing reference-based methods that cluster query sequences to a single centroid sequence in each reference OTU, OptiFit considers all sequences in each reference OTU when clustering query sequences, resulting in OTUs of a similar high quality as those produced by the *de novo* OptiClust algorithm. Potential applications include clustering sequences to reference databases, comparing taxonomic composition of microbiomes across different studies, or using OTU-based machine learning models to make predictions on new data. OptiFit fills the missing option for clustering query sequences to existing OTUs that does not sacrifice OTU quality for consistency of OTU assignments.

2.5 Materials and Methods

2.5.1 Data processing steps

We downloaded 16S rRNA gene amplicon sequences from four published datasets isolated from soil [55], marine [56], mouse gut [57], and human gut [30] samples. These datasets contain sequences from the V4 region of the 16S rRNA gene and represent a selection of the broad types of natural communities that microbial ecologists study. We processed the raw sequences using mothur according to the Schloss Lab MiSeq SOP [62] and accompanying study by Kozich *et al.* [63]. These steps included trimming and filtering for quality, aligning to the SILVA reference alignment [59], discarding sequences that aligned outside the V4 region, removing chimeric reads with UCHIME [64], and calculating distances between all pairs of sequences within each dataset prior to clustering.

2.5.2 Reference database clustering

To generate reference OTUs from public databases, we downloaded sequences from the Greengenes database (v13_8_99) [58], SILVA non-redundant database (v132) [59], and the Ribosomal Database Project (v16) [60]. These sequences were processed using the same steps outlined above followed by clustering sequences into *de novo* OTUs with OptiClust. Processed reads from each of the four datasets were clustered with OptiFit to the reference OTUs generated from each of the three databases. When reference clustering with VSEARCH, processed datasets were clustered directly to the unprocessed Greengenes 97% OTU reference alignment, since this method is how VSEARCH is typically used by the QIIME2 software for reference-based clustering [54], [65].

2.5.3 Split dataset clustering

For each dataset, half of the sequences were selected to be clustered *de novo* into reference OTUs with OptiClust. We used three methods for selecting the subset of sequences to be used as the reference: a simple random sample, weighting sequences by relative abundance, and weighting by similarity to other sequences in the dataset. Dataset splitting was repeated with 100 random seeds. With the simple random sampling method, dataset splitting was also repeated with reference fractions ranging from 10% to 90% of the dataset. For each dataset split, the remaining query sequences were clustered into the reference OTUs with OptiFit.

2.5.4 Benchmarking

OptiClust and OptiFit randomize the order of query sequences prior to clustering and employ a random number generator to break ties when OTU assignments are of equal quality. As a result, they produce slightly different OTU assignments when repeated with different random seeds. To capture any variation in OTU quality or execution time, clustering was repeated with 100 random seeds for each combination of parameters and input datasets. We used the benchmark feature provided by Snakemake to measure the run time of every clustering job. We calculated the MCC on each set of OTUs to quantify the quality of clustering, as described by Westcott *et al.* [50].

2.5.5 Data and code availability

We implemented the analysis workflow in Snakemake [66] and wrote scripts in R [67], Python [68], and GNU bash [69]. Software used includes mothur v1.47.0 [70], VSEARCH v2.15.2

[53], the tidyverse metapackage [71], R Markdown [72], ggraph [73], ggtext [74], numpy [75], the SRA toolkit [76], and conda. The complete workflow and supporting files required to reproduce this manuscript are available at https://github.com/SchlossLab/Sovacool_OptiFit_mSphere_2022.

2.6 Acknowledgements

We thank members of the Schloss Lab for their feedback on the figures.

KLS received support from the NIH Training Program in Bioinformatics (T32 GM070449). Salary support for PDS came from NIH grants R01CA215574 and U01AI124255. The funders had no role in study design, data collection and interpretation, or the decision to submit the work for publication.

2.7 Author Contributions

KLS wrote the analysis code, evaluated the algorithm, and wrote the original draft of the manuscript. SLW designed and implemented the OptiFit algorithm and assisted in debugging the analysis code. MBM and GAD contributed analysis code. PDS conceived the study, supervised the project, and assisted in debugging the analysis code. All authors reviewed and edited the manuscript.

CHAPTER 3

Predicting Severity of *C. difficile* Infections from the Taxonomic Composition of the Gut Microbiome

3.1 Preamble

This chapter aims to predict CDI severity from the taxonomic composition of the gut microbiome. We trained models on OTU relative abundances to predict four different severity definitions, identified features of the microbiota that may prevent or promote severity, and assessed the potential clinical value of micriobiome-based prediction models.

I performed all of the analysis and created the figures and tables for this chapter. Other co-authors helped to conceive of the study, processed samples, and assisted in training ML models. This chapter will be submitted to a peer-reviewed journal with the following co-authors: Kelly L. Sovacool, Sarah E. Tomkovich, Megan L. Coden, Jenna Wiens, Vincent B. Young, Krishna Rao, and Patrick D. Schloss.

3.2 Introduction

Clostridoides difficile infection (CDI) is the most common nosocomial infection in the United States, and community-acquired cases are on the rise [77, 12]. The classic CDI case typically occurs soon after antibiotic use, which perturbs the protective gut microbiota and allows *C. difficile* to proliferate [22]. Non-antibiotic medications including proton-pump inhibitors and osmotic laxatives have also been associated with increased CDI susceptibilty and inhibited clearance [10, 11]. Diarrhea is the primary symptom, with some patients developing colitis, toxic megacolon, or requiring intensive care with an in-hospital mortality rate of approximately 8-9% [19, 20]. Furthermore, 5-20% of initial cases reoccur within 2-8 weeks, and recurrent cases are associated with increased morbidity and mortality risk [78, 22]. Patient

risk factors for CDI-related morbidity and mortality include age greater than 65 years, history of recurrent CDI, and co-morbid chronic illnesses [79]. CDI remains a significant burden on the US health care system with approximately 500,000 cases annually [80, 81].

There is a need for robust, accurate methods to identify patients at risk of severe CDI outcomes. When paired with treatment options that may reduce risk of severity, prediction models can guide clinician decision-making to improve patient outcomes while minimizing harms and costs from unnecessary treatment. Clinicians could choose more aggressive treatment options for patients predicted as being at high risk for severity, while using less costly or less invasive treatments for low-risk patients. Numerous scoring systems for predicting severe CDI outcomes based on patient clinical factors have been developed, but none have generalized to external datasets nor are any in use in routine clinical practice [82, 83]. Rather than relying on limited sets of human-curated variables, machine learning (ML) is a promising approach that allows for use of thousands of features to classify samples and predict outcomes. Indeed, ML models trained on entire electronic health record (EHR) data have demonstrated improved performance over curated models [84, 85]. However, EHR-based ML models also suffer from generalizability issues as EHR standards and structures vary widely across hospital systems, making it difficult to integrate disparate EHR data and deploy models in different hospitals [86].

Aside from patient factors encoded in EHRs, the state of the patient gut microbiome is a promising factor to predict severity, as the host microbiota can play either a protective or harmful role in *C. difficile* colonization, infection, and clearance. Mouse studies have found that the initial taxonomic composition of the gut microbiome predicts differences in clearance, moribundity, and cecal tissue damage in mice infected with CDI [16, 18]. Identifying features of the human gut microbiota that promote or prevent severe infections can guide further experiments to elucidate microbial mechanisms of CDI severity, and incorporating these features into CDI severity models may improve model performance to help guide clinical treatment decisions. Furthermore, ML models trained on microbiome data may be more generalizable across disparate datasets compared to EHR data as long as the same metagenomic or marker gene sequencing protocol is used across datasets [87]. While the variables encoded in EHRs vary across hospitals depending on individual hospital practices and the EHR software vendor used, the definition of a microbial marker gene is universal.

We set out to investigate whether ML models trained on the taxonomic composition of the gut microbiome can predict CDI severity in a human cohort, whether the severity definition employed affects model performance, and whether there is potential clinical value in deploying OTU-based models. Stool samples from 1,277 CDI patients were collected on the day of diagnosis and 16S rRNA gene amplicon sequencing was performed, followed

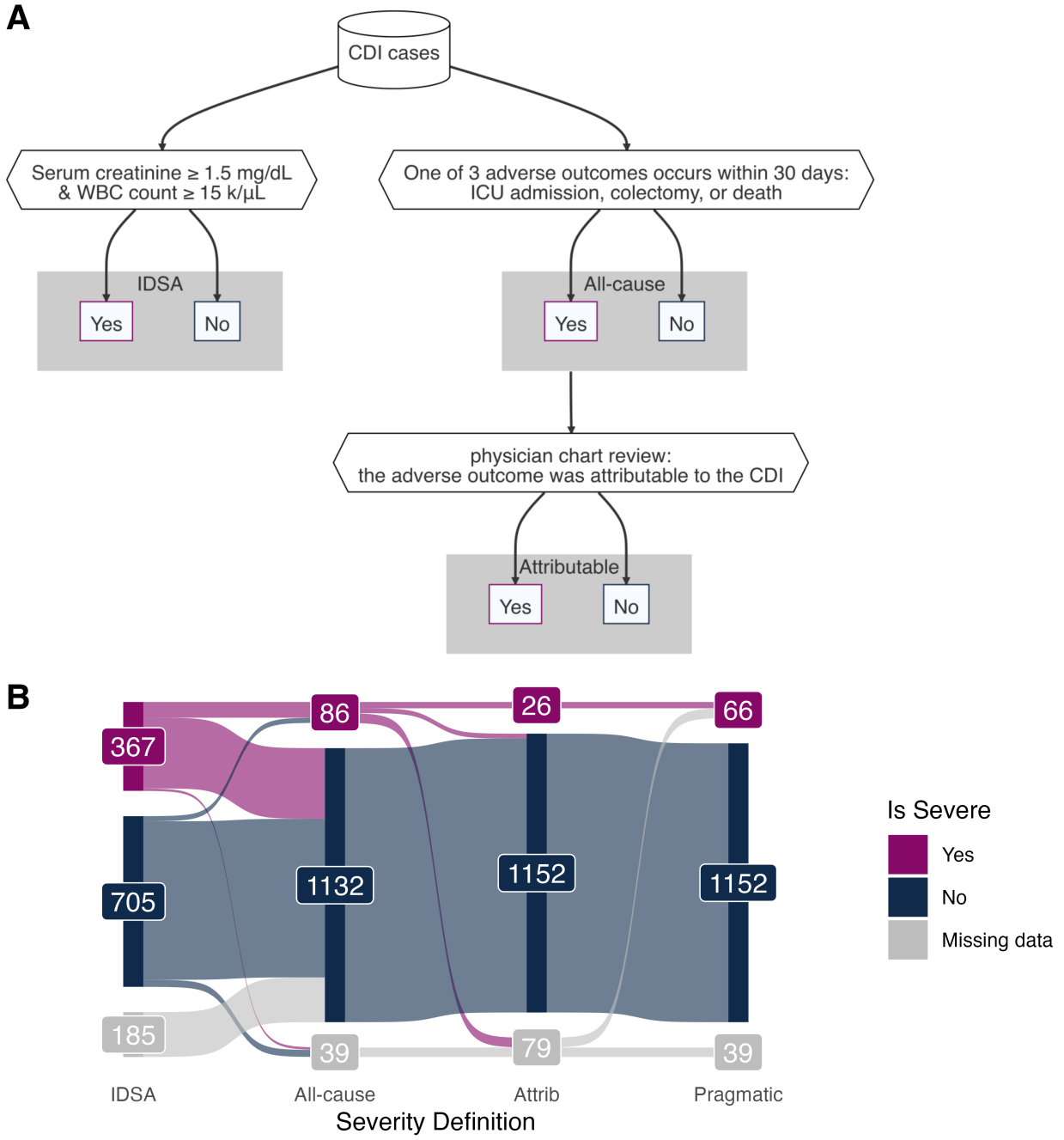
by clustering sequences into Operational Taxonomic Units (OTUs). We then trained ML models to classify or predict each of four severity definitions from OTU relative abundances, identified which microbial features contributed most to model performance, and conducted a proof-of-concept analysis of the potential clinical value of these OTU-based models and compared these to prior EHR-based models.

3.3 Results

3.3.1 CDI severity

There is not currently a consensus definition of CDI severity. Some scoring systems leverage clinical data available during the course of CDI, while others focus on adverse outcomes of CDI at 30 days after diagnosis [79, 29]. We explored four different ways to define CDI cases as severe or not (Figure 3.1). The Infectious Diseases Society of America (IDSA) definition of severe CDI is based on laboratory values collected on the day of diagnosis, with a case being severe if serum creatinine level is greater than or equal to $1.5 mg/dL$ and the white blood cell count is greater than or equal to $15 k/\mu L$ [88]. Although data for the IDSA score is straightforward to collect, it is known to be a poor predictor of adverse outcomes [89]. The remaining definitions we employed focus on the occurrence of adverse outcomes, which may be more clinically relevant. The "attributable" severity definition is based on disease-related complications defined by the Centers for Disease Control and Prevention, where an adverse event of ICU admission, colectomy, or death occurs within 30 days of CDI diagnosis, and the adverse event is determined to be attributable to the CDI by physician chart review [90]. However, physician chart review is time-consuming and has not been completed for all cases (n=46 out of 86 cases with an adverse outcome), so we defined "all-cause" severity where a case is severe if an adverse event occurs within 30 days of the diagnosis regardless of the cause of the adverse event. Finally, we defined a "pragmatic" severity definition that makes use of the attributable definition when available and uses the all-cause definition when chart review has not been completed, allowing us to use as many samples as we have available while taking physicians' expert opinions into account where possible (Figure 3.1 B). We trained ML models to classify (in the case of the IDSA definition) or predict (in the case of the three other definitions) severity and determined how well OTU-based models perform for each definition.

Figure 3.1: CDI severity definitions.



A) Decision flow chart to define CDI cases as severe according to the Infectious Diseases Society of America (IDSA) based on lab values, the occurrence of an adverse outcome due to any cause (All-cause), and the occurrence of disease-related complications confirmed as attributable to CDI with chart review (Attributable). **B)** The proportion of severe CDI cases labelled according to each definition. An additional 'Pragmatic' severity definition uses the Attributable definition when possible, and falls back to the All-cause definition when chart review is not available. See Table 3.1 for sample counts and proportions of severe cases across severity definitions.

Table 3.1: **Sample counts and proportion of severe cases.** Each severity definition has a different number of patient samples available, as well as a different proportion of cases labelled as severe.

(a) Full datasets

Severity	n	% severe
All-cause	1,218	7.1
Attributable	1,178	2.2
IDSA	1,072	34.2
Pragmatic	1,218	5.4

(b) Intersection of samples with all labels available

Severity	n	% severe
All-cause	993	4.6
Attributable	993	2.6
IDSA	993	32.7
Pragmatic	993	2.6

3.3.2 Model performance

We first set out to train the best models possible for each severity definition. Not all samples have outcomes available for all four severity definitions due to missing data for some patient lab values and incomplete chart review (Figure 3.1 B), thus each severity definition had a different number of samples when using as many samples as possible (Table 3.1 A). We referred to these as the full datasets. Random forest models were trained on 100 splits of the datasets into training and test sets, and performance was evaluated on the held-out test set using the area under the receiver-operator characteristic curve (AUROC). Since the severity outcomes were highly imbalanced with different proportions of severe samples between definitions, we also calculated the balanced precision and the area under the balanced precision-recall curve (AUBPRC) as first proposed by Wu *et al.* to describe the precision that would be expected if the outcomes were balanced [91].

After training on the full datasets, the performance as measured by the AUROCs of the training set cross-validation folds were similar to those of the held-out test sets, indicating that the models are neither overfit nor underfit (Figure 3.2 A). As measured by AUROC on the held-out test sets, models predicting pragmatic severity performed best with a median AUROC of 0.69, and this was significantly different from that of the other definitions on the full datasets ($P < 0.05$). Models predicting IDSA, all-cause, and attributable severity performed similarly with median test set AUROCs of 0.61, 0.63, and 0.61 respectively. The test set AUROCs were not significantly different ($P > 0.05$) for attributable and IDSA nor for attributable and all-cause, but the IDSA and all-cause AUROCs were significantly different from each other ($P < 0.05$). We plotted the receiver-operator characteristic curve and found that the pragmatic severity models outperformed the others at all specificity values (Figure 3.2 B). For comparison, a prior study with a different dataset trained a logistic regression model on electronic health record data extracted on the day of CDI diagnosis to predict attributable severity, yielding an AUROC of 0.69 [85]. While our attributable

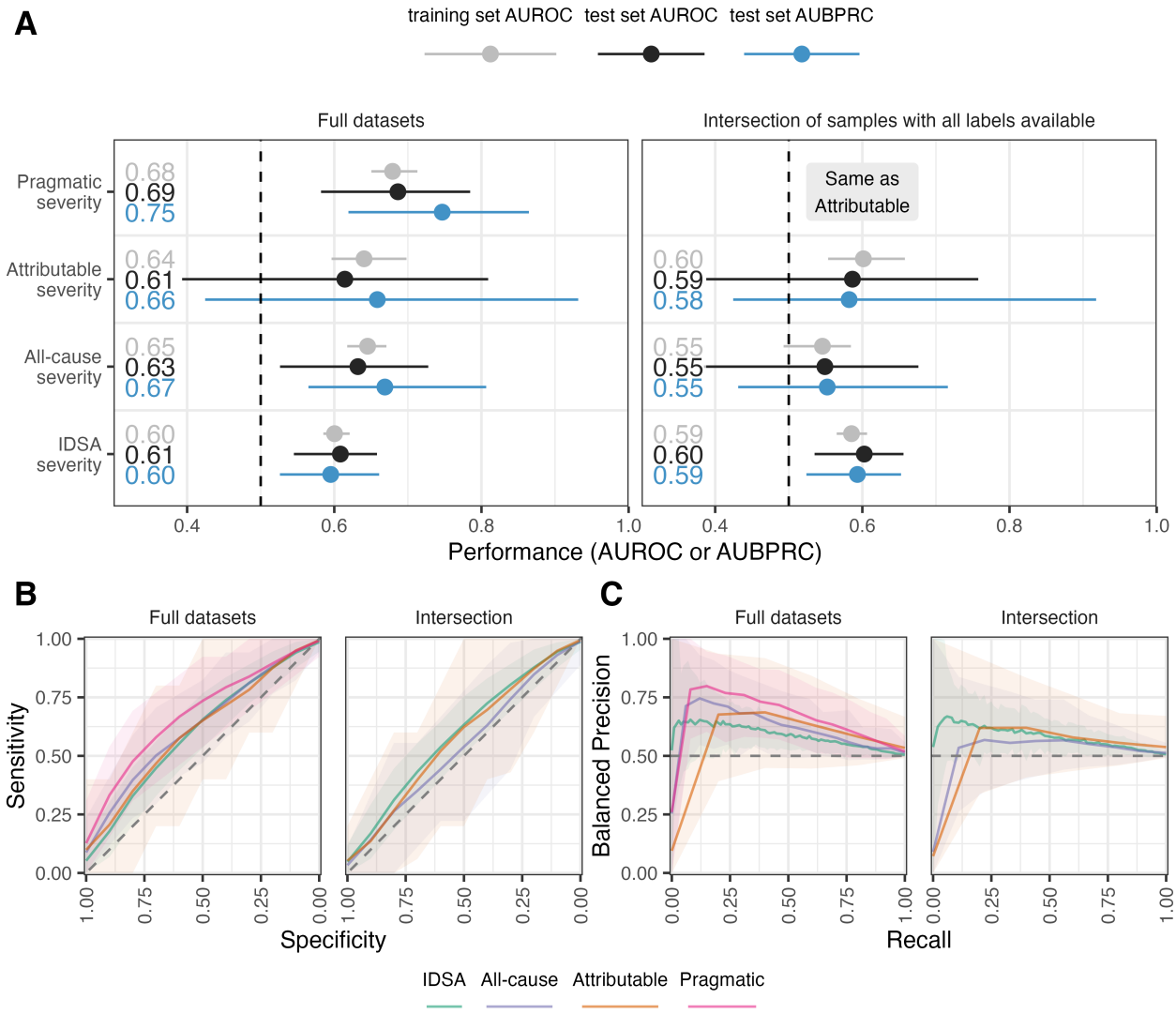
severity model did not meet this performance, the pragmatic severity model performed just as well as the EHR-based model in terms of AUROC.

Since the data are highly imbalanced with only a small proportion of CDI cases having a severe outcome, evaluating the trade-off between precision and recall is more informative than the receiver-operator characteristic because precision and recall do not consider true negatives, which may overinflate the AUROC. However, unlike for AUROC, the baseline for the area under the precision-recall curve depends on the proportion of positive outcomes (i.e. severe cases) in the data, which vary across these severity definitions. To allow comparison of precision across datasets with different proportions of positives, Wu *et al.* introduced the concept of balanced precision, a transformation of precision based on Bayes' theorem that represents the precision that would have been expected if the proportion of positives were balanced at 0.5 [91]. Reporting the area under the balanced precision-recall curve (AUBPRC) allows us to compare the trade-off between precision and recall for our different severity defintions. The test set median AUBPRCs from the full datasets followed a similar pattern as the test set AUROCs with 0.60 for IDSA severity, 0.67 for all-cause severity, 0.66 for attributable severity, and 0.75 for pragmatic severity. The AUBPRCs were significantly different from each other ($P < 0.05$) for each pair of severity definitions except for attributable versus all-cause. We plotted the balanced precision-recall curve and found that the IDSA definition outperformed all other models at very low recall values, but the others outperform IDSA at all other points of the curve (Figure 3.2 C). The 95% confidence intervals overlapped the baseline AUROC and AUBPRC for the attributable severity models, while all others did not overlap the baseline.

While it is advantageous to use as much data as available to train the best models possible, comparing performances of models trained on different subsets of the data is not entirely fair. To enable fair comparisons of the model performances across different severity definitions, we also selected the intersection of samples (n=993) that had labels for all four severity definitions and repeated the model training and evaluation process on this intersection dataset. The attributable definition is exactly the same as the pragmatic definition for the intersection dataset, as we defined pragmatic severity to use the attributable definition when available. The performance results on the intersection dataset are shown in the right facets of each panel of Figure 3.2.

As with the full datasets, the AUROCs of the training sets and test sets were similar within each severity definition. The median test set AUROCs were 0.60 for IDSA severity, 0.55 for all-cause severity, 0.59 and for attributable severity. The AUROCs on the intersection dataset were significantly different for all-cause versus attributable and all-cause versus IDSA severity ($P < 0.05$), but not for IDSA versus attributable severity ($P > 0.05$). The median test set

Figure 3.2: Performance of ML models.



In the left facets, models were trained on the full datasets, with different numbers of samples available for each severity definition. In the right facets, models were trained on the same dataset consisting of the intersection of samples with labels available for all definitions. Note that the intersection dataset has exactly the same labels for attributable and pragmatic severity, thus these have identical performance. **A)** Area under the receiver-operator characteristic curve (AUROC) for the test sets and cross-validation folds of the training sets, and the area under the balanced precision-recall curve (AUBPRC) for the test sets. Each point is annotated with the median performance across 100 train/test splits with tails as the 95% CI. **B)** Receiver-operator characteristic curves for the test sets. Mean specificity is reported at each sensitivity value, with ribbons as the 95% CI. **C)** Balanced precision-recall curves for the test sets. Mean balanced precision is reported at each recall (sensitivity) value, with ribbons as the 95% CI. Original unbalanced precision-recall curves are shown in Supplementary Figure 3.5.

AUBPRCs were 0.59 for IDSA severity, 0.55 for all-cause severity, 0.58 and for attributable severity. Just as with the AUROCs, the AUBPRCs were significantly different for all-cause versus attributable and all-cause versus IDSA severity ($P < 0.05$), but not for IDSA versus attributable severity ($P > 0.05$). For all severity definitions, performance dropped between the full dataset and the intersection dataset since fewer samples are available, but this effect is least dramatic for IDSA severity as the full and intersection datasets are more similar for this definition (Table 3.1 B). The 95% confidence interval overlaps with the baseline for both AUROC and AUBPRC for all definitions on the intersection dataset except for IDSA severity.

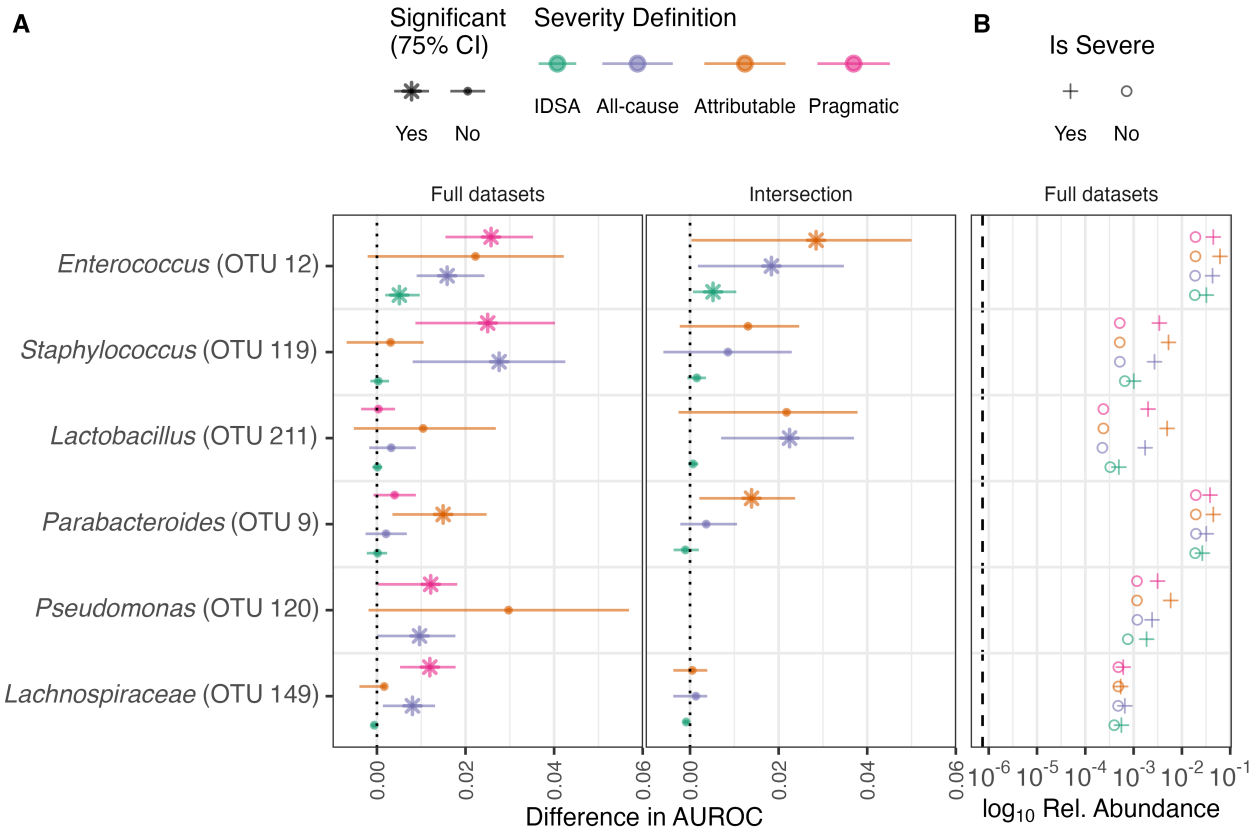
3.3.3 Feature importance

We performed permutation feature importance to determine which OTUs contributed the most to model performance. An OTU was considered important if performance decreased when it was permuted in at least 75% of the train/test splits, with greater differences in AUROC meaning greater importance. We plotted mean decrease in AUROC alongside $\log_{10}$-transformed mean relative abundances for the top OTUs (Figure 3.3). *Enterococcus* was the most important OTU, being significantly important for all models except for attributable severity on the full dataset. *Staphylococcus* was important for the pragmatic and all-cause definitions on the full datasets, but not for models trained on the intersection dataset. *Lactobacillus* was important only for the all-cause definition on the intersection dataset. All remaining OTUs had differences in AUROC < 0.02 and were only significantly important in one or two of the models at most. All of the significantly importance OTUs had an increased mean relative abundance in severe cases relative to not severe cases.

3.3.4 Estimating clinical value

Even if a model performs well, it may not be useful in a clinical setting unless it can guide clinicians to choose between treatment options. At this time, we are not aware of any direct evidence that a particular treatment reduces the risk of severe CDI outcomes. However, with some assumptions we offer a proof-of-concept analysis of the potential clinical value of OTU-based severity prediction models when paired with treatments that may reduce severity. When considering the suitability of a model for deployment in clinical settings, the number needed to screen (NNS) is a highly relevant metric representing how many patients must be predicted as severe by the model to identify one true positive. NNS is calculated as the reciprocal of precision (Equation 3.1) [92]. Similarly, the number needed to treat (NNT) is the number of true positive patients that must be treated by an intervention in order for one

Figure 3.3: Most important OTUs for model performance.



A) Feature importance via permutation test. For each OTU, the order of samples was randomized in the test set 100 times and the AUROC was re-calculated to estimate the permutation performance. OTUs with a greater difference in AUROC (actual performance minus permutation performance) are more important. Mean difference in AUROC and the 75% confidence interval (CI) is reported for each OTU that had a mean difference ≥ 0.01 for at least one severity definition, with starred OTUs being significant for the 75% CI. Notably, the OTU most likely corresponding to *C. difficile* was not important (see Supplementary Figure 3.6). Left: models were trained on the full datasets, with different numbers of samples available for each severity definition. Right: models were trained on the intersection of samples with all labels available for each definition. Note that Attributable and Pragmatic severity are exactly the same for the intersection dataset. *Pseudomonas* (OTU 120) is not shown for IDSA severity in the full datasets nor in the intersection dataset because it was removed during pre-processing due to having near-zero variance. **B)** Log_{10}-transformed mean relative abundances of the most important OTUs on the full datasets, grouped by severity (shape). The vertical dashed line is the limit of detection.

patient to benefit from the treatment. NNT is calculated as the reciprocal of the absolute risk reduction (ARR) from randomized controlled trials (Equation 3.2 and Equation 3.3) [93, 94, 95]. Multiplying NNS by NNT yields the number needed to benefit (NNB): the number of patients predicted to have a severe outcome who then benefit from the treatment (Equation 3.4) [96]. Thus the NNB pairs model performance with treatment effectiveness to estimate the benefit of using predictive models in clinical practice. Lower values of NNS, NNT, and NNB are better, with the minimum value being 1, as fewer patients must be screened and treated in order to benefit a single patient.

$$NNS = \frac{1}{Precision} \tag{3.1}$$

$$ARR = Control\ Event\ Rate - Experimental\ Event\ Rate \tag{3.2}$$

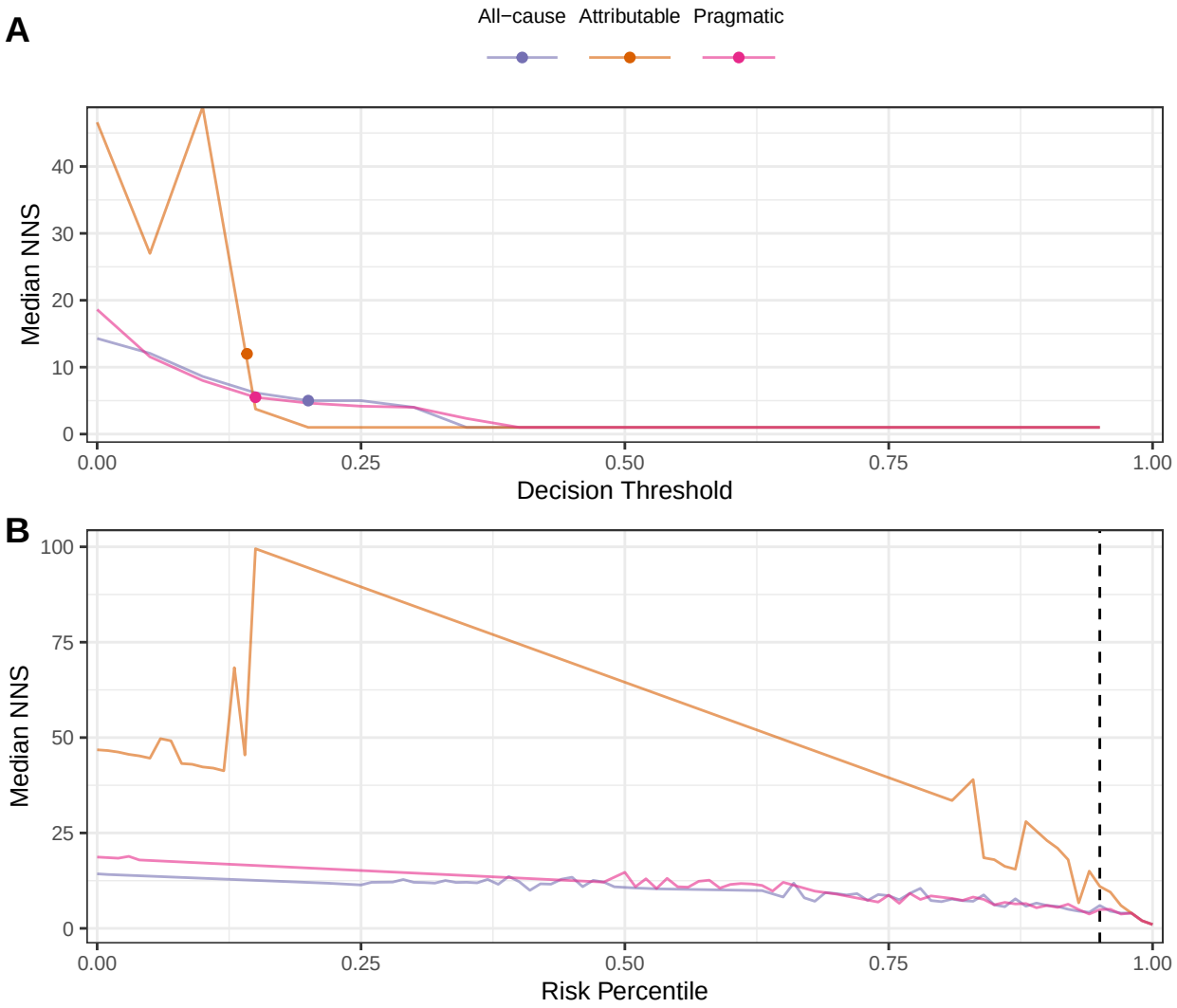
$$NNT = \frac{1}{ARR} \tag{3.3}$$

$$NNB = NNS \times NNT \tag{3.4}$$

Current clinical guidelines specify vancomycin and fidaxomicin as the standard antibiotics to treat CDI, with a preference for fidaxomicin due to its higher rate of sustained resolution of CDI and lower rate of recurrence [97]. The NNTs of fidaxomicin for sustained resolution and prevention of recurrence are each estimated to be 10 [98, 99]. However, fidaxomicin is considerably more expensive than vancomycin. If fidaxomicin were shown to reduce the risk of severe CDI outcomes, it could be preferentially prescribed to patients predicted to be at risk, while prescribing vancomycin to low-risk patients. If we assume that the superior efficacy of fidaxomicin for sustained resolution and reduced recurrence also translates to reducing the risk of severe outcomes, we can pair the NNT of fidaxomicin with the NNS of OTU-based prediction models to estimate the NNB.

To calculate a clinically-relevant NNS for these models, we computed the NNS across decision thresholds and risk percentiles for each prediction model trained on the full datasets (Figure 3.4). We excluded the IDSA severity models as the IDSA severity scores were calculated on the day of diagnosis, thus they are classification rather than prediction problems. Furthermore, IDSA severity scores do not correlate well with disease-related adverse events which are a more salient outcome to prevent. We report the median NNS for each decision threshold and risk percentile from 0 to 1 (Figure 3.4). The decision threshold is the risk level at which patients are predicted to have a severe outcome. For example, a decision threshold of 0.20 means that patients with at least a 20% risk of severity are predicted to have a severe outcome by the model. The decision threshold at a given risk percentile is different for each model, with the 95th percentile of risk corresponding to the decision threshold where 5% of

Figure 3.4: Model performance in terms of the number needed to screen across decision thresholds and risk percentiles.



The number needed to screen (NNS) represents how many patients must be predicted as severe by the model to identify one true positive (Equation 3.1). NNS ranges from 1 to infinity, with 1 being perfect. **A)** The median NNS was computed for each decision threshold from 0 to 1, incremented by 0.05. A decision threshold of 0.20 means that patients with at least a 20% risk of severity are predicted as severe. The points mark the decision threshold at the 95th percentile of risk for each severity prediction model, which corresponds to 5% of cases predicted to have a severe outcome. **B)** The median NNS is shown across risk percentiles. The vertical dashed line marks the 95th percentile of risk.

patients are predicted to have a severe outcome.

We further focused on the 95th percentile of risk in order to compare our models to EHR-based models from prior studies that reported model precision at this risk threshold. Among the models predicting severe outcomes, those trained on the full datasets performed best with an NNS of 5 for the all-cause definition, 12 for the attributable definition, and 5.5 for the pragmatic definition at the 95th percentile of risk (Figure 3.4). Multiplying the NNS of the OTU-based models by the estimated NNT of 10 for fidaxomicin yields NNB values of 50 for all-cause severity, 120 for attributable severity, and 55 for pragmatic severity. Thus, in a hypothetical scenario where these assumptions about fidaxomicin hold true, at best 50 and at worst 120 patients would need to be predicted to experience a severe outcome and be treated with fidaxomicin in order for one patient to benefit, with the all-cause severity models yielding the best performance. For comparison, prior studies predicted CDI-attributable severity using electronic health record data extracted two days after diagnosis and from a smaller set of manually curated variables, achieving precision values of 0.42 (NNS = 2.4) for the EHR model and 0.17 (NNS = 6.0) for the curated model at the 95th percentile of risk [85, 84]. Pairing the prior EHR-based model with fidaxomicin would yield an NNB of 24. Thus the all-cause and pragmatic OTU-based models outperformed the curated model but not the EHR-based model, although the EHR data were extracted two days after diagnosis while OTUs in this study are from stool samples collected on the day of diagnosis. These estimates represent a proof-of-concept demonstration of the potential value and trade-offs of deploying severity prediction models trained on microbial factors versus EHRs to guide clinicians' treatment decisions.

3.4 Discussion

We trained ML models based on gut microbial communities on the day of CDI diagnosis to predict CDI severity according to four different severity definitions. The purpose of the full datasets was to train the best models possible given the constraints, while using the intersection dataset allows for comparing severity definitions. We found that models predicting pragmatic severity with as much data as available performed best, while models classifying IDSA severity outperformed the all-cause and attributable definitions only with the intersection. Performance dropped substantially when reducing to the intersection dataset for all definitions, likely due to the particularly imbalanced nature of the all-cause and attributable definitions. These results demonstrate the importance of using as many samples as possible when data are sparse and the outcome is low prevalence, as well as the need to incorporate physician's expertise when possible.

Permutation feature importance revealed patterns of important bacteria that concord with the literature. Enrichment of *Enterococcus* and *Lactobacillus* in *C. difficile* infection and severity have been well-documented in prior studies, thus their importance and increase in abundance for severe cases is not surprising [31, 100, 101, 18]. For many of the top OTUs, there is a wide range in importance. Notably, the OTU represented by *Pseudomonas* had wide variance in importance for the full dataset in models predicting attributable severity, with the maximum point more important than any other OTU yet a minimum below zero. However, for the intersection dataset, this OTU was removed due to having near-zero variance. The presence of *Pseudomonas* was thus informative in a small number of patient samples, but not in others, and these samples were lost in the intersection dataset. Overall the abundance data are patchy, as these patients were likely all taking antibiotics for unrelated infections prior to CDI onset. A limitation of permutation importance is that the contribution of each feature is considered in isolation, but members of microbial communities interact and compete with each other, thus these complicated relationships are not well captured by permutation importance.

The full pragmatic severity model performed just as well as a prior EHR-based model trained on the day of diagnosis, demonstrating the potential utility of OTU-based models. In terms of the number needed to screen, the OTU-based pragmatic and all-cause severity models outperformed a prior model of manually curated clinical variables, but not a model trained on EHR data extracted two days after diagnosis. The attributable definition had the worst NNS of all models, despite its clinical relevance. Obtaining EHR data for the dataset in this study would allow a more direct comparison of the performance of models trained on OTUs, EHRs, or both, as well as extracting EHR data on the day of diagnosis rather than two days after.

However, it is not enough for models to perform well to justify deploying them in a clinical setting; benefit over current practices must be shown [34]. Although no known treatment options have been shown to reduce the risk of severe CDI outcomes, fidaxomicin is promising due to its improved time to resolution and reduced recurrence. Despite its increased cost, fidaxomicin is also attractive as a preferential antibiotic option as vancomycin-resistant *Enterococcus* is on the rise and enterococci are known to worsen CDI [24, 25]. We extended our analysis of clinical value to incorporate the number needed to treat for fidaxomicin alongside the predictive models in order to calculate the number needed to benefit. The NNB contextualizes model performance within clinical reality, as it combines both model performance and treatment effectiveness [96]. A more robust analysis of clinical value would further consider the cost of treatment options versus the savings of averting severe outcomes across a range of decision thresholds, as economic disparities are a major barrier to treatment in the

US [97]. Cost-benefit analyses based on clinical trial data have reported that fidaxomicin may be as cost-effective as vancomycin as a treatment for initial CDI cases, largely due to the reduced risk of recurrence [102, 103]. While our analysis of clinical value is only a proof-of-concept, if evidence emerges that new or existing treatments significantly reduce the risk of severe CDI, our results can be incorporated into future considerations of whether to build severity prediction models and what features should be incorporated. In practice, EHR-based models are less costly to deploy than OTU-based models and do not require additional clinical sample collection. However, EHR systems notoriously lack interoperability across hospitals, which inhibits the ability of EHR-based models to generalize to datasets from different hospitals. OTU-based models may be more generalizable across disparate hospital systems than EHR-based models as long as the same sample collection and sequencing protocol is used. Amplicon sequencing is not typically performed for CDI patients, however, routinely profiling the microbial communities of CDI patients could be justified if models that incorporate microbial features were shown to improve patient outcomes.

In all, we found that our models to predict severity from features of the gut microbiome performed moderately well. Our approach enabled us to identify bacteria that contributed to model performance and evaluate how well the state of the gut microbiome can predict several different definitions of CDI severity. Further work is needed to determine whether the performance of OTU-based models is sufficient to justify their deployment in clinical settings, especially as compared to EHR-based models. If and when new evidence emerges of improved treatments to prevent severe CDI outcomes, deploying performant and robust models for clinicians to tailor treatment options may improve patient outcomes and reduce the burden of severe CDI.

3.5 Materials and Methods

3.5.1 Sample collection

This study was approved by the University of Michigan Institutional Review Board. Samples were collected from patients diagnosed with CDI by the University of Michigan Health System from January 2016 through December 2017. Stool samples that had unformed stool consistency were tested for *C. difficile* by the clinical microbiology lab with a two-step algorithm that included detection of *C. difficile* glutamate dehydrogenase and toxins A and B by enzyme immunoassay with reflex to PCR for the *tcdB* gene when results were discordant. 1,517 stool samples were collected from patients diagnosed with a CDI. Leftover stool samples that were sent to the clinical microbiology lab were collected and split into

different aliquots. For 16S sequencing, the aliquot of stool was re-suspended in DNA genotek stabilization buffer and then stored in the -80°C freezer.

3.5.2 16S rRNA gene amplicon sequencing

Samples stored in DNA genotek buffer were thawed from the -80°C, vortexed, and then transferred to a 96-well bead beating plate for DNA extractions. DNA was extracted using the DNeasy Powersoil HTP 96 kit (Qiagen) and an EpMotion 5075 automated pipetting system (Eppendorf). The V4 region of the 16S rRNA gene was amplified with the AccuPrime Pfx DNA polymerase (Thermo Fisher Scientific) using custom barcoded primers, as previously described [63]. Each library preparation plate for sequencing contained a negative control (water) and mock community control (ZymoBIOMICS microbial community DNA standards). The PCR amplicons were normalized (SequalPrep normalization plate kit from Thermo Fisher Scientific), pooled and quantified (KAPA library quantification kit from KAPA Biosystems), and sequenced with the MiSeq system (Illumina).

All sequences were processed with mothur (v1.46) using the MiSeq SOP protocol [70, 63]. Paired sequencing reads were combined and aligned with the SILVA (v132) reference database [59] and taxonomy was assigned with a modified version of the Ribosomal Database Project reference sequences (v16) [60]. Sequences were clustered into *de novo* OTUs with the OptiClust algorithm in mothur [50], resulting in 9,939 OTUs. Samples were then subsampled to 5,000 sequences per sample. Only the first CDI sample per patient was used for subsequent ML analyses such that no patient is represented more than once, resulting in a dataset of 1,277 samples.

3.5.3 Defining CDI severity

We chose to explore four different ways to define CDI cases as severe or not (Figure 3.1).

- **IDSA**: A case is severe if serum creatinine level is greater than or equal to $1.5 mg/dL$ and the white blood cell count is greater than or equal to $15 k/\mu L$ on the day of diagnosis [88].
- **All-cause**: A case is severe if ICU admission, colectomy, or death occurred within 30 days of CDI diagnosis, regardless of the cause of the adverse event.
- **Attributable**: A case is severe if an adverse event of ICU admission, colectomy, or death occurred within 30 days of CDI diagnosis, and the adverse event was determined to be attributable to the CDI by two physicians who reviewed the medical chart [90].
- **Pragmatic**: A case's severity is determined by the attributable definition if it is available, otherwise it is determiend by the all-cause definition.

3.5.4 Model training

Random forest models were used to examine whether OTU data collected on the day of diagnosis could classify CDI cases as severe according to each severity definition. We used the mikropml R package v1.5.0 [104] implemented in a custom version of the mikropml Snakemake workflow [105] for all steps of the machine learning analysis. We have full datasets which use all samples available for each severity definition, and an intersection dataset which consists of only the samples that have all four definitions labelled. The intersection dataset is the most fair for comparing model performance across definitions, while the full dataset allows us to use as much data as possible for model training and evaluation. Datasets were pre-processed with the default options in mikropml to remove features with near-zero variance and scale continuous features from -1 to 1. During pre-processing, 9,757 to 9,760 features were removed due to having near-zero variance, resulting in datasets having 179 to 182 features depending on the severity definition. No features had missing values and no features were perfectly correlated. We randomly split the data into an 80% training and 20% test set and repeated this 100 times, followed by training models with 5-fold cross-validation.

3.5.5 Model evaluation

Model performance was calculated on the held-out test sets using the area under the receiver-operator characteristic curve (AUROC) and the area under the balanced precision-recall curve (AUBPRC). Statistical significance for differences in performance across severity definitions was determined via permutation tests at an alpha level of 0.05. Permutation feature importance was then performed to determine which OTUs contributed most to model performance. We reported OTUs with a significant permutation test at an alpha level of 0.05 in at least 75 of the 100 train/test splits for any severity definition.

Since the severity labels are imbalanced with different frequencies of severity for each definition, we calculated balanced precision, the precision expected if the labels were balanced. The balanced precision and the area under the balanced precision-recall curve (AUBPRC) were calculated with Equations 1 and 7 from Wu *et al.* [91].

3.5.6 Number needed to benefit

For the severity prediction models (which excludes the IDSA definition), we set out to estimate the potential benefit of deploying models in clinical settings. We determined the decision threshold at the 95th percentile of risk for each model, which corresponds to 5% of cases being predicted by the model to experience a severe outcome. At this threshold

we computed the number needed to screen (NNS), which is the reciprocal of precision and represents the number of cases that must be predicted as severe to identify one true positive (Equation 3.1) [92]. The number needed to treat (NNT) is the number of true positive patients that must be treated by an intervention in order for one patient to benefit, and is calculated from the reciprocal of absolute risk reduction in randomized controlled trials (Equation 3.3 and Equation 3.2) [93, 94, 95]. Multiplying the NNS of a model by the NNT of a treatment yields the number needed to benefit (NNB) - the number of patients that must be predicted to have a severe outcome and undergo a treatment to benefit from it (Equation 3.4) [96]. NNB encapsulates the benefit of pairing a predictive model with a treatment in a clinical setting, with lower NNB numbers being better.

3.5.7 Code availability

The complete workflow, code, and supporting files required to reproduce this manuscript with accompanying figures is available on GitHub (https://github.com/SchlossLab/severe-CDI) and archived in Zenodo [106].

The workflow was defined with Snakemake [66] and dependencies were managed with conda environments. Scripts were written in R [67], Python [68], and GNU bash. Additional software and packages used in the creation of this manuscript include cowplot [107], ggtext [74], ggsankey [108], schtools [109], the tidyverse metapackage [71], Quarto, and vegan [110].

3.5.8 Data availability

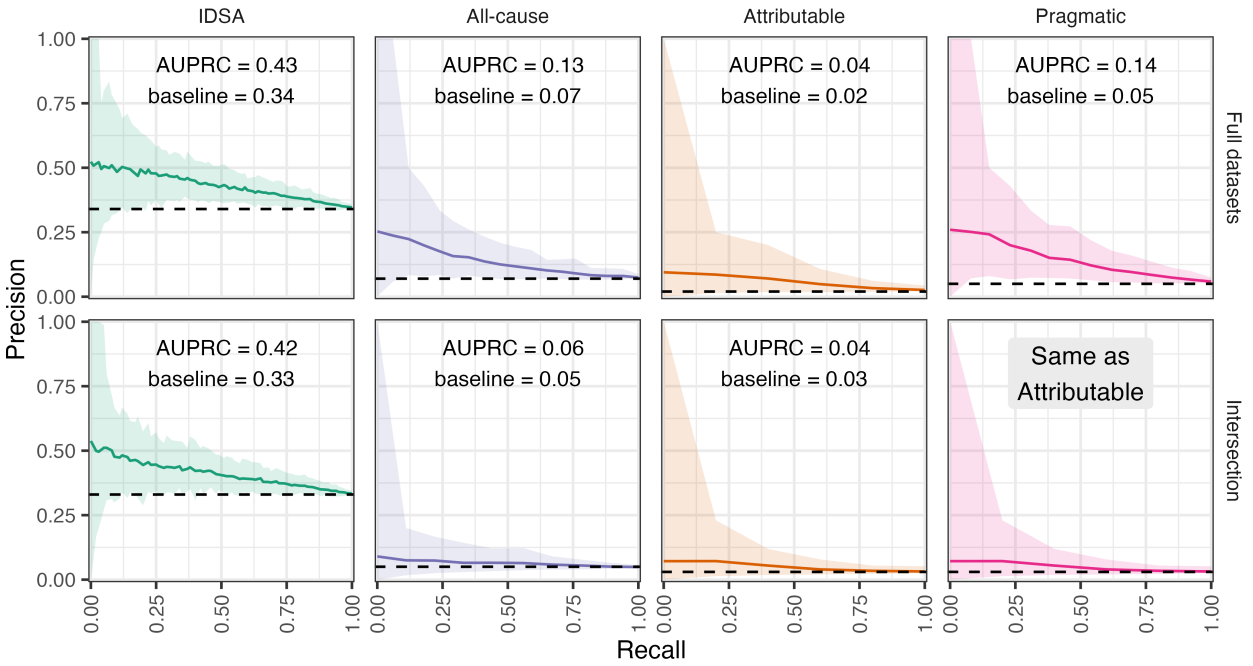
The 16S rRNA sequencing data have been deposited in the National Center for Biotechnology Information Sequence Read Archive (BioProject Accession no. PRJNA729511).

3.6 Acknowledgements

We thank the patients for donating stool samples and the research team members who collected, stored, and processed the samples.

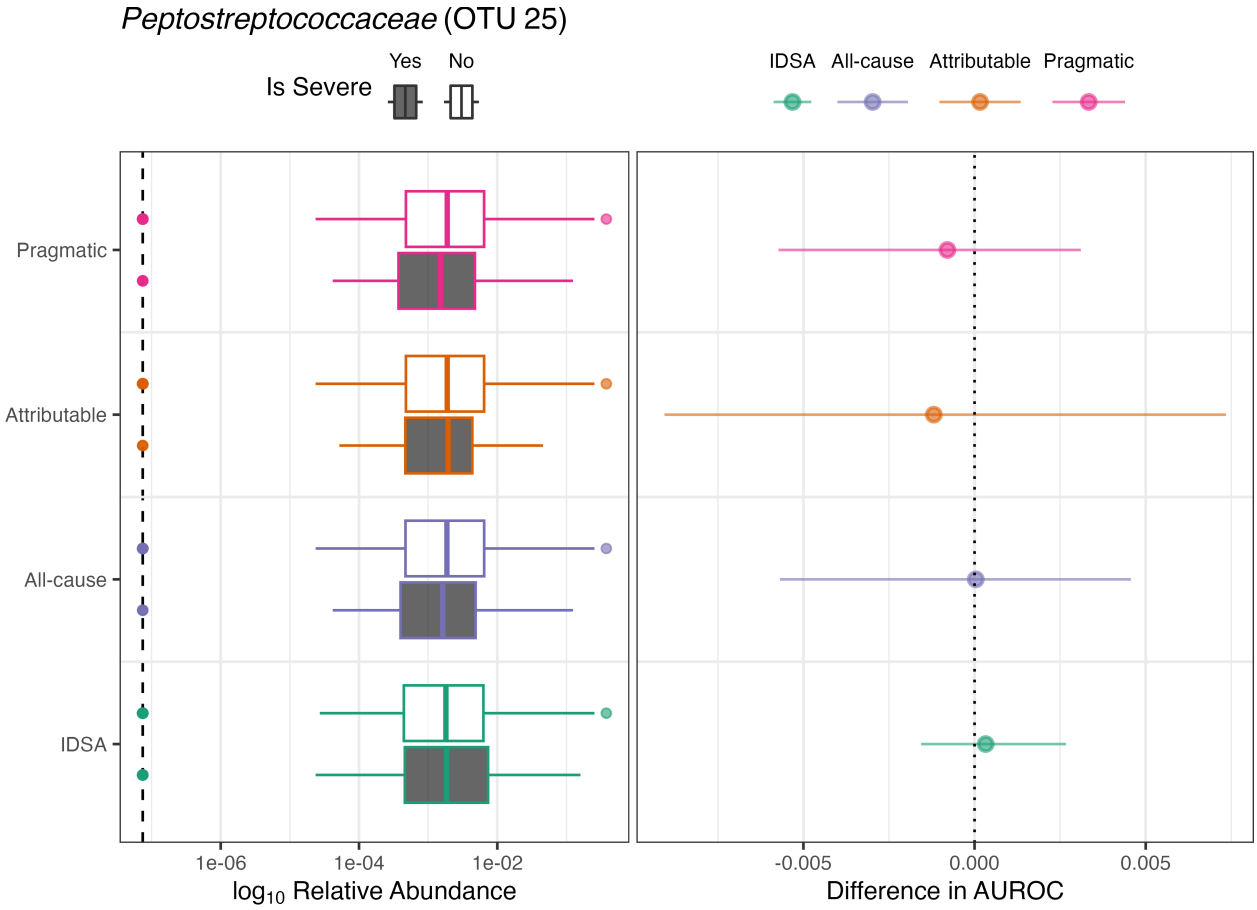
3.7 Supplement

Figure 3.5: Precision-recall curves.



The original precision-recall curves for each model. The horizontal line is the baseline precision, i.e. the proportion of severe cases in the dataset for each severity definition. Since each definition has a different baseline precision, the PRCs cannot be compared directly without balancing the precision (see Figure 3.2).

Figure 3.6: *C. difficile* relative abundance and feature importance.



Of the 45 OTUs belonging to the *Peptostreptococcaceae* family, only one (OTU 25) had abundance values above the limit of detection. **Left**: $\log_{10}$-transformed relative abundance of OTU 25 in the full datasets. The dashed line is the limit of detection. **Right**: Permutation feature importance as measured by AUROC for OTU 25. The point is the mean difference in AUROC and the tails are the 75% confidence interval. The dotted line is a feature importance of zero, meaning the feature is not important.

CHAPTER 4

Democratizing Data Science With Open Curricula and User-Friendly Software Tools

4.1 Preamble

In this chapter, we contributed to the democratization of data science for three key audiences: 1) high school students who wish to learn how to code for a potential career in data science, 2) academics who wish to learn how to code for reproducible research, and 3) scientists who wish to apply machine learning methods toward their areas of study. We developed a curriculum to teach the basics of Python for data science via a Girls Who Code club, a curriculum to teach introductory programming for reproducible research via Software Carpentry workshops, and an R package that implements current best practices in machine learning for novice practitioners. These free and open source contributions make data science education more accessible to a range of audiences and promote responsible use of data science methods.

4.2 Teaching Python for Data Science: Collaborative development of a modular and interactive curriculum

This paper was originally published in 2021 in the Journal of Open Source Education with the following co-authors: Marlena Duda*, Kelly L. Sovacool*, Negar Farzaneh, Vy Kim Nguyen, Sarah E. Haynes, Hayley Falk, Katherine L. Furman, Logan A. Walker, Rucheng Diao, Morgan Oneka, Audrey C. Drotos, Alana Woloshin, Gabrielle A. Dotson, April Kriebel, Lucy Meng, Stephanie N. Thiede, Zena Lapp, and Brooke N. Wolford [111].

*Indicates co-first author

4.2.1 Summary

We are bioinformatics trainees at the University of Michigan who started a local chapter of Girls Who Code to provide a fun and supportive environment for high school women to learn the power of coding. Our goal was to cover basic coding topics and data science concepts through live coding and hands-on practice. However, we could not find a resource that exactly met our needs. Therefore, over the past three years, we have developed a curriculum and instructional format using Jupyter notebooks to effectively teach introductory Python for data science. This method, inspired by The Carpentries organization, uses bite-sized lessons followed by independent practice time to reinforce coding concepts, and culminates in a data science capstone project using real-world data. We believe our open curriculum is a valuable resource to the wider education community and hope that educators will use and improve our lessons, practice problems, and teaching best practices. Anyone can contribute to our Open Educational Resources on GitHub.

4.2.2 Statement of Need

As women bioinformatics trainees at the University of Michigan (U-M), we experience the gender gap in our field first-hand. During the 1974-1975 academic year, women achieved 18.9% of total Bachelor's degrees in computer and information sciences in the US [112]. By 1983-1984 this peaked at 37.1%, but fell to 17.6% by 2010-2011. We also see this national trend in the training of the next generation of Bioinformaticians at Michigan Medicine. Since accepting its first students in 2001, the U-M Bioinformatics Graduate Program has graduated 66 male and 22 female doctorates as of 2019. This disparity begins at the applicant level; during 2016-2019 the average percentage of females applying directly to the Bioinformatics PhD program was 35.2%, and the average percentage of female applicants listing Bioinformatics as first or second choice in the Program in Biomedical Sciences, U-M's biomedical PhD umbrella program was 41%.

Previous research on women's educational experiences in science, technology, engineering, and mathematics (STEM) have produced various explanations for persistent gender disparities [113]. One explanation is that women often experience stereotype threats that negatively influence their math and science performance and deter them from pursuing STEM as a career [114]. The majority of our organization's founding graduate students (all women) began coding in our undergraduate careers or later. We wanted to provide a safe environment for local high school women to develop confidence in themselves and their computational skills before college, and be exposed to successful women role models in STEM to counter negative stereotypes.

Girls Who Code, a national organization whose mission is to close the gender gap in technology [115], was founded in 2012. Because of our personal experiences and the paucity of women in our field [112, 116], we began a Girls Who Code student organization at the University of Michigan in 2017. For the past four academic years we have registered annually as a recognized Girls Who Code Club because the national organization provides name recognition, curriculum resources, guidance for a Capstone Impact Project, and a framework for launching a coding club. Participants in the Club attend weekly meetings at the University of Michigan (when the club is run in person rather than virtually), and are thus largely high school women from the Ann Arbor area. In 2019 we launched our own summer program, the Data Science Summer Experience. When held in person, the Summer Experience is hosted in Detroit to provide the opportunity for high school women outside of Ann Arbor to learn coding skills in an inclusive environment.

The national Girls Who Code organization provides a curriculum that teaches website and application development through programming languages like HTML and Java; however, our biomedical science graduate students generally have limited experience with these languages and with web development. In contrast, many of us have extensive experience performing data science using the Python programming language. Data Scientist was rated the #1 job in America by Glassdoor in 2016-2019, #3 in 2020, and #2 in 2021 [117]. Furthermore, Python is the most popular programming language according to the PYPL PopularitY of Programming Language Index [118]. Therefore, we believe career exploration in data science using the Python programming language will optimally prepare our learners for careers that provide financial stability and upward economic mobility. By leveraging the data science expertise of our Club facilitators (hereafter termed instructors), we created a specialized curriculum focused on computational data science in the Python programming language.

Girls Who Code encourages participants to learn programming skills while working on an Impact Project website or application throughout the Club [119]. We created an open source Data Science curriculum that teaches the requisite Python and statistics skills to complete a Capstone Project, where learners explore, analyze, and present a data set of their choosing. Using this curriculum, we employ participatory live coding, where learners type and run code along with the instructor in real time. Using paired activities, our curriculum follows the "I do, we do, you do" didactic paradigm [120]. We provide open source resources for both in-person and virtual versions of our curriculum, including videos corresponding to each lesson. While we developed this curriculum for our Girls Who Code Club and Summer Experience, we believe that it can be widely used for teaching introductory coding for data science.

4.2.3 Collaborative Curriculum Development

We assembled a team of volunteers involved in our club to develop a custom curriculum to teach introductory Python for data science. We chose the content based on what our learners would need to learn to complete a small data analysis project and communicate their findings to their peers. We divided the content by topic into Jupyter notebooks for each lesson, with each lesson taking approximately 15-20 minutes to teach via live coding. Every lesson has a corresponding practice notebook with additional exercises on the same content taught in the lesson, but using different data or variables. We used a similar development workflow as the U-M Carpentries curriculum [121]. Briefly, we hosted the curriculum notebooks in a public GitHub repository to facilitate collaborative development and peer review using pull requests. In the initial curriculum drafting phase, developers were assigned lesson and practice notebooks to write. Once the draft of a lesson was completed, the writer opened a pull request and asked for review from a different developer. The reviewer then provided feedback and approved the pull request to be merged into the main branch after the writer made any requested changes. This way, more than one person viewed each notebook before it could be incorporated into the public curriculum, which reduced mistakes and ensured higher quality content. While teaching from the curriculum at the first Data Science Summer Experience, instructors took notes on their experience and made revisions afterward. Maintainers continue to monitor the repository and resolve issues as they arise.

Following the onset of the COVID-19 pandemic, we quickly pivoted our club to a virtual format. In preparation for the 2020 Summer Experience, we switched to a flipped classroom style following feedback from our club participants that it was too difficult to follow along live coding via Zoom (see Instructional Design).

4.2.4 Curriculum

Our curriculum was designed for high school students with no prior coding experience who are interested in learning Python programming for data science. However, this course material would be useful for anyone interested in teaching or learning basic programming for data analysis.

4.2.4.1 Learning Objectives

The learning objectives of this curriculum are:

1. Write code in Python with correct syntax and following best practices.

Figure 4.1: Lesson modules. All Jupyter notebooks are available on GitHub (https://github.com/GWC-DCMB/curriculum-notebooks).

2. Implement fundamental programming concepts when presented with a programmatic problem set.
3. Apply data analysis to real world data to answer scientific questions.
4. Create informative summary statistics and data visualizations in Python.

These skills provide a solid foundation for basic data analysis in Python. Participation in our program exposes learners to the many ways coding and data science can be impactful across many disciplines.

4.2.4.2 Course Content

Our curriculum design consists of 27 lessons broken up into 5 modules that cover Jupyter notebook setup, Python coding fundamentals, use of essential data science packages including pandas and numpy, basic statistical analyses, and plotting using seaborn and matplotlib (Figure 4.1) [75, 122, 123]. Each lesson consists of a lesson notebook and a practice notebook containing similar exercises for the learner to complete on their own following the lesson.

Each lesson builds on those before it, beginning with relevant content reminders from the previous lessons and ending with a concise summary of the skills presented within. As they progress through the curriculum, the learners begin simultaneously working on a data

science project using a real world dataset of their choosing. While more time is dedicated to lessons early in the program, the formal curriculum tapers off until the learners are solely applying their skills to the data science project. Through this Capstone Project, learners gain practical experience with each skill as they learn it in the lessons; including importing and cleaning data, data visualization, and basic statistical analyses.

4.2.5 Instructional Design

We modeled our instructional design in the style of Software Carpentry [45].

1. Each lesson begins with a recapping of the relevant core skills presented in the previous lessons.
2. All lessons are designed to be taught via 15-minute live-coding sessions. This method is used by The Carpentries and is demonstrated to be an effective method that engages learners [45, 124] since learners must actively engage with the material and deal with errors and bugs as they arise.
3. Each lesson ends with a summary of core skills presented within the material.
4. Each short lesson is also accompanied by a subsequent 10-minute independent practice, providing further opportunity for practical experience implementing the coding skill at hand and testing learners' understanding of the content.

To better facilitate virtual instruction during the COVID-19 pandemic, we switched to a flipped classroom. Prior to meeting, learners watch videos of instructors explaining the material through "live" coding and code along in the lesson notebook with while watching the video. Each video shows the Jupyter notebook alongside the instructor themselves teaching. Learners then complete a practice notebook corresponding to the lesson. During the virtual meeting time, instructors answer questions and review the core concepts in the practice exercises. This virtual format is especially beneficial because it 1) allows learners to learn at their own pace, and 2) enables dissemination of our curriculum to a wider audience interested in learning introductory Python programming for data science.

For both in-person and virtual instruction, once learners have completed the Fundamentals module and reach the Data Science Essentials module they begin simultaneous work on their data science projects. Projects are completed in a pair programming style, where partners take turns assuming the "driver" (i.e. the typer) and "navigator" (i.e. the helper) roles [125]. Switching off in this way helps both partners assume equal responsibility for the project workload, but more importantly it enables improved knowledge transfer through peer-to-peer learning. The culmination of the project is a presentation to peers, instructors, and family members. Through this process learners gain hands-on experience coding,

cleaning data, performing statistical analyses, creating informative data visualizations, and communicating their results to others.

In addition to our coding curriculum, another key component of our programming is hosting women guest speakers from diverse fields across academia and industry. Our guest speakers come to discuss the journey they have taken to their career paths as well as how they utilize programming and data science in their jobs. These varied perspectives are extremely valuable to our learners as they provide several practical examples of programming careers in the real world, and expose them to successful women in STEM.

4.2.5.1 Experience of Use

We have used this curriculum to teach the Data Science Summer Experience and Girls Who Code Club in person in 2019 and virtually in 2020-2021. For both in-person and virtual instances, we had several instructors present at each session to answer questions and help learners debug. Furthermore, one or two instructors were assigned to each project group to help learners define data analysis questions, develop and execute a data analysis plan, visualize and communicate their findings, and troubleshoot coding problems. Projects have ranged from investigating exoplanets to studying the genomics of psoriasis.

We credit the success of our curriculum not only to the skill of the instructors, but also to the way we organized and executed the lessons and project:

1. The instructors and learners used Google Colaboratory (Colab) to write and execute code in Jupyter notebooks. We chose this option because learners do not have to install any programs to use Google Colab and can easily open and edit the Jupyter notebooks from GitHub. When meeting in person, most learners use Google Chromebooks which have limited programming capabilities, but easy use of a web browser.
2. Assigning instructors to groups allowed learners to build a more personal connection with their instructors, making them feel more comfortable asking questions.
3. Group projects were performed using pair programming to allow learners to collaborate and learn from each other.
4. We used the "sticky note" system from The Carpentries by which learners can ask for help by putting up a colored sticky note (or a Zoom emoji in the case of virtual meetings) [126].
5. We exposed the learners to different aspects of data science by bringing in women guest speakers from academics and industry. This allowed them to better put what they were learning into context, think about how they might use the skills they were learning in potential future careers, and exposed them to successful women in STEM.

Figure 4.2: Post-survey responses. Learners were asked if they felt that their skills in Python programming, problem solving, critical thinking, and collaboration had improved.

Learner experiences We surveyed learners anonymously after each Club and Summer Experience and found that most felt that their skills in Python programming, problem solving, critical thinking, and collaboration had improved (Figure 4.2). Furthermore, on a 10 question skills assessment during the 2019-2020 instance of the Club, the average increase in correct answers between the first meeting and the last meeting was 4.2 with a standard deviation of 2.8 (N=5 respondents). We also surveyed Club and Summer Experience alumni and found that 75% (N=20) want to pursue a STEM career. 62% (N=21) are still coding. On a 5-point scale from 'Strongly Disagree' to 'Strongly Agree,' the average answer for 'My participation in GWC impacted my career aspirations' is 4 (s.d.=0.9), with 4.5 (s.d.=0.6) for 'Participating in GWC made me feel more confident in analyzing data' and 3.9 (s.d.=1) for 'Participating in GWC made me more confident in myself.'

Overwhelmingly, learners' favorite parts of the program are the guest speakers and the project. These aspects of our curriculum expose them to new fields and allow them to apply their newfound coding skills to asking an interesting question. A 2021 Club learner shared, "I plan to go to college for Computer Science and get a robotics minor when my college

offers it. GWC has inspired me to consider pursuing a Masters or PhD in CS as well as take some electives in Data Science." Five of our 86 alumni have gone on to perform research with U-M faculty members, with one presenting her work at an international conference. In fact, about a third of participants claim that they are now more interested in pursuing a career in computer or data science compared to before their Girls Who Code experience.

4.2.6 Acknowledgements

We would like to acknowledge our faculty co-sponsors Maureen Sartor & Cristina Mitrea. We appreciate the continued support of U-M DCMB staff and faculty including Julia Eussen, Mary Freer, Linda Peasley, Jane Wiesner, Brian Athey, and Margit Burmeister. We are grateful for the resources provided by the national Girls Who Code organization.

Our programming is made possible by the dedication of past and present Executive Committee members, Club and Summer Experience Facilitators, and Capstone Project mentors including Shweta Ramdas, Alex Weber, Arushi Varshney, Sophie Hoffman, Hojae Lee, Ruma Deb, Saige Rutherford, Michelle McNulty, Bailey Peck, Chloe Whicker, Carolina Rojas Ramirez, Verity Sturm, Zoe Drasner, Sarah Latto, Emily Roberts, Angel Chu, Vivek Rai, Hillary Miller, Ashton Baker, Murchtricia Jones, Lauren Jepsen, Aubrey Annis, Awanti Sambarey, Mengtong Hu, Maribel Okiye, Yingxiao Zhang, and Neslihan Bisgin.

We are grateful for the funding, assistance, and other support provided to our student organization from the following sponsors: the U-M Department of Computational Medicine and Bioinformatics, the U-M Department of Biostatistics, the U-M Department of Statistics, the U-M Office of Graduate and Postdoctoral Studies, the U-M Endowment in Basic Sciences, the U-M Detroit Center, the U-M Life Sciences Institute, the U-M Office of Research, the Michigan Council of Women in Technology Foundation, DELL Technologies, Cisco Systems, Zingerman's Delicatessen, the Girls Who Code Support Fund, and anonymous donations from Giving Blue Day 2019.

We also thank the learners who have participated in our Club and Summer Experience events.

4.2.7 Funding

MD, ACD, ZL, and BNW received support from the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE 1256260. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

MD, KLS, NF, and VKN received support from the NIH Training Program in Bioinformatics (T32 GM070449). NF was supported by the National Institute of Health (NIH) Ruth L. Kirschstein National Research Service Award (NRSA) Individual Predoctoral Fellowship Program (F31 LM012946-01). VKN was supported by a NIH Research Project Grant on Breast Cancer Disparities (RO1-ES028802) and the CDC through the National Institute for Occupational Safety and Health (NIOSH) Pilot Project Research Training Program (T42-OH008455). KLF received support from The University of Michigan NIDA Training Program in Neuroscience (T32-DA7281) and from the NIH Early Stage Training in the Neurosciences Training Grant (T32-NS076401). MO received support from the Advanced Proteome Informatics of Cancer Training Grant (T32 CA140044). SNT was supported by the Molecular Mechanisms in Microbial Pathogenesis training grant (NIH T32 AI007528). ZL and BNW received support from the NIH Training Program in Genomic Science (T32-HG000040-22).

4.2.8 Author Contributions

MD, KLS, ZL, and BNW wrote the initial draft of the manuscript. All authors contributed to the curriculum and reviewed the manuscript.

4.2.9 Conflicts of Interest

None.

4.3 Developing and deploying an integrated workshop curriculum teaching computational skills for reproducible research

This paper was originally published in 2022 in the Journal of Open Source Education with the following co-authors: Zena Lapp*, Kelly L. Sovacool*, Nick Lesniak, Dana King, Catherine Barnier, Matthew Flickinger, Jule Krüger, Courtney R. Armour, Maya M. Lapp, Jason Tallant, Rucheng Diao, Morgan Oneka, Sarah Tomkovich, Jacqueline Moltzau Anderson, Sarah K. Lucas, and Patrick D. Schloss [121].

*Indicates co-first author

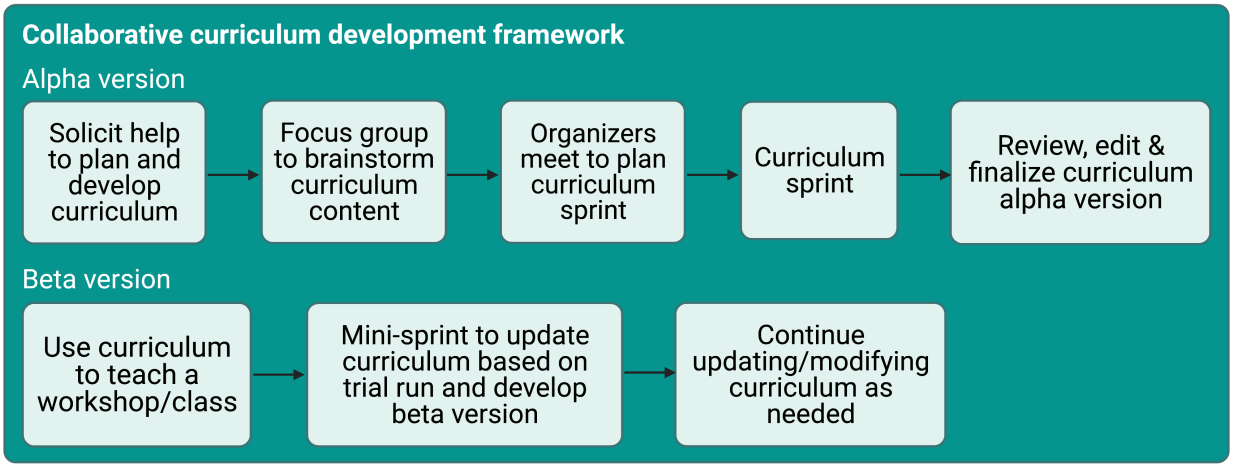
4.3.1 Summary

Inspired by well-established material and pedagogy provided by The Carpentries [45], we developed a two-day workshop curriculum that teaches introductory R programming for managing, analyzing, plotting and reporting data using packages from the tidyverse [71], the Unix shell, version control with git, and GitHub. While the official Software Carpentry curriculum is comprehensive, we found that it contains too much content for a two-day workshop. We also felt that the independent nature of the lessons left learners confused about how to integrate the newly acquired programming skills in their own work. Thus, we developed a new curriculum that aims to teach novices how to implement reproducible research principles in their own data analysis. The curriculum integrates live coding lessons with individual-level and group-based practice exercises, and also serves as a succinct resource that learners can reference both during and after the workshop. Moreover, it lowers the entry barrier for new instructors as they do not have to develop their own teaching materials or sift through extensive content. We developed this curriculum during a two-day sprint, successfully used it to host a two-day virtual workshop with almost 40 participants, and updated the material based on instructor and learner feedback. We hope that our new curriculum will prove useful to future instructors interested in teaching workshops with similar learning objectives.

4.3.2 Statement of Need

For the past five years, the University of Michigan instance of The Carpentries has taught workshops using versions of curriculum originally created by The Carpentries organization. In that time, our instructors found several advantages and disadvantages to using the original Software Carpentry curriculum. Some of the advantages were that any programming language lesson (e.g., R or Python) could be paired with lessons on the Unix shell and version control, lessons had been refined by many contributors over the years and taught at workshops around the world, and the instructional design demonstrated good pedagogy for teaching novice data science practitioners. However, The Carpentries materials have evolved from lesson plans to reference materials, and thus there was too much content for the time available during a two-day workshop. As a result, workshops taught with this material were inconsistent depending on who was teaching, and new instructors faced an overwhelming amount of work to prepare for their first workshop. Furthermore, the modular nature of the curriculum meant that each lesson was independent from the others, so it was not apparent to learners how all of the skills could be integrated for the purpose of a reproducible research project.

Figure 4.3: Curriculum development framework



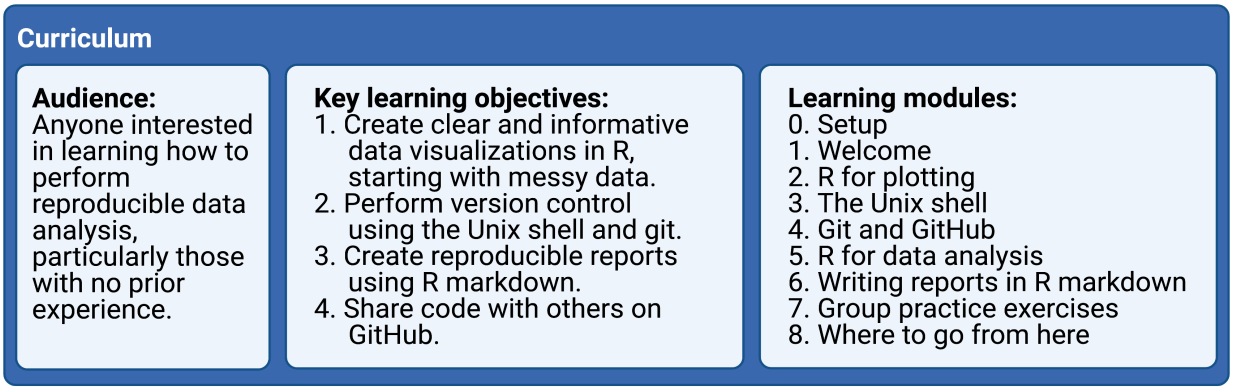
Given these constraints, we sought to create a new curriculum that would allow us to teach computational skills in an integrated manner, demonstrate the reproducible research workflows we use in our own work, deliver an appropriate and consistent amount of content, and reduce the burden for new instructors to get involved, all while maintaining the same inclusive pedagogy that has been refined by The Carpentries organization.

4.3.3 Collaborative Curriculum Development

We drew on the expertise of The Carpentries community at the University of Michigan to develop a custom curriculum that would meet our goals (Figure 4.3). To start, we organized a two-day sprint, where members of our community worked collaboratively to create an initial draft of the content. During the sprint, we met virtually to discuss our goals, then broke up into teams to work on individual lessons before coming back together to review our progress. We hosted the curriculum in a public GitHub repository (https://github.com/umcarpentries/intro-curriculum-r) to facilitate collaborative work and peer review using issues, branches, and pull requests. Under this model, a team member created or edited content in a new branch to resolve an issue, then created a pull request and asked for review from another team member, who finally merged the changes into the default branch. GitHub pages automatically uses the default branch to build a website that allows us to host the polished curriculum (https://umcarpentries.org/intro-curriculum-r/). Our collaborative model ensured that at least two pairs of eyes viewed any changes before they could be included in the curriculum. This strategy helped us reduce mistakes and create better quality content.

Following the sprint, contributors finalized edits and continued to review each others' pull

Figure 4.4: Curriculum overview



requests to complete the alpha version of our curriculum. Next, we hosted a workshop for instructors to pilot the curriculum. We collected feedback from the learners and instructors at the end of the pilot workshop and then held a smaller half-day sprint to revise the curriculum based on the feedback. Currently, our community members are continuously able to create issues, make edits, and review pull requests to keep refining the curriculum for future use. We are planning more workshops with new instructors who were not involved in the original curriculum development to gather their feedback.

4.3.4 Curriculum

Our curriculum is tailored to people with no prior coding experience who want to learn how to use R programming for data analysis, visualization and the reporting of results (Figure 4.4). Not only do we aim to teach our learners the basics of performing empirical data analysis, we also seek to provide a rigorous framework for adhering to reproducible research principles that enable researchers to easily share their empirical work with others.

4.3.4.1 Learning Objectives

The key learning objectives for our curriculum are:

1. Create clear and informative data visualizations in R, starting with messy data.
2. Perform version control using the Unix shell and git.
3. Create reproducible reports using R Markdown.
4. Share code with others on GitHub.

We believe these skills provide learners with a solid foundation from which they can teach themselves any additional coding skills for future use.

4.3.4.2 Course Content

Our curriculum consists of nine modules that cover software setup, data analysis and visualization in R, version control, sharing code, and writing reproducible reports (see below for more details). The R programming lessons take a "tidyverse first" approach [127] to effectively and efficiently teach learners powerful tools for plotting and data analysis. We also set an overall goal for the workshop to make the content substantively interesting and relatable to a wide audience regardless of their original academic discipline or professional practice. Specifically, we task our learners with producing a fictitious report to the United Nations that examines the relationship between gross domestic product (GDP), life expectancy, and CO_2 emissions. The nine curriculum modules are:

0. Setup
1. Welcome
2. R for plotting (uses the tidyverse R packages [71])
3. The Unix shell
4. Git and GitHub
5. R for data analysis (uses the tidyverse R packages [71])
6. Writing reports in R Markdown (uses the rmarkdown R package [72])
7. Group practice exercises
8. Where to go from here

Each lesson builds on the previous ones. The Unix shell, git, and GitHub are introduced using the files generated in the R for plotting lesson. The lesson content for subsequent modules is then intermittently committed and pushed to GitHub. The 'Writing reports in R Markdown' lesson combines all of the skills learned previously to produce a report that one could share with the United Nations. Next, learners put everything they have learned into practice by forming small groups and working on practice problems that cover the entire course content ("Integrating it all together: Paired exercise"). The workshop completes with a short module recapping everything that the curriculum covered as well as offering suggestions on how learners can continue to get help and keep learning once the workshop ends.

4.3.4.3 Instructional Design

Our modules and teaching suggestions are developed in the style of Software Carpentry:

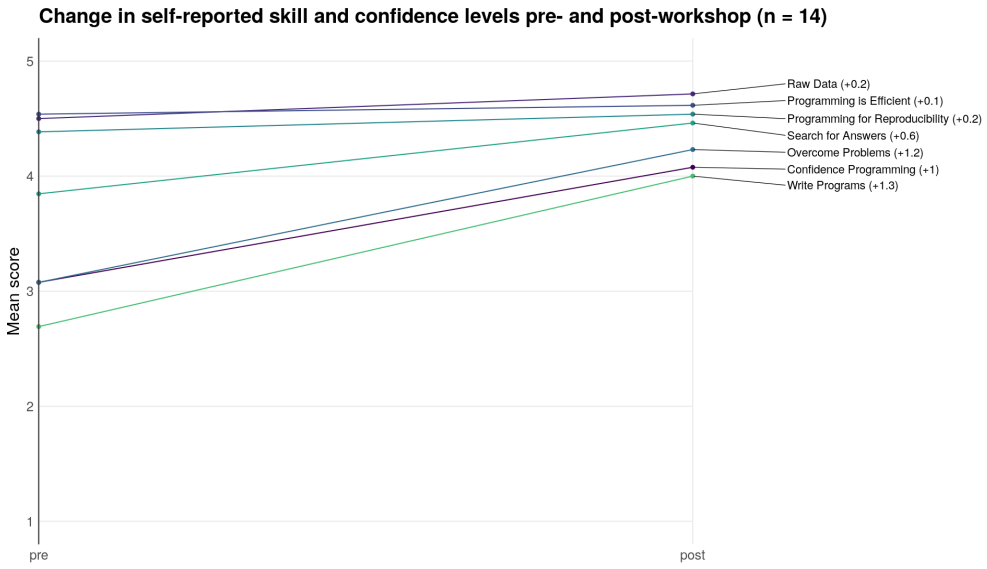
1. Each module contains learning objectives at the beginning of each lesson and a summary of key points at the end.

2. The five core modules (2 to 6) are designed to be taught via live coding of the content to learners. This is a central feature of Carpentries lessons, and we believe it is a great way to learn how to program. It requires learners to follow along and encounter errors that they must debug along the way, fostering additional questions about the course content. It also leads to instructors making mistakes and then demonstrating how to deal with them in an ad hoc and iterative manner.
3. We incorporate formative assessments in the form of short practice exercises throughout each lesson such that learners can practice what they have learned, while instructors can gauge learner understanding of the material.
4. We use the "sticky note" system for formative assessment, where learners indicate their progress on exercises and request help by using different colored sticky notes [126, 128]. At virtual workshops, we use Zoom reaction icons as virtual sticky notes, with the red X reaction to ask for help and the green checkmark to indicate that an exercise was successfully completed.
5. We have several helpers attend each workshop to address learner questions and technical issues.

We also incorporated a few additional key components into the curriculum:

1. Each lesson built off of previous lessons, with the goal of creating a final report that can be shared with others.
2. We structured the curriculum such that it could be taught through an in-person or virtual workshop. Virtual workshops are sometimes necessary, as during the COVID-19 pandemic, but are also useful to allow people from a variety of geographic locations to instruct and attend.
3. We not only required learners to install all software before the workshop (as The Carpentries also requires), but also asked them to run an example script that tests whether everything is installed correctly. To attend the workshop, learners were required to send screenshots of the script output to the workshop lead in advance. We withheld the login details for the workshop until we received the screenshot. This ensured that any installation issues could be addressed before the workshop began.
4. An extensive small group practice module towards the end of the workshop allowed learners to more independently practice the skills they have learned.
5. The workshop concluded with a recap of what was covered and resources available for learners to continue learning and getting help as their skills develop.

Figure 4.5: Pre- and post-workshop survey results



4.3.4.4 Pilot Workshop

We piloted our curriculum during a virtual two-day Software Carpentry workshop. In line with The Carpentries recommendations [129], we had four instructors and six helpers at the workshop to assist with learner questions and technical issues. We had thirty-nine learners of various skill levels from several different countries, all of whom provided very positive reviews of the workshop. To assess the effectiveness of the workshop, learners were asked to complete a pre- and post-workshop survey administered by the Carpentries. By the end of the workshop, learners on average felt more confident writing programs, using programming to work with data, overcoming problems while programming, and searching for answers to technical questions online (n = 14 survey respondents; see Figure 4.5). All attendees who filled out the post-workshop survey (n = 19) would recommend the workshop to others.

Virtual Workshop Reflection We credit the success of our first virtual workshop in large part due to the curriculum structure and content, as well as the instructors and helpers involved. However, we also believe that the following helped make the workshop as smooth as possible:

1. We suggested that learners have Zoom and RStudio (or the Unix shell) open side-by-side on their computer to minimize toggling between different windows [130].
2. We used Slack for communication among instructors and helpers, as well as between

helpers and learners. Learners asked questions in a group Slack channel where helpers could respond. This allowed us to address the vast majority of learner questions and bugs quickly, clearly, and efficiently without disrupting the lesson or moving the learner to a Zoom breakout room. Furthermore, Slack worked much better than the Zoom chat as questions could be answered in threads, were preserved and visible to all learners regardless of whether they were connected to Zoom at the time, and didn't get lost as easily.

3. Whenever a learner needed more help than was possible on Slack, a helper and the learner entered a Zoom breakout room together to troubleshoot. However, we tried to minimize this option as much as possible to prevent the learner from missing content covered in the main room.

4.3.5 Acknowledgements

We thank The Carpentries organization for providing instructor training, workshop protocols, and the open-source Software Carpentry curriculum upon which this curriculum is based. We also thank them for allowing us to use the pre- and post-workshop survey results in this manuscript. The Carpentries is a fiscally sponsored project of Community Initiatives, a registered 501(c)3 non-profit organisation based in California, USA.

We are grateful to Victoria Alden and Scott Martin for assisting us in organizing and advertising our pilot workshop. We thank Shelly Johnson for volunteering as a helper at the workshop and contributing to the setup instructions. We also thank Bennet Fauber for contributing to the setup instructions.

We thank the learners who participated in the workshop, provided feedback, and completed the surveys.

4.3.6 Funding

Salary support for PDS came from NIH grants R01CA215574 and U01AI124255. KLS received support from the NIH Training Program in Bioinformatics (T32 GM070449). ZL received support from the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE 1256260. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

4.3.7 Author Contributions

ZL and KLS contributed equally. ZL is first among the co-first authors because KLS threatened to reject all pull requests where ZL put KLS first. :)

PDS supervised the project. ZL and KLS organized the initial sprint, led the development of the curriculum, and drafted the manuscript. ZL, KLS, JK, and MML instructed at the first pilot workshop while CRA, JMA, ST, SKL, and CB assisted learners. All authors contributed to the development of the curriculum.

4.3.8 Conflicts of Interest

None.

4.4 mikropml: User-Friendly R Package for Supervised Machine Learning Pipelines

This paper was originally published in 2021 in the Journal of Open Source Software with the following co-authors: Begüm D. Topçuoğlu*, Zena Lapp*, Kelly L. Sovacool*, Evan Snitkin, Jenna Wiens, and Patrick D. Schloss [104].

*Indicates co-first author

4.4.1 Summary

Machine learning (ML) for classification and prediction based on a set of features is used to make decisions in healthcare, economics, criminal justice and more. However, implementing an ML pipeline including preprocessing, model selection, and evaluation can be time-consuming, confusing, and difficult. Here, we present `mikropml` (pronounced "meek-ROPE em el"), an easy-to-use R package that implements ML pipelines using regression, support vector machines, decision trees, random forest, or gradient-boosted trees. The package is available on GitHub, CRAN, and conda.

4.4.2 Statement of need

Most applications of machine learning (ML) require reproducible steps for data preprocessing, cross-validation, testing, model evaluation, and often interpretation of why the model makes particular predictions. Performing these steps is important, as failure to implement them can result in incorrect and misleading results [131, 34].

Supervised ML is widely used to recognize patterns in large datasets and to make predictions about outcomes of interest. Several packages including `caret` [132] and `tidymodels` [133] in R, `scikitlearn` [134] in Python, and the H2O `autoML` platform [135] allow scientists to train ML models with a variety of algorithms. While these packages provide the tools necessary for each ML step, they do not implement a complete ML pipeline according to good practices in the literature. This makes it difficult for practitioners new to ML to easily begin to perform ML analyses.

To enable a broader range of researchers to apply ML to their problem domains, we created `mikropml`, an easy-to-use R package [67] that implements the ML pipeline created by Topçuoğlu *et al.* [32] in a single function that returns a trained model, model performance metrics and feature importance. `mikropml` leverages the `caret` package to support several ML algorithms: linear regression, logistic regression, support vector machines with a radial basis kernel, decision trees, random forest, and gradient boosted trees. It incorporates good practices in ML training, testing, and model evaluation [32, 131]. Furthermore, it provides data preprocessing steps based on the FIDDLE (FlexIble Data-Driven pipeLinE) framework outlined in Tang *et al.* [136] and post-training permutation importance steps to estimate the importance of each feature in the models trained [137, 138].

`mikropml` can be used as a starting point in the application of ML to datasets from many different fields. It has already been applied to microbiome data to categorize patients with colorectal cancer [32], to identify differences in genomic and clinical features associated with bacterial infections [139], and to predict gender-based biases in academic publishing [140].

4.4.3 mikropml package

The `mikropml` package includes functionality to preprocess the data, train ML models, evaluate model performance, and quantify feature importance (Figure 4.6). We also provide vignettes and an example Snakemake workflow [66] to showcase how to run an ideal ML pipeline with multiple different train/test data splits. The results can be visualized using helper functions that use `ggplot2` [141].

While mikropml allows users to get started quickly and facilitates reproducibility, it is not a replacement for understanding the ML workflow which is still necessary when interpreting results [142]. To facilitate understanding and enable one to tailor the code to their application, we have heavily commented the code and have provided supporting documentation which can be read online.

4.4.3.1 Preprocessing data

We provide the function `preprocess_data()` to preprocess features using several different functions from the `caret` package. `preprocess_data()` takes continuous and categorical data, re-factors categorical data into binary features, and provides options to normalize continuous data, remove features with near-zero variance, and keep only one instance of perfectly correlated features. We set the default options based on those implemented in FIDDLE [136]. More details on how to use `preprocess_data()` can be found in the accompanying vignette.

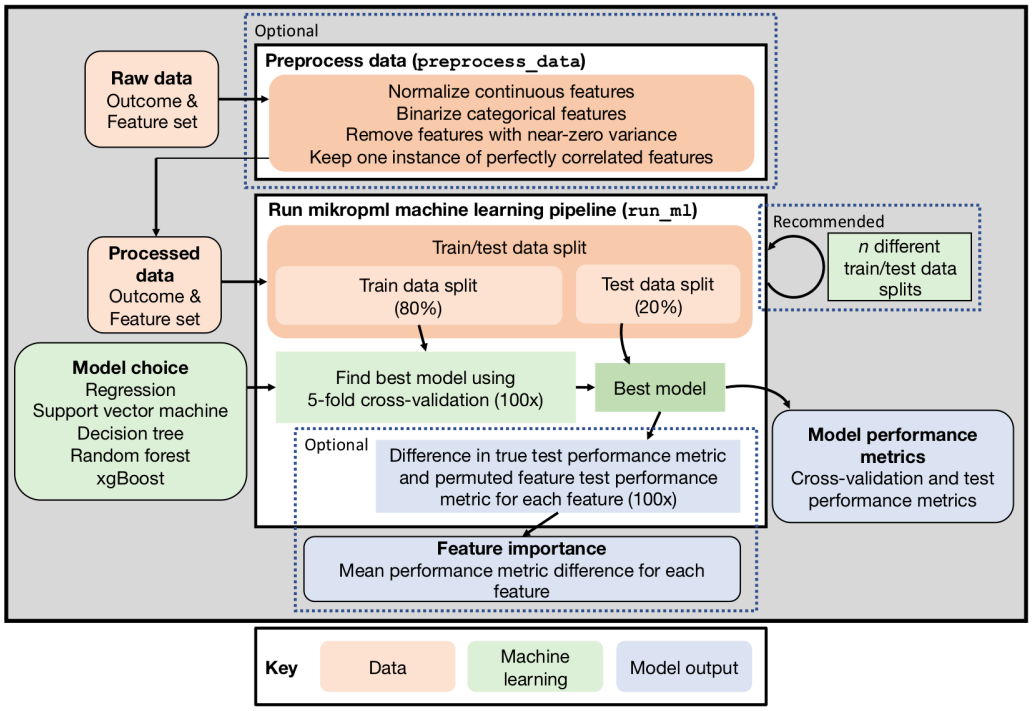
4.4.3.2 Running ML

The main function in mikropml, `run_ml()`, minimally takes in the model choice and a data frame with an outcome column and feature columns. For model choice, `mikropml` currently supports logistic and linear regression [`glmnet`: 143], support vector machines with a radial basis kernel [`kernlab`: 144], decision trees [`rpart`: 145], random forest [`randomForest`: 146], and gradient-boosted trees [`xgboost`: 147]. `run_ml()` randomly splits the data into train and test sets while maintaining the distribution of the outcomes found in the full dataset. It also provides the option to split the data into train and test sets based on categorical variables (e.g. batch, geographic location, etc.). `mikropml` uses the `caret` package [132] to train and evaluate the models, and optionally quantifies feature importance. The output includes the best model built based on tuning hyperparameters in an internal and repeated cross-validation step, model evaluation metrics, and optional feature importances. Feature importances are calculated using a permutation test, which breaks the relationship between the feature and the true outcome in the test data, and measures the change in model performance. This provides an intuitive metric of how individual features influence model performance and is comparable across model types, which is particularly useful for model interpretation [32]. Our introductory vignette contains a comprehensive tutorial on how to use `run_ml()`.

4.4.3.3 Ideal workflow for running mikropml with many different train/test splits

To investigate the variation in model performance depending on the train and test set used [32, 139], we provide examples of how to `run_ml()` many times with different train/test splits and how to get summary information about model performance on a local computer or on a high-performance computing cluster using a Snakemake workflow.

Figure 4.6: The mikropml pipeline



4.4.3.4 Tuning & visualization

One particularly important aspect of ML is hyperparameter tuning. We provide a reasonable range of default hyperparameters for each model type. However practitioners should explore whether that range is appropriate for their data, or if they should customize the hyperparameter range. Therefore, we provide a function `plot_hp_performance()` to plot the cross-validation performance metric of a single model or models built using different train/test splits. This helps evaluate if the hyperparameter range is being searched exhaustively and allows the user to pick the ideal set. We also provide summary plots of test performance metrics for the many train/test splits with different models using `plot_model_performance()`. Examples are described in the accompanying vignette on hyperparameter tuning.

4.4.3.5 Dependencies

mikropml is written in R [67] and depends on several packages: `dplyr` [148], `rlang` [149] and `caret` [132]. The ML algorithms supported by `mikropml` require: `glmnet` [143], `e1071` [150], and `MLmetrics` [151] for logistic regression, `rpart2` [145] for decision trees, `randomForest` [146] for random forest, `xgboost` [147] for xgboost, and `kernlab` [144] for support vector machines. We also allow for parallelization of cross-validation and other steps using the `foreach`, `doFuture`, `future.apply`, and `future` packages [152]. Finally, we use `ggplot2` for

plotting [141].

4.4.4 Acknowledgments

We thank members of the Schloss Lab who participated in code clubs related to the initial development of the pipeline, made documentation improvements, and provided general feedback. We also thank Nick Lesniak for designing the mikropml logo.

We thank the US Research Software Sustainability Institute (NSF #1743188) for providing training to KLS at the Winter School in Research Software Engineering.

4.4.5 Funding

Salary support for PDS came from NIH grant 1R01CA215574. KLS received support from the NIH Training Program in Bioinformatics (T32 GM070449). ZL received support from the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE 1256260. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

4.4.6 Author contributions

BDT, ZL, and KLS contributed equally. Author order among the co-first authors was determined by time since joining the project.

BDT, ZL, and KLS conceptualized the study and wrote the code. KLS structured the code in R package form. BDT, ZL, JW, and PDS developed methodology. PDS, ES, and JW supervised the project. BDT, ZL, and KLS wrote the original draft. All authors reviewed and edited the manuscript.

4.4.7 Conflicts of interest

None.

CHAPTER 5

Discussion

5.1 Major contributions

This dissertation introduces two new tools that improve ML capabilities for microbiome research and beyond, applies ML with microbiome data to CDI severity prediction, and introduces two new educational resources that teach coding for data science to young audiences and scientists. For all of the analyses described in this dissertation, the complete software workflows and dependencies required to reproduce the results are publicly available with open source licenses so that anyone can reproduce, replicate, or build upon our work. The impact of this work spans microbial ecology, gut microbiome research, applied machine learning, and data science education.

5.1.1 Novel method for reference-based OTU clustering

OptiFit is a novel OTU clustering method that enables high quality OTUs for ML workflows and other applications where consistent OTUs are required. Prior to the development of OptiFit, the only option for researchers who wanted to deploy OTU-based ML models was to cluster both the training set and external validation sets to the same database using a closed-reference clustering method. Existing tools for reference-based clustering against databases produce lower quality OTUs than *de novo* clustering with OptiClust. However, *de novo* clustering results in slightly different OTU assignments when adding new sequences, thus models trained on one dataset could not be deployed on new data due to incompatible features. Now with OptiFit, an initial dataset can be clustered *de novo* with OptiClust and then used to train a model, then new sequences from an external validation set can be fit to the OTUs from the training data prior to deploying the model on the new dataset. A follow-up paper demonstrated the suitability of OptiFit for this very task on a colorectal cancer dataset to distinguish patients with screen-relevant neoplasias from normal controls

[48]. OptiFit opens a new door for microbial ecologists to deploy ML models using higher quality OTUs than were possible before.

5.1.2 Microbiome models for prediction of severe CDI outcomes

Prior studies to date have trained models to predict severe CDI outcomes using routine clinical data, selected serum biomarkers, curated variables from EHR data, or entire EHRs. However, none have focused on using the initial taxonomic composition of the gut microbiome to predict CDI severity, despite ample evidence for a link between dysbiosis and *C. difficile* colonization, infection, and recurrence. We trained models on OTU relative abundances collected on the day of CDI diagnosis to predict four different definitions of severity. Models trained to predict the pragmatic severity definition performed best, as this definition uses as much data as possible while also using physicians' determinations of whether severe outcomes were CDI-attributable when available. While these models did not outperform prior EHR-based models extracted two days after diagnosis, the pragmatic severity models matched the performance of EHR-based models from the day of diagnosis. These results provide an initial exploration of the utility of OTU-based models for predicting CDI severity, and they may become more clinically relevant in the future as new evidence emerges of efficacious treatments for preventing severity.

5.1.3 Educational resources

In Chapter 4.2, we introduced a new curriculum to teach introductory Python for data science via live-coding or a flipped classroom format. We deployed the curriculum for in-person and virtual Girls Who Code clubs during a three year period with high school students as the audience. The curriculum takes students from having no knowledge of programming to being able to analyze a real-world dataset and present their findings to the group. In a post-survey, students overwhelmingly reported that they improved their Python programming skills, problem solving and critical thinking, collaboration with others, and self-confidence. Not only were the students we taught positively impacted; our curriculum is free and available with an open source license so any other educators can use our curriculum or modify it for their own needs. This curriculum is continually improved upon and is still in use for the chapter of Girls Who Code at the University of Michigan Department of Computational Medicine and Bioinformatics.

In Chapter 4.3, we introduced a new curriculum to teach coding for reproducible research practices to scientists and other researchers in an academic setting. The Carpentries materials that inspired us taught three topics in a disparate manner: introductory R programming,

the Unix shell, and version control with git and GitHub. Our curriculum covers these topics in an integrated manner so that learners understand how they are used together in practice. We piloted the curriculum in a virtual workshop and assessed our work with a post-workshop survey. On average, learners reported that they felt more confident writing programs, using programming to work with data, overcoming problems while programming, and searching for answers to technical questions online. This curriculum is still in use today for Carpentries workshops at the University of Michigan and is freely available with an open source license for anyone to use and modify.

5.1.4 Software

In Chapter 4.4, we introduced a tool that integrates current best practices for ML in a user-friendly R package. Our goal was to enable researchers who are novices in ML to train and evaluate models with guard rails to prevent common pitfalls, while allowing experienced users to tailor the package for advanced needs. At the time of this writing, mikropml has been downloaded 13,471 times from the Comprehensive R Archive Network and 24,727 times from the Anaconda package manager, suggesting a healthy user base. The reach of mikropml has expanded outside of our immediate scientific network and into fields spanning gut microbiome research, microbial ecology, public health, and environmental research. Rather than write code intended for one-time-use-only to conduct the ML analyses we routinely perform, we chose to bundle our methods into a package for others within and outside our lab to reuse for their own research. As a result, our efforts have contributed directly to the greater scientific endeavor, with 18 citations to date of the mikropml publication.

In addition to mikropml, other software tools were developed while conducting the research described in this dissertation. These include: schtools, an R package for processing output from the mothur program and miscellaneous functions for microbiome research [109]; the mikropml snakemake workflow, a template for building reusable and scalable machine learning pipelines with mikropml for use in high performance computing environments [105], and the mothur snakemake workflow, a template implementing the mothur MiSeq SOP for processing 16S rRNA gene amplicon sequence data and authoring reproducible scientific manuscripts [153]. While we have not published any stand-alone papers to describe these tools, they have been used within the Schloss Lab for several manuscripts-in-process as well as published studies [48, 154].

5.2 Future work

Below, we discuss key areas of improvement and propose ideas to build on the work described in Chapters 2 through 4.

5.2.1 Integrate microbiota with clinical factors for improved CDI severity prediction

Our OTU-based models described in Chapter 3 were trained on a different dataset as the EHR-based models we compared them to. Since the different datasets have different proportions of severe cases, precision and AUPRC are not directly comparable. While AUROC has the same baseline regardless of the dataset and is thus always directly comparable, it is not as useful for rare outcomes because the model may identify many true negatives but few true positives and yet report a high AUROC. A more salient comparison would train models on the same cohort of patients using either OTUs, EHRs, or both in order to determine which approach leads to the best performance in terms of AUPRC. However, to demonstrate clinical value, it is not enough to simply show that one modeling approach outperforms another. How a model might improve clinical practice if it were deployed must be considered. This especially relates to the treatment options available along with their potential risks, which influences which performance metrics are most meaningful. A large increase in AUROC, AUPRC, or other metrics may or may not translate to a large increase in benefit to patients. In situations where predicting a severe case may lead clinicians to choose a treatment option that has an established record of safety, such as oral fidaxomicin instead of vancomycin, some false positives are tolerable to a certain extent and a lower precision is acceptable (although still better than a no-skill model). On the other hand, if new evidence were to emerge of a treatment preventing severe outcomes but with substantial risk of negative side effects, fewer false positives and a higher precision would be required. Collaborating with clinicians in the infectious dieases specialty is paramount to discuss the performance requried depending on the intervention at hand. The ultimate goal of CDI severity prediction models is to help clinicians identify early on which patients are at risk of experiencing a severe outcome so they can tailor treatments to prevent the outcome from ever occurring, but care must be taken to ensure no harm is inflicted on patients who never would have experienced a severe outcome, and to ensure that clinicians will actually find the model useful to support their decision-making.

5.2.1.1 Decision curve analysis

For the analysis of potential clinical value, we reported the precision at the 95th percentile of risk, which is the decision threshold where 5% of cases are predicted to be positive and would thus undergo a different treatment in order to prevent the adverse outcome from occurring. Choosing this threshold allowed for comparison to the previously published EHR models which reported precision at that threshold. However, rather than evaluating performance at a single threshold, we could extend this across a range of thresholds. Decision curve analysis would explore how the confusion matrix varies across a range of thresholds for models of interest [155]. We could then compare the net benefit, NNS, or other metrics for different modeling approaches, as their relative performance may vary across decision thresholds. A model based on only OTUs may perform optimally at a different threshold than a model based on only EHRs, and different thresholds could be selected for model deployment depending on the importance of recall versus precision for the alternative treatment being considered by clinicians.

5.2.1.2 Cost-benefit analysis

The costs of model training, deployment, and treatment are significant factors that influence the practicality of deploying models in clinical settings. If a model has good discriminative performance, it may never be used if it is expensive to collect the data for deployment. Similarly, an inexpensive model may never be used if the alternative treatment it would be paired with is too expensive. We did not consider these costs when evaluating the potential clinical value, although we reported the NNS and NNB when paired with the NNT of fidaxomicin so the work could be extended to consider costs as well as other treatment options. (For example, bezlotoxumab has also been shown to prevent recurrent CDI in humans as well as systemic organ damage in mice [156, 157]. However, it is used as an adjuvant therapy and as such it does not replace antibiotics for CDI treatment.) A predictive model paired with a treatment may be cost-effective if the decrease in costs for averting severe outcomes outweighs the increase in treatment costs for cases predicted positive plus the costs of deployment, or if any increase in cost is deemed worth the benefit [158]. A limitation of cost-benefit analysis techniques is that the most often used metric of benefit (Quality-Adjusted Life-Years) is controversial, as it is prone to systematic bias and devalues health gains for patients with disabilities [158, 159, 160, 161]. Although existing methodologies for cost-benefit analyses are imperfect, performing a thorough cost-benefit analysis would provide more information about whether deploying CDI severity prediction models could be worth the estimated benefits gained.

5.2.2 Beyond taxonomic composition

Efforts to find consistent changes in taxonomic composition of microbiomes between normal and dysbiotic states have found mixed success, in part because interpersonal variability in taxonomic composition sometimes exceeds the variability between disease states [162, 163]. ML models for diagnosing colorectal cancer or predicting severe CDI perform moderately well, but may not perform well enough to justify clinical deployment. Variability of microbiome composition between individuals with the same disease status may be explained by functional redundancy, where different microbial species carry out the same functions and thus can replace each other with little effect on the overall function of the community [164]. Extending analyses of taxonomic composition to also include the functional composition of the microbiome may shed more light on how the microbiome changes in disease states. Sequencing whole metagenomes to identify the genes present and annotate known gene functions is commonly used to build a profile of functional potential of the microbiome. Functional potential could be paired with meta-transcriptomics or untargeted mass spectrometry to validate the gene products that are active in a community, thus painting a more precise picture of active microbial functions than with metagenomics alone. Incorporating the known functional potential of the microbiome from metagenomic data may help account for functional redundancy and improve the performance of OTU-based models in classifying CRC, predicting CDI severity, or other microbiome modeling problems. These insights could inform the design of future experiments to determine the mechanisms of dysbiosis or improve performance of ML models for clinical decision making.

5.2.3 Continued maintenance of software tools and educational resources

It is notoriously difficult to fund the development and maintenance of scientific software and educational resources. Nevertheless, we initially developed and continue to maintain the open source contributions described in Chapter 4 with our discretionary time because we believe they are valuable to the scientific community and society at large. Developing software and educational resources is never a one-and-done task; they must be maintained as users discover and report bugs, new methods are discovered, and the preferences of the community change over time. While no tool is designed to be used forever (despite the best intentions of fans of *certain* programming languages), neglecting to maintain a tool will unnecessarily hasten its obsolescence. We would much prefer to honor the time and effort spent during initial development, as well as that of end-users who adopt our tools and resources, by continuing to maintain them. However, few funding mechanisms through

traditional grant-making agencies exist to maintain existing resources, as most value new discoveries and ideas [9]. We are hopeful that the landscape is changing for the better with the announcement of programs such as the Better Software for Science initiative by the Alfred P. Sloan Foundation [165]. Funding mechanisms like these will enable scientists and researchers to not only create new tools and resources but also maintain them over time, so that the time, effort, and other resources expended in creating and adopting them are used efficiently.

5.3 Conclusions

In this work, we introduced a novel method for OTU clustering that improves the ability of researchers to apply ML to microbiome research, applied ML to predict the severity of CDI infections from the composition of the gut microbiome, and introduced three new resources that empower data scientists from a broad range of backgrounds to go from coding novices to ML practitioners. This dissertation advances bioinformatics for microbiome research from the start of the data analysis pipeline through applying machine learning to biological and clinical problems, and ultimately toward enabling other scientists to reproduce, replicate, and build upon our work.

BIBLIOGRAPHY

[1] Allan Konopka. 2009. What is microbial community ecology? ISME J 3(11):1223–1230. http://dx.doi.org/10.1038/ismej.2009.88.

[2] Gabriele Berg, Daria Rybakova, Doreen Fischer, Tomislav Cernava, Marie-Christine Champomier Vergès, Trevor Charles, Xiaoyulong Chen, Luca Cocolin, Kellye Eversole, Gema Herrero Corral, Maria Kazou, Linda Kinkel, Lene Lange, Nelson Lima, Alexander Loy, James A. Macklin, Emmanuelle Maguin, Tim Mauchline, Ryan McClure, Birgit Mitter, Matthew Ryan, Inga Sarand, Hauke Smidt, Bettina Schelkle, Hugo Roume, G. Seghal Kiran, Joseph Selvin, Rafael Soares Correa de Souza, Leo van Overbeek, Brajesh K. Singh, Michael Wagner, Aaron Walsh, Angela Sessitsch, and Michael Schloter. 2020. Microbiome definition re-visited: Old concepts and new challenges. Microbiome 8(1):103. http://dx.doi.org/10.1186/s40168-020-00875-0.

[3] Human Microbiome Project Consortium. 2012. Structure, function and diversity of the healthy human microbiome. Nature 486(7402):207–214. http://dx.doi.org/10.1038/nature11234.

[4] Katharine Z. Coyte and Seth Rakoff-Nahoum. 2019. Understanding Competition and Cooperation within the Mammalian Gut Microbiome. Current Biology 29(11):R538–R544. http://dx.doi.org/10.1016/j.cub.2019.04.017.

[5] Valentina Tremaroli and Fredrik Bäckhed. 2012. Functional interactions between the gut microbiota and host metabolism. Nature 489(7415):242–249. http://dx.doi.org/10.1038/nature11552.

[6] Robert P. Dickson, John R. Erb-Downward, Hallie C. Prescott, Fernando J. Martinez, Jeffrey L. Curtis, Vibha N. Lama, and Gary B. Huffnagle. 2020. Analysis of Culture-Dependent versus Culture-Independent Techniques for Identification of Bacteria in Clinically Obtained Bronchoalveolar Lavage Fluid. Journal of Clinical Microbiology 52(10):3605–3613. http://dx.doi.org/10.1128/jcm.01028-14.

[7] Norman R. Pace, David A. Stahl, David J. Lane, and Gary J. Olsen. 1986. The Analysis of Natural Microbial Populations by Ribosomal RNA Sequences. In K. C. Marshall, editor, Advances in Microbial Ecology, Advances in Microbial Ecology, pages 1–55. Springer US, Boston, MA. ISBN 978-1-4757-0611-6. http://dx.doi.org/10.1007/978-1-4757-0611-6_1.

[8] Christopher P Kolbert and David H Persing. 1999. Ribosomal DNA sequencing as a tool for identification of bacterial pathogens. Current Opinion in Microbiology 2(3):299–305. http://dx.doi.org/10.1016/S1369-5274(99)80052-6.

[9] Patrick D. Schloss. 2019. Reintroducing mothur: 10 years later. Appl Environ Microbiol http://dx.doi.org/10.1128/AEM.02343-19.

[10] Sailajah Janarthanan, Ivo Ditah, Douglas G. Adler, and Murray N. Ehrinpreis. 2012. *Clostridium Difficile*-Associated Diarrhea and Proton Pump Inhibitor Therapy: A Meta-Analysis. Official journal of the American College of Gastroenterology | ACG 107(7):1001. http://dx.doi.org/10.1038/ajg.2012.179.

[11] Sarah Tomkovich, Ana Taylor, Jacob King, Joanna Colovas, Lucas Bishop, Kathryn McBride, Sonya Royzenblat, Nicholas A. Lesniak, Ingrid L. Bergin, and Patrick D. Schloss. 2021. An Osmotic Laxative Renders Mice Susceptible to Prolonged *Clostridioides difficile* Colonization and Hinders Clearance. mSphere 6(5):10.1128/msphere.00629–21. http://dx.doi.org/10.1128/msphere.00629-21.

[12] Paul Feuerstadt, Nicolette Theriault, and Glenn Tillotson. 2023. The burden of CDI in the United States: A multifactorial challenge. BMC Infectious Diseases 23(1):132. http://dx.doi.org/10.1186/s12879-023-08096-0.

[13] Casey M. Theriot, Alison A. Bowman, and Vincent B. Young. 2016. Antibiotic-Induced Alterations of the Gut Microbiota Alter Secondary Bile Acid Production and Allow for Clostridium difficile Spore Germination and Outgrowth in the Large Intestine. mSphere 1(1):10.1128/msphere.00045–15. http://dx.doi.org/10.1128/msphere.00045-15.

[14] Charlie G. Buffie, Vanni Bucci, Richard R. Stein, Peter T. McKenney, Lilan Ling, Asia Gobourne, Daniel No, Hui Liu, Melissa Kinnebrew, Agnes Viale, Eric Littmann, Marcel R. M. van den Brink, Robert R. Jenq, Ying Taur, Chris Sander, Justin R. Cross, Nora C. Toussaint, Joao B. Xavier, and Eric G. Pamer. 2015. Precision microbiome reconstitution restores bile acid mediated resistance to Clostridium difficile. Nature 517(7533):205–208. http://dx.doi.org/10.1038/nature13828.

[15] Joseph A. Sorg and Abraham L. Sonenshein. 2010. Inhibiting the Initiation of Clostridium difficile Spore Germination using Analogs of Chenodeoxycholic Acid, a Bile Acid. Journal of Bacteriology 192(19):4983–4990. http://dx.doi.org/10.1128/jb.00610-10.

[16] Sarah Tomkovich, Joshua M. A. Stough, Lucas Bishop, and Patrick D. Schloss. 2020. The Initial Gut Microbiota and Response to Antibiotic Perturbation Influence *Clostridioides difficile* Clearance in Mice. mSphere 5(5). http://dx.doi.org/10.1128/mSphere.00869-20.

[17] Nicholas A. Lesniak, Alyxandria M. Schubert, Hamide Sinani, and Patrick D. Schloss. 2021. Clearance of Clostridioides difficile Colonization Is Associated with Antibiotic-Specific Bacterial Changes. mSphere 6(3). http://dx.doi.org/10.1128/mSphere.01238-20.

[18] Nicholas A. Lesniak, Alyxandria M. Schubert, Kaitlin J. Flynn, Jhansi L. Leslie, Hamide Sinani, Ingrid L. Bergin, Vincent B. Young, and Patrick D. Schloss. 2022. The Gut Bacterial Community Potentiates *Clostridioides difficile* Infection Severity. mBio 13(4):e01183–22. http://dx.doi.org/10.1128/mbio.01183-22.

[19] Jennifer Lucado and Anne Elixhauser. 2012. *Clostridium Difficile* Infections (CDI) in Hospital Stays, 2009. HCUP Statistical Brief #124. AHRQ http://www.hcup-us.ahrq.gov/reports/statbriefs/sb124.pdf.

[20] Z. Kassam, C. Cribb Fabersunne, M. B. Smith, E. J. Alm, G. G. Kaplan, G. C. Nguyen, and A. N. Ananthakrishnan. 2016. *Clostridium Difficile* associated risk of death score (CARDS): A novel severity score to predict mortality among hospitalised patients with *C. difficile<i/> infection*. Aliment Pharmacol Ther 43(6):725–733. http://dx.doi.org/10.1111/apt.13546.

[21] David Peprah, Alexander S. Chiu, Raymond A. Jean, and Kevin Y. Pei. 2019. Comparison of Outcomes Between Total Abdominal and Partial Colectomy for the Management of Severe, ComplicatedClostridium DifficileInfection. Journal of the American College of Surgeons 228(6):925. http://dx.doi.org/10.1016/j.jamcollsurg.2018.11.015.

[22] C. P. Kelly. 2012. Can we identify patients at high risk of recurrent *Clostridium difficile* infection? Clinical Microbiology and Infection 18:21–27. http://dx.doi.org/10.1111/1469-0691.12046.

[23] Sofía de la Villa, Sergio Herrero, Patricia Muñoz, Carmen Rodríguez, Maricela Valerio, Elena Reigadas, Ana Álvarez-Uría, Luis Alcalá, Mercedes Marín, María Olmedo, Martha Kestler, Esther Chamorro, and Emilio Bouza. 2023. Real-world Use of Bezlotoxumab and Fecal Microbiota Transplantation for the Treatment of Clostridioides difficile Infection. Open Forum Infectious Diseases 10(2):ofad028. http://dx.doi.org/10.1093/ofid/ofad028.

[24] Rajiv D. Poduval, Ramdas P. Kamath, Marilou Corpuz, Edward P. Norkus, and C. S. Pitchumoni. 2000. *Clostridium Difficile* and Vancomycin-Resistant Enterococcus: The New Nosocomial Alliance. Official journal of the American College of Gastroenterology | ACG 95(12):3513. http://dx.doi.org/10.1111/j.1572-0241.2000.03291.x.

[25] Alexander B. Smith, Matthew L. Jenior, Orlaith Keenan, Jessica L. Hart, Jonathan Specker, Arwa Abbas, Paula C. Rangel, Chao Di, Jamal Green, Katelyn A. Bustin, Jennifer A. Gaddy, Maribeth R. Nicholson, Clare Laut, Brendan J. Kelly, Megan L. Matthews, Daniel R. Evans, Daria Van Tyne, Emma E. Furth, Jason A. Papin, Frederic D. Bushman, Jessi Erlichman, Robert N. Baldassano, Michael A. Silverman, Gary M. Dunny, Boone M. Prentice, Eric P. Skaar, and Joseph P. Zackular. 2022. Enterococci enhance *Clostridioides difficile* pathogenesis. Nature 611(7937):780–786. http://dx.doi.org/10.1038/s41586-022-05438-x.

[26] Sayash Kapoor and Arvind Narayanan. 2022. Leakage and the Reproducibility Crisis in ML-based Science. http://dx.doi.org/10.48550/arXiv.2207.07048.

[27] Fahad Kamran, Shengpu Tang, Erkin Otles, Dustin S. McEvoy, Sameh N. Saleh, Jen Gong, Benjamin Y. Li, Sayon Dutta, Xinran Liu, Richard J. Medford, Thomas S. Valley, Lauren R. West, Karandeep Singh, Seth Blumberg, John P. Donnelly, Erica S. Shenoy, John Z. Ayanian, Brahmajee K. Nallamothu, Michael W. Sjoding, and Jenna Wiens. 2022. Early identification of patients admitted to hospital for covid-19 at risk of clinical deterioration: Model development and multisite external validation study. BMJ 376:e068576. http://dx.doi.org/10.1136/bmj-2021-068576.

[28] Erkin Ötleş, Emily A. Balczewski, Micah Keidan, Jeeheh Oh, Alieysa Patel, Vincent B. Young, Krishna Rao, and Jenna Wiens. 2023. *Clostridioides Difficile* infection surveillance in intensive care units and oncology wards using machine learning. Infection Control & Hospital Epidemiology pages 1–6. http://dx.doi.org/10.1017/ice.2023.54.

[29] Michael G. Dieterle, Rosemary Putler, D. Alexander Perry, Anitha Menon, Lisa Abernathy-Close, Naomi S. Perlman, Aline Penkevich, Alex Standke, Micah Keidan, Kimberly C. Vendrov, Ingrid L. Bergin, Vincent B. Young, and Krishna Rao. 2020. Systemic Inflammatory Mediators Are Effective Biomarkers for Predicting Adverse Outcomes in *Clostridioides difficile* Infection. mBio 11(3):e00180–20. http://dx.doi.org/10.1128/mBio.00180-20.

[30] Nielson T. Baxter, Mack T. Ruffin, Mary A. M. Rogers, and Patrick D. Schloss. 2016. Microbiota-based model improves the sensitivity of fecal immunochemical test for detecting colonic lesions. Genome Med 8(1):37. http://dx.doi.org/10.1186/s13073-016-0290-3.

[31] Alyxandria M. Schubert, Mary A. M. Rogers, Cathrin Ring, Jill Mogle, Joseph P. Petrosino, Vincent B. Young, David M. Aronoff, and Patrick D. Schloss. 2014. Microbiome Data Distinguish Patients with *Clostridium difficile* Infection and Non-*C. difficile*-Associated Diarrhea from Healthy Controls. mBio 5(3). http://dx.doi.org/10.1128/mBio.01021-14.

[32] Begüm D. Topçuoğlu, Nicholas A. Lesniak, Mack T. Ruffin, Jenna Wiens, and Patrick D. Schloss. 2020. A Framework for Effective Application of Machine Learning to Microbiome-Based Classification Problems. mBio 11(3). http://dx.doi.org/10.1128/mBio.00434-20.

[33] Andrew L. Beam, Arjun K. Manrai, and Marzyeh Ghassemi. 2020. Challenges to the Reproducibility of Machine Learning Models in Health Care. JAMA 323(4):305–306. http://dx.doi.org/10.1001/jama.2019.20866.

[34] Jenna Wiens, Suchi Saria, Mark Sendak, Marzyeh Ghassemi, Vincent X. Liu, Finale Doshi-Velez, Kenneth Jung, Katherine Heller, David Kale, Mohammed Saeed, Pilar N. Ossorio, Sonoo Thadaney-Israni, and Anna Goldenberg. 2019. Do no harm: A roadmap for responsible machine learning for health care. Nat Med 25(9):1337–1340. http://dx.doi.org/10.1038/s41591-019-0548-6.

[35] National Center for Science and Engineering Statistics (NCSES). 2023. Diversity and STEM: Women, Minorities, and Persons with Disabilities 2023. Technical report, National Science Foundation. https://ncses.nsf.gov/pubs/nsf23315/report/.

[36] Deborah Gilshan. 2021. The Ethics of Diversity. https://corpgov.law.harvard.edu/2021/02/03/the-ethics-of-diversity/.

[37] Bo Cowgill, Fabrizio Dell'Acqua, Samuel Deng, Daniel Hsu, Nakul Verma, and Augustin Chaintreau. 2020. Biased Programmers? Or Biased Data? A Field Experiment in Operationalizing AI Ethics. http://arxiv.org/abs/2012.02394.

[38] Lu Hong and Scott E. Page. 2004. Groups of diverse problem solvers can outperform groups of high-ability problem solvers. Proc Natl Acad Sci USA 101(46):16385–16389. http://dx.doi.org/10.1073/pnas.0403723101.

[39] Patrick D. Schloss. 2018. Identifying and Overcoming Threats to Reproducibility, Replicability, Robustness, and Generalizability in Microbiome Research. mBio 9(3):e00525–18, /mbio/9/3/mBio.00525–18.atom. http://dx.doi.org/10.1128/mBio.00525-18.

[40] Geir Kjetil Sandve, Anton Nekrutenko, James Taylor, and Eivind Hovig. 2013. Ten Simple Rules for Reproducible Computational Research. PLOS Computational Biology 9(10):e1003285. http://dx.doi.org/10.1371/journal.pcbi.1003285.

[41] Greg Wilson, Jennifer Bryan, Karen Cranston, Justin Kitzes, Lex Nederbragt, and Tracy K. Teal. 2017. Good enough practices in scientific computing. PLOS Computational Biology 13(6):e1005510. http://dx.doi.org/10.1371/journal.pcbi.1005510.

[42] Andrew Gelman and Eric Loken. 2016. The statistical crisis in science. The best writing on mathematics (Pitici M, ed) pages 305–318.

[43] Matthew B. A. McDermott, Shirly Wang, Nikki Marinsek, Rajesh Ranganath, Luca Foschini, and Marzyeh Ghassemi. 2021. Reproducibility in machine learning for health research: Still a ways to go. Science Translational Medicine 13(586):eabb1655. http://dx.doi.org/10.1126/scitranslmed.abb1655.

[44] Kim Cuddington, Karen C Abbott, Frederick R Adler, Mehmet Aydeniz, Rene Dale, Louis J Gross, Alan Hastings, Elizabeth A Hobson, Vadim A Karatayev, Alexander Killion, Aasakiran Madamanchi, Michelle L Marraffini, Audrey L McCombs, Widodo Samyono, Shin-Han Shiu, Karen H Watanabe, and Easton R White. 2023. Challenges and opportunities to build quantitative self-confidence in biologists. BioScience page biad015. http://dx.doi.org/10.1093/biosci/biad015.

[45] Greg Wilson. 2016. Software Carpentry: Lessons learned. F1000Res 3:62. http://dx.doi.org/10.12688/f1000research.3-62.v2.

[46] Deans for Impact. 2015. The Science of Learning. Technical report. https://carpentries.github.io/instructor-training/files/papers/science-of-learning-2015.pdf.

[47] Karen Word, Maneesha Sane, Kelly Barnes, Sarah M Brown, François Michonneau, Christina Koch, Rayna M Harris, Erin Becker, Brian Ballsun-Stanton, Gerard Capes, Toby Hodges, Serah Njambi, Sarah Stevens, Zhian Namir Kamvar, Angela Li, Ariel Deardorff, Pradeep Eranti, SherAaron Hurt, Aleksandra Nenadic, Eric Jankowski, Dr. Kari L. Jordan, Laurent Heirendt, Murray Cadzow, Aaron Tran, Alec L. Robitaille, Alexander James Ball, Bianca Peterson, Christa de Kock, Daniel Chen, Darya P Vanichkina, Pérez-Suárez, Elizabeth McAulay, George Milunovich, GIUSEPPE PROFITI, Hugo Gruson, Jeffrey Oliver, Jonah Duckles, Matti Juvonen, Konrad U. Förstner, Lex Nederbragt, Liz Stokes, Martin Stoffers, Michael Black, Michael Henry, Neal Davis, Sarah Peter, Sichong, Tong Liang, and Ashwin Vishnu Mohanan. 2021. Carpentries/instructor-training: The carpentries instructor training november 2021. Zenodo. http://dx.doi.org/10.5281/zenodo.5709383.

[48] Courtney R. Armour, Kelly L. Sovacool, William L. Close, Begüm D. Topçuoğlu, Jenna Wiens, and Patrick D. Schloss. 2023. Machine learning classification by fitting amplicon sequences to existing OTUs. http://dx.doi.org/10.1101/2022.09.01.506299.

[49] Kelly L. Sovacool, Sarah L. Westcott, M. Brodie Mumphrey, Gabrielle A. Dotson, and Patrick D. Schloss. 2022. OptiFit: An Improved Method for Fitting Amplicon Sequences to Existing OTUs. mSphere 7(1):e00916–21. http://dx.doi.org/10.1128/msphere.00916-21.

[50] Sarah L. Westcott and Patrick D. Schloss. 2017. OptiClust, an Improved Method for Assigning Amplicon-Based Sequence Data to Operational Taxonomic Units. mSphere 2(2):e00073–17. http://dx.doi.org/10.1128/mSphereDirect.00073-17.

[51] Sarah L. Westcott and Patrick D. Schloss. 2015. De novo clustering methods outperform reference-based methods for assigning 16S rRNA gene sequences to operational taxonomic units. PeerJ 3:e1487. http://dx.doi.org/10.7717/peerj.1487.

[52] Robert C. Edgar. 2010. Search and clustering orders of magnitude faster than BLAST. Bioinformatics 26(19):2460–2461. http://dx.doi.org/10.1093/bioinformatics/btq461.

[53] Torbjørn Rognes, Tomáš Flouri, Ben Nichols, Christopher Quince, and Frédéric Mahé. 2016. VSEARCH: A versatile open source tool for metagenomics. PeerJ 4:e2584. http://dx.doi.org/10.7717/peerj.2584.

[54] Clustering sequences into OTUs using q2-vsearch — QIIME 2 2021.2.0 documentation. https://docs.qiime2.org/2021.2/tutorials/otu-clustering/.

[55] Eric R. Johnston, Luis M. Rodriguez-R, Chengwei Luo, Mengting M. Yuan, Liyou Wu, Zhili He, Edward A. G. Schuur, Yiqi Luo, James M. Tiedje, Jizhong Zhou, and Konstantinos T. Konstantinidis. 2016. Metagenomics Reveals Pervasive Bacterial Popula-

tions and Reduced Community Diversity across the Alaska Tundra Ecosystem. Front Microbiol 7. http://dx.doi.org/10.3389/fmicb.2016.00579.

[56] Michael W. Henson, David M. Pitre, Jessica Lee Weckhorst, V. Celeste Lanclos, Austen T. Webber, and J. Cameron Thrash. 2016. Artificial Seawater Media Facilitate Cultivating Members of the Microbial Majority from the Gulf of Mexico. mSphere 1(2). http://dx.doi.org/10.1128/mSphere.00028-16.

[57] Patrick D. Schloss, Alyxandria M. Schubert, Joseph P. Zackular, Kathryn D. Iverson, Vincent B. Young, and Joseph F. Petrosino. 2012. Stabilization of the murine gut microbiome following weaning. Gut Microbes 3(4):383–393. http://dx.doi.org/10.4161/gmic.21008.

[58] T. Z. DeSantis, P. Hugenholtz, N. Larsen, M. Rojas, E. L. Brodie, K. Keller, T. Huber, D. Dalevi, P. Hu, and G. L. Andersen. 2006. Greengenes, a Chimera-Checked 16S rRNA Gene Database and Workbench Compatible with ARB. AEM 72(7):5069–5072. http://dx.doi.org/10.1128/AEM.03006-05.

[59] Christian Quast, Elmar Pruesse, Pelin Yilmaz, Jan Gerken, Timmy Schweer, Pablo Yarza, Jörg Peplies, and Frank Oliver Glöckner. 2013. The SILVA ribosomal RNA gene database project: Improved data processing and web-based tools. Nucleic Acids Research 41(D1):D590–D596. http://dx.doi.org/10.1093/nar/gks1219.

[60] James R. Cole, Qiong Wang, Jordan A. Fish, Benli Chai, Donna M. McGarrell, Yanni Sun, C. Titus Brown, Andrea Porras-Alfaro, Cheryl R. Kuske, and James M. Tiedje. 2014. Ribosomal Database Project: Data and tools for high throughput rRNA analysis. Nucl Acids Res 42(D1):D633–D642. http://dx.doi.org/10.1093/nar/gkt1244.

[61] José A. Navas-Molina, Juan M. Peralta-Sánchez, Antonio González, Paul J. McMurdie, Yoshiki Vázquez-Baeza, Zhenjiang Xu, Luke K. Ursell, Christian Lauber, Hongwei Zhou, Se Jin Song, James Huntley, Gail L. Ackermann, Donna Berg-Lyons, Susan Holmes, J. Gregory Caporaso, and Rob Knight. 2013. Chapter Nineteen - Advancing Our Understanding of the Human Microbiome Using QIIME. In Edward F. DeLong, editor, Methods in Enzymology, volume 531 of *Microbial Metagenomics, Metatranscriptomics, and Metaproteomics*, pages 371–444. Academic Press. http://dx.doi.org/10.1016/B978-0-12-407863-5.00019-8.

[62] Patrick D. Schloss and Sarah L. Westcott. MiSeq SOP. https://mothur.org/wiki/MiSeq_SOP.

[63] James J. Kozich, Sarah L. Westcott, Nielson T. Baxter, Sarah K. Highlander, and Patrick D. Schloss. 2013. Development of a Dual-Index Sequencing Strategy and Curation Pipeline for Analyzing Amplicon Sequence Data on the MiSeq Illumina Sequencing Platform. Appl Environ Microbiol 79(17):5112–5120. http://dx.doi.org/10.1128/AEM.01043-13.

[64] Robert C. Edgar, Brian J. Haas, Jose C. Clemente, Christopher Quince, and Rob Knight. 2011. UCHIME improves sensitivity and speed of chimera detection. Bioinformatics 27(16):2194–2200. http://dx.doi.org/10.1093/bioinformatics/btr381.

[65] Evan Bolyen, Jai Ram Rideout, Matthew R. Dillon, Nicholas A. Bokulich, Christian C. Abnet, Gabriel A. Al-Ghalith, Harriet Alexander, Eric J. Alm, Manimozhiyan Arumugam, Francesco Asnicar, Yang Bai, Jordan E. Bisanz, Kyle Bittinger, Asker Brejnrod, Colin J. Brislawn, C. Titus Brown, Benjamin J. Callahan, Andrés Mauricio Caraballo-Rodríguez, John Chase, Emily K. Cope, Ricardo Da Silva, Christian Diener, Pieter C. Dorrestein, Gavin M. Douglas, Daniel M. Durall, Claire Duvallet, Christian F. Edwardson, Madeleine Ernst, Mehrbod Estaki, Jennifer Fouquier, Julia M. Gauglitz, Sean M. Gibbons, Deanna L. Gibson, Antonio Gonzalez, Kestrel Gorlick, Jiarong Guo, Benjamin Hillmann, Susan Holmes, Hannes Holste, Curtis Huttenhower, Gavin A. Huttley, Stefan Janssen, Alan K. Jarmusch, Lingjing Jiang, Benjamin D. Kaehler, Kyo Bin Kang, Christopher R. Keefe, Paul Keim, Scott T. Kelley, Dan Knights, Irina Koester, Tomasz Kosciolek, Jorden Kreps, Morgan G. I. Langille, Joslynn Lee, Ruth Ley, Yong-Xin Liu, Erikka Loftfield, Catherine Lozupone, Massoud Maher, Clarisse Marotz, Bryan D. Martin, Daniel McDonald, Lauren J. McIver, Alexey V. Melnik, Jessica L. Metcalf, Sydney C. Morgan, Jamie T. Morton, Ahmad Turan Naimey, Jose A. Navas-Molina, Louis Felix Nothias, Stephanie B. Orchanian, Talima Pearson, Samuel L. Peoples, Daniel Petras, Mary Lai Preuss, Elmar Pruesse, Lasse Buur Rasmussen, Adam Rivers, Michael S. Robeson, Patrick Rosenthal, Nicola Segata, Michael Shaffer, Arron Shiffer, Rashmi Sinha, Se Jin Song, John R. Spear, Austin D. Swafford, Luke R. Thompson, Pedro J. Torres, Pauline Trinh, Anupriya Tripathi, Peter J. Turnbaugh, Sabah Ul-Hasan, Justin J. J. van der Hooft, Fernando Vargas, Yoshiki Vázquez-Baeza, Emily Vogtmann, Max von Hippel, William Walters, Yunhu Wan, Mingxun Wang, Jonathan Warren, Kyle C. Weber, Charles H. D. Williamson, Amy D. Willis, Zhenjiang Zech Xu, Jesse R. Zaneveld, Yilong Zhang, Qiyun Zhu, Rob Knight, and J. Gregory Caporaso. 2019. Reproducible, interactive, scalable and extensible microbiome data science using QIIME 2. Nat Biotechnol 37(8):852–857. http://dx.doi.org/10.1038/s41587-019-0209-9.

[66] Johannes Köster and Sven Rahmann. 2012. Snakemake — a scalable bioinformatics workflow engine. Bioinformatics 28(19):2520–2522. http://dx.doi.org/10.1093/bioinformatics/bts480.

[67] R Core Team. 2023. R: A Language and Environment for Statistical Computing. Vienna, Austria. https://www.R-project.org/.

[68] Guido Van Rossum and Fred L. Drake. 2009. Python 3 Reference Manual | Guide books. https://dl.acm.org/doi/book/10.5555/1593511.

[69] Bash Reference Manual. https://www.gnu.org/software/bash/manual/bash.html.

[70] Patrick D. Schloss, Sarah L. Westcott, Thomas Ryabin, Justine R. Hall, Martin Hartmann, Emily B. Hollister, Ryan A. Lesniewski, Brian B. Oakley, Donovan H. Parks, Courtney J. Robinson, Jason W. Sahl, Blaz Stres, Gerhard G. Thallinger,

David J. Van Horn, and Carolyn F. Weber. 2009. Introducing mothur: Open-Source, platform-independent, community-supported software for describing and comparing microbial communities. Applied and Environmental Microbiology 75(23):7537–7541. http://dx.doi.org/10.1128/AEM.01541-09.

[71] Hadley Wickham, Mara Averick, Jennifer Bryan, Winston Chang, Lucy D'Agostino McGowan, Romain François, Garrett Grolemund, Alex Hayes, Lionel Henry, Jim Hester, Max Kuhn, Thomas Lin Pedersen, Evan Miller, Stephan Milton Bache, Kirill Müller, Jeroen Ooms, David Robinson, Dana Paige Seidel, Vitalie Spinu, Kohske Takahashi, Davis Vaughan, Claus Wilke, Kara Woo, and Hiroaki Yutani. 2019. Welcome to the Tidyverse. Journal of Open Source Software 4(43):1686. http://dx.doi.org/10.21105/joss.01686.

[72] Yihui Xie, J. J. Allaire, and Garrett Grolemund. 2018. R Markdown: The Definitive Guide. Taylor & Francis, CRC Press. ISBN 978-1-138-35933-8.

[73] Thomas Lin Pedersen. 2021. Ggraph: An Implementation of Grammar of Graphics for Graphs and Networks. https://CRAN.R-project.org/package=ggraph.

[74] Claus O. Wilke. 2020. Ggtext: Improved text rendering support for 'ggplot2'. https://CRAN.R-project.org/package=ggtext.

[75] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. 2020. Array programming with NumPy. Nature 585(7825):357–362. http://dx.doi.org/10.1038/s41586-020-2649-2.

[76] SRA-Tools - NCBI. https://github.com/ncbi/sra-tools.

[77] Shelley S. Magill, Jonathan R. Edwards, Wendy Bamberg, Zintars G. Beldavs, Ghinwa Dumyati, Marion A. Kainer, Ruth Lynfield, Meghan Maloney, Laura McAllister-Hollod, Joelle Nadle, Susan M. Ray, Deborah L. Thompson, Lucy E. Wilson, and Scott K. Fridkin. 2014. Multistate Point-Prevalence Survey of Health Care–Associated Infections. N Engl J Med 370(13):1198–1208. http://dx.doi.org/10.1056/NEJMoa1306801.

[78] Lena M. Napolitano and Charles E. Edmiston. 2017. *Clostridium Difficile* disease: Diagnosis, pathogenesis, and treatment update. Surgery 162(2):325–348. http://dx.doi.org/10.1016/j.surg.2017.01.018.

[79] Adam Ressler, Joyce Wang, and Krishna Rao. 2021. Defining the black box: A narrative review of factors associated with adverse outcomes from severe *Clostridioides difficile* infection. Therap Adv Gastroenterol 14:17562848211048127. http://dx.doi.org/10.1177/17562848211048127.

[80] Alice Y. Guh, Yi Mu, Lisa G. Winston, Helen Johnston, Danyel Olson, Monica M. Farley, Lucy E. Wilson, Stacy M. Holzbauer, Erin C. Phipps, Ghinwa K. Dumyati, Zintars G. Beldavs, Marion A. Kainer, Maria Karlsson, Dale N. Gerding, and L. Clifford McDonald. 2020. Trends in U.S. Burden of *Clostridioides difficile* Infection and Outcomes. N Engl J Med 382(14):1320–1330. http://dx.doi.org/10.1056/NEJMoa1910215.

[81] Jennie H. Kwon, Margaret A. Olsen, and Erik R. Dubberke. 2015. The morbidity, mortality, and costs associated with *Clostridium difficile* infection. Infect Dis Clin North Am 29(1):123–134. http://dx.doi.org/10.1016/j.idc.2014.11.003.

[82] Claire Nour Abou Chakra, Jacques Pepin, and Louis Valiquette. 2012. Prediction Tools for Unfavourable Outcomes in *Clostridium difficile* Infection: A Systematic Review. PLOS ONE 7(1):e30258. http://dx.doi.org/10.1371/journal.pone.0030258.

[83] D Alexander Perry, Daniel Shirley, Dejan Micic, Pratish C Patel, Rosemary Putler, Anitha Menon, Vincent B Young, and Krishna Rao. 2022. External Validation and Comparison of *Clostridioides difficile* Severity Scoring Systems. Clinical Infectious Diseases 74(11):2028–2035. http://dx.doi.org/10.1093/cid/ciab737.

[84] Krishna Rao, Dejan Micic, Mukil Natarajan, Spencer Winters, Mark J. Kiel, Seth T. Walk, Kavitha Santhosh, Jill A. Mogle, Andrzej T. Galecki, William LeBar, Peter D. R. Higgins, Vincent B. Young, and David M. Aronoff. 2015. *Clostridium Difficile* Ribotype 027: Relationship to Age, Detectability of Toxins A or B in Stool With Rapid Testing, Severe Infection, and Mortality. Clinical Infectious Diseases 61(2):233–241. http://dx.doi.org/10.1093/cid/civ254.

[85] Benjamin Y. Li, Jeeheh Oh, Vincent B. Young, Krishna Rao, and Jenna Wiens. 2019. Using Machine Learning and the Electronic Health Record to Predict Complicated *Clostridium difficile* Infection. Open Forum Infect Dis 6(5):ofz186. http://dx.doi.org/10.1093/ofid/ofz186.

[86] Ira S. Hofer, Michael Burns, Samir Kendale, and Jonathan P. Wanderer. 2020. Realistically Integrating Machine Learning Into Clinical Practice: A Road Map of Opportunities, Challenges, and a Potential Future. Anesthesia & Analgesia 130(5):1115. http://dx.doi.org/10.1213/ANE.0000000000004575.

[87] Min Li, Jinxin Liu, Jiaying Zhu, Huarui Wang, Chuqing Sun, Na L. Gao, Xing-Ming Zhao, and Wei-Hua Chen. 2023. Performance of Gut Microbiome as an Independent Diagnostic Tool for 20 Diseases: Cross-Cohort Validation of Machine-Learning Classifiers. Gut Microbes 15(1):2205386. http://dx.doi.org/10.1080/19490976.2023.2205386.

[88] L Clifford McDonald, Dale N Gerding, Stuart Johnson, Johan S Bakken, Karen C Carroll, Susan E Coffin, Erik R Dubberke, Kevin W Garey, Carolyn V Gould, Ciaran Kelly, Vivian Loo, Julia Shaklee Sammons, Thomas J Sandora, and Mark H Wilcox. 2018. Clinical Practice Guidelines for *Clostridium difficile* Infection in Adults and

Children: 2017 Update by the Infectious Diseases Society of America (IDSA) and Society for Healthcare Epidemiology of America (SHEA). Clinical Infectious Diseases 66(7):e1–e48. http://dx.doi.org/10.1093/cid/cix1085.

[89] Vanessa W. Stevens, Holly E. Shoemaker, Makoto M. Jones, Barbara E. Jones, Richard E. Nelson, Karim Khader, Matthew H. Samore, and Michael A. Rubin. 2020. Validation of the SHEA/IDSA severity criteria to predict poor outcomes among inpatients and outpatients with *Clostridioides difficile* infection. Infection Control & Hospital Epidemiology 41(5):510–516. http://dx.doi.org/10.1017/ice.2020.8.

[90] L. Clifford McDonald, Bruno Coignard, Erik Dubberke, Xiaoyan Song, Teresa Horan, and Preeta K. Kutty. 2007. Recommendations for surveillance of *Clostridium difficile*–Associated disease. Infection Control & Hospital Epidemiology 28(2):140–145. http://dx.doi.org/10.1086/511798.

[91] Yingzhou Wu, Hanqing Liu, Roujia Li, Song Sun, Jochen Weile, and Frederick P. Roth. 2021. Improved pathogenicity prediction for rare human missense variants. The American Journal of Human Genetics 108(10):1891–1906. http://dx.doi.org/10.1016/j.ajhg.2021.08.012.

[92] Christopher M. Rembold. 1998. Number needed to screen: Development of a statistic for disease screening. BMJ 317(7154):307–312. http://dx.doi.org/10.1136/bmj.317.7154.307.

[93] Richard J. Cook and David L. Sackett. 1995. The number needed to treat: A clinically useful measure of treatment effect. BMJ 310(6977):452–454. http://dx.doi.org/10.1136/bmj.310.6977.452.

[94] Andreas Laupacis, David L. Sackett, and Robin S. Roberts. 1988. An assessment of clinically useful measures of the consequences of treatment. New England Journal of Medicine 318(26):1728–1733. http://dx.doi.org/10.1056/NEJM198806303182605.

[95] Les Irwig, Judy Irwig, Lyndal Trevena, and Melissa Sweet. 2008. Relative risk, relative and absolute risk reduction, number needed to treat and confidence intervals. In Smart Health Choices: Making Sense of Health Advice. Hammersmith Press. https://www.ncbi.nlm.nih.gov/books/NBK63647/.

[96] Vincent X Liu, David W Bates, Jenna Wiens, and Nigam H Shah. 2019. The number needed to benefit: Estimating the value of predictive analytics in healthcare. Journal of the American Medical Informatics Association 26(12):1655–1659. http://dx.doi.org/10.1093/jamia/ocz088.

[97] Stuart Johnson, Valéry Lavergne, Andrew M Skinner, Anne J Gonzales-Luna, Kevin W Garey, Ciaran P Kelly, and Mark H Wilcox. 2021. Clinical Practice Guideline by the Infectious Diseases Society of America (IDSA) and Society for Healthcare Epidemiology of America (SHEA): 2021 Focused Update Guidelines on Management of *Clostridioides difficile* Infection in Adults. Clinical Infectious Diseases 73(5):e1029–e1044. http://dx.doi.org/10.1093/cid/ciab549.

[98] Brit Long and Michael Gottlieb. 2022. Oral fidaxomicin versus vancomycin for *Clostridioides difficile* infection. Academic Emergency Medicine 29(12):1506–1507. http://dx.doi.org/10.1111/acem.14600.

[99] Sho Tashiro, Takayuki Mihara, Moe Sasaki, Chiaki Shimamura, Rina Shimamura, Shiho Suzuki, Maiko Yoshikawa, Tatsuki Hasegawa, Yuki Enoki, Kazuaki Taguchi, Kazuaki Matsumoto, Hiroki Ohge, Hiromichi Suzuki, Atsushi Nakamura, Nobuaki Mori, Yoshitomo Morinaga, Yuka Yamagishi, Sadako Yoshizawa, Katsunori Yanagihara, Hiroshige Mikamo, and Hiroyuki Kunishima. 2022. Oral fidaxomicin versus vancomycin for the treatment of *Clostridioides difficile* infection: A systematic review and meta-analysis of randomized controlled trials. Journal of Infection and Chemotherapy 28(11):1536–1545. http://dx.doi.org/10.1016/j.jiac.2022.08.008.

[100] Vijay C. Antharam, Eric C. Li, Arif Ishmael, Anuj Sharma, Volker Mai, Kenneth H. Rand, and Gary P. Wang. 2013. Intestinal Dysbiosis and Depletion of Butyrogenic Bacteria in *Clostridium difficile* Infection and Nosocomial Diarrhea. J Clin Microbiol 51(9):2884–2892. http://dx.doi.org/10.1128/JCM.00845-13.

[101] Matilda Berkell, Mohamed Mysara, Basil Britto Xavier, Cornelis H. van Werkhoven, Pieter Monsieurs, Christine Lammens, Annie Ducher, Maria J. G. T. Vehreschild, Herman Goossens, Jean de Gunzburg, Marc J. M. Bonten, and Surbhi Malhotra-Kumar. 2021. Microbiota-based markers predictive of development of *Clostridioides difficile* infection. Nat Commun 12(1):2241. http://dx.doi.org/10.1038/s41467-021-22302-0.

[102] Yiling Jiang, Eric M. Sarpong, Pamela Sears, and Engels N. Obi. 2022. Budget Impact Analysis of Fidaxomicin Versus Vancomycin for the Treatment of *Clostridioides difficile* Infection in the United States. Infect Dis Ther 11(1):111–126. http://dx.doi.org/10.1007/s40121-021-00480-0.

[103] Kelly R. Reveles, Jennifer L. Backo, Frank A. Corvino, Marko Zivkovic, and Kelly C. Broderick. 2017. Fidaxomicin versus Vancomycin as a First-Line Treatment for *Clostridium difficile*–Associated Diarrhea in Specific Patient Populations: A Pharmacoeconomic Evaluation. Pharmacotherapy: The Journal of Human Pharmacology and Drug Therapy 37(12):1489–1497. http://dx.doi.org/10.1002/phar.2049.

[104] Begüm D. Topçuoğlu, Zena Lapp, Kelly L. Sovacool, Evan Snitkin, Jenna Wiens, and Patrick D. Schloss. 2021. Mikropml: User-Friendly R Package for Supervised Machine Learning Pipelines. JOSS 6(61):3073. http://dx.doi.org/10.21105/joss.03073.

[105] Kelly Sovacool, Zena Lapp, Courtney Armour, Sarah K. Lucas, and Patrick Schloss. 2023. Mikropml Snakemake workflow. Zenodo. http://dx.doi.org/10.5281/zenodo.4759351.

[106] Kelly L Sovacool, Sarah L Tomkovich, Megan L Coden, Jenna Wiens, Vincent B Young, Krishna Rao, and Patrick D Schloss. 2023. Software accompanying the manuscript: Predicting severity of C. difficile infections from the taxonomic composition of the gut microbiome. Zenodo. http://dx.doi.org/10.5281/zenodo.8018793.

[107] Claus O. Wilke. 2020. Cowplot: Streamlined Plot Theme and Plot Annotations for 'ggplot2'. https://CRAN.R-project.org/package=cowplot.

[108] David Sjoberg. 2022. Ggsankey: Sankey, Alluvial and Sankey Bump Plots. https://github.com/davidsjoberg/ggsankey.

[109] Kelly Sovacool, Nick Lesniak, Sarah Lucas, Courtney Armour, and Patrick Schloss. 2022. Schtools: Schloss lab tools for reproducible microbiome research. http://dx.doi.org/10.5281/zenodo.6540686.

[110] Jari Oksanen, Gavin L. Simpson, F. Guillaume Blanchet, Roeland Kindt, Pierre Legendre, Peter R. Minchin, R. B. O'Hara, Peter Solymos, M. Henry H. Stevens, Eduard Szoecs, Helene Wagner, Matt Barbour, Michael Bedward, Ben Bolker, Daniel Borcard, Gustavo Carvalho, Michael Chirico, Miquel De Caceres, Sebastien Durand, Heloisa Beatriz Antoniazi Evangelista, Rich FitzJohn, Michael Friendly, Brendan Furneaux, Geoffrey Hannigan, Mark O. Hill, Leo Lahti, Dan McGlinn, Marie-Helene Ouellette, Eduardo Ribeiro Cunha, Tyler Smith, Adrian Stier, Cajo J. F. Ter Braak, and James Weedon. 2023. Vegan: Community Ecology Package. https://github.com/vegandevs/vegan.

[111] Marlena Duda, Kelly L. Sovacool, Negar Farzaneh, Vy Kim Nguyen, Sarah E. Haynes, Hayley Falk, Katherine L. Furman, Logan A. Walker, Rucheng Diao, Morgan Oneka, Audrey C. Drotos, Alana Woloshin, Gabrielle A. Dotson, April Kriebel, Lucy Meng, Stephanie N. Thiede, Zena Lapp, and Brooke N. Wolford. 2021. Teaching Python for Data Science: Collaborative development of a modular & interactive curriculum. JOSE 4(46):138. http://dx.doi.org/10.21105/jose.00138.

[112] National Center for Education Statistics. 2012. Digest of Education Statistics. https://nces.ed.gov/programs/digest/d12/tables/dt12_349.asp.

[113] Ross J. Benbow and Erika Vivyan. 2016. Gender and Belonging in Undergraduate Computer Science: A Comparative Case Study of Student Experiences in Gateway Courses. Technical report.

[114] Catherine Hill, Christianne Corbett, and Andresse St. Rose. 2010. Why so Few? Women in Science, Technology, Engineering, and Mathematics. AAUW, Washington, D.C. ISBN 978-1-879922-40-2.

[115] Reshma Saujani. 2015. Girls Who Code: Annual Report 2015. http://girlswhocode.com/2015report/.

[116] Kevin S. Bonham and Melanie I. Stefan. 2017. Women are underrepresented in computational biology: An analysis of the scholarly literature in biology, computer science and computational biology. PLoS Comput Biol 13(10):e1005134. http://dx.doi.org/10.1371/journal.pcbi.1005134.

[117] Amanda Stansell. 2019. Breaking Down the 50 Best Jobs in America for 2019 - Glassdoor. https://www.glassdoor.com/research/best-jobs-2019/.

[118] PYPL PopularitY of Programming Language index. https://pypl.github.io/PYPL.html.

[119] Girls Who Code HQ. 2021. Girls Who Code Project Gallery. https://hq.girlswhocode.com/project-gallery.

[120] Douglas Fisher and Nancy Frey. 2013. Better Learning Through Structured Teaching: A Framework for the Gradual Release of Responsibility, 2nd Edition.

[121] Zena Lapp, Kelly L. Sovacool, Nick Lesniak, Dana King, Catherine Barnier, Matthew Flickinger, Jule Krüger, Courtney R. Armour, Maya M. Lapp, Jason Tallant, Rucheng Diao, Morgan Oneka, Sarah Tomkovich, Jacqueline Moltzau Anderson, Sarah K. Lucas, and Patrick D. Schloss. 2022. Developing and deploying an integrated workshop curriculum teaching computational skills for reproducible research. JOSE http://dx.doi.org/10.21105/jose.00144.

[122] Michael Waskom. 2021. Seaborn: Statistical data visualization. JOSS 6(60):3021. http://dx.doi.org/10.21105/joss.03021.

[123] John D. Hunter. 2007. Matplotlib: A 2D graphics environment. Computing in Science Engineering 9(3):90–95. http://dx.doi.org/10.1109/MCSE.2007.55.

[124] Alexander Nederbragt, Rayna Michelle Harris, Alison Presmanes Hill, and Greg Wilson. 2020. Ten quick tips for teaching with participatory live coding. PLOS Computational Biology 16(9):e1008090. http://dx.doi.org/10.1371/journal.pcbi.1008090.

[125] Jo E. Hannay, Tore Dybå, Erik Arisholm, and Dag I. K. Sjøberg. 2009. The effectiveness of pair programming: A meta-analysis. Information and Software Technology 51(7):1110–1122. http://dx.doi.org/10.1016/j.infsof.2009.02.001.

[126] Erin Becker. 2016. Responding to your Learners. https://datacarpentry.org/blog/2016/09/formative-assessment.

[127] David Robinson. 2017. Teach the tidyverse to beginners. http://varianceexplained.org/r/teach-tidyverse/.

[128] The Carpentries. 2018. Live Coding is a Skill. https://carpentries.github.io/instructor-training/14-live/#sticky-notes.

[129] The Carpentries. 2018. The Carpentries Handbook. https://docs.carpentries.org/index.html.

[130] Daniel Chen. 2020. Online Workshop Logistics and Screen Layouts. https://carpentries.org/blog/2020/06/online-workshop-logistics-and_screen-layouts/.

[131] Andrew E. Teschendorff. 2019. Avoiding common pitfalls in machine learning omic data science. Nat Mater 18(5):422–427. http://dx.doi.org/10.1038/s41563-018-0241-z.

[132] Max Kuhn. 2008. Building Predictive Models in R Using the caret Package. Journal of Statistical Software 28(1):1–26. http://dx.doi.org/10.18637/jss.v028.i05.

[133] Max Kuhn, Hadley Wickham, and RStudio. 2020. Tidymodels: Easily Install and Load the 'Tidymodels' Packages .

[134] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research 12(85):2825–2830.

[135] H2O.ai. 2020. H2O: Scalable Machine Learning Platform. https://github.com/h2oai/h2o-3.

[136] Shengpu Tang, Parmida Davarmanesh, Yanmeng Song, Danai Koutra, Michael W. Sjoding, and Jenna Wiens. 2020. Democratizing EHR analyses with FIDDLE: A flexible data-driven preprocessing pipeline for structured clinical data. J Am Med Inform Assoc http://dx.doi.org/10.1093/jamia/ocaa139.

[137] Leo Breiman. 2001. Random forests. Machine Learning 45(1):5–32. http://dx.doi.org/10.1023/A:1010933404324.

[138] Aaron Fisher, Cynthia Rudin, and Francesca Dominici. 2018. All models are wrong, but many are useful: Learning a variable's importance by studying an entire class of prediction models simultaneously .

[139] Zena Lapp, Jennifer Han, Jenna Wiens, Ellie JC Goldstein, Ebbing Lautenbach, and Evan Snitkin. 2020. Machine learning models to identify patient and microbial genetic factors associated with carbapenem-resistant Klebsiella pneumoniae infection. medRxiv page 2020.07.06.20147306. http://dx.doi.org/10.1101/2020.07.06.20147306.

[140] Ada K. Hagan, Begüm D. Topçuoğlu, Mia E. Gregory, Hazel A. Barton, and Patrick D. Schloss. 2020. Women Are Underrepresented and Receive Differential Outcomes at ASM Journals: A Six-Year Retrospective Analysis. mBio 11(6). http://dx.doi.org/10.1128/mBio.01680-20.

[141] Hadley Wickham. 2016. Ggplot2: Elegant Graphics for Data Analysis. Use R! Springer International Publishing, Cham. ISBN 978-3-319-24275-0 978-3-319-24277-4. http://dx.doi.org/10.1007/978-3-319-24277-4.

[142] Tom J Pollard, Irene Chen, Jenna Wiens, Steven Horng, Danny Wong, Marzyeh Ghassemi, Heather Mattie, Emily Lindemer, and Trishan Panch. 2019. Turning the crank for machine learning: Ease, at what expense? The Lancet Digital Health 1(5):e198–e199. http://dx.doi.org/10.1016/S2589-7500(19)30112-8.

[143] Jerome H. Friedman, Trevor Hastie, and Rob Tibshirani. 2010. Regularization Paths for Generalized Linear Models via Coordinate Descent. Journal of Statistical Software 33(1):1–22. http://dx.doi.org/10.18637/jss.v033.i01.

[144] Alexandros Karatzoglou, Alexandros Smola, Kurt Hornik, and Achim Zeileis. 2004. Kernlab - An S4 Package for Kernel Methods in R. Journal of Statistical Software 11(1):1–20. http://dx.doi.org/10.18637/jss.v011.i09.

[145] Terry Therneau, Beth Atkinson, Brian Ripley (producer of the initial R. port, and maintainer 1999-2017). 2019. Rpart: Recursive Partitioning and Regression Trees .

[146] Andy Liaw and Matthew Wiener. 2002. Classification and Regression by randomForest 2:5.

[147] Tianqi Chen, Tong He, Michael Benesty, Vadim Khotilovich, Yuan Tang, Hyunsu Cho, Kailong Chen, Rory Mitchell, Ignacio Cano, Tianyi Zhou, Mu Li, Junyuan Xie, Min Lin, Yifeng Geng, Yutian Li, and XGBoost contributors (base XGBoost implementation). 2020. Xgboost: Extreme Gradient Boosting .

[148] Hadley Wickham, Romain François, Lionel Henry, Kirill Müller, and RStudio. 2020. Dplyr: A Grammar of Data Manipulation .

[149] Lionel Henry, Hadley Wickham, and RStudio. 2020. Rlang: Functions for Base Types and Core R and 'Tidyverse' Features .

[150] David Meyer, Evgenia Dimitriadou, Kurt Hornik, Andreas Weingessel, Friedrich Leisch, Chih-Chung Chang (libsvm C++-code), and Chih-Chen Lin (libsvm C++-code). 2020. E1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien.

[151] Yachen Yan. 2016. MLmetrics: Machine Learning Evaluation Metrics.

[152] Henrik Bengtsson and R Core Team. 2020. Future.Apply: Apply Function to Elements in Parallel using Futures .

[153] Kelly L Sovacool, Sarah K Lucas, and Patrick D Schloss. 2023. {mothur} Snakemake workflow: A template for microbial amplicon sequence analysis with mothur. SchlossLab. https://github.com/SchlossLab/mothur-snakemake-workflow.

[154] Madeline R. Barron, Kelly L. Sovacool, Lisa Abernathy-Close, Kimberly C. Vendrov, Alexandra K. Standke, Ingrid L. Bergin, Patrick D. Schloss, and Vincent B. Young. 2022. Intestinal Inflammation Reversibly Alters the Microbiota to Drive Susceptibility to Clostridioides difficile Colonization in a Mouse Model of Colitis. mBio 0(0):e01904–22. http://dx.doi.org/10.1128/mbio.01904-22.

[155] Andrew J. Vickers and Elena B. Elkin. 2006. Decision Curve Analysis: A Novel Method for Evaluating Prediction Models. Med Decis Making 26(6):565–574. http://dx.doi.org/10.1177/0272989X06295361.

[156] Abhay Thandavaram, Aneeta Channar, Ansh Purohit, Bijay Shrestha, Deepkumar Patel, Hriday Shah, Kerollos Hanna, Harkirat Kaur, Mohammad S. Alazzeh, Lubna Mohammed, Abhay Thandavaram, Aneeta Channar, Ansh Purohit, Bijay Shrestha, Deepkumar Patel, Hriday Shah, Kerollos S. Hanna, Harkirat Kaur, Mohammad S. Alazzeh, and Lubna Mohammed. 2022. The Efficacy of Bezlotoxumab in the Prevention of Recurrent Clostridium difficile: A Systematic Review. Cureus 14(8). http://dx.doi.org/10.7759/cureus.27979.

[157] Steven J. Mileto, Melanie L. Hutton, Sarah L. Walton, Antariksh Das, Lisa J. Ioannidis, Don Ketagoda, Kylie M. Quinn, Kate M. Denton, Diana S. Hansen, and Dena Lyras. 2022. Bezlotoxumab prevents extraintestinal organ damage induced by Clostridioides difficile infection. Gut Microbes 14(1):2117504. http://dx.doi.org/10.1080/19490976.2022.2117504.

[158] David D. Kim and Anirban Basu. 2021. How Does Cost-Effectiveness Analysis Inform Health Care Decisions? AMA Journal of Ethics 23(8):639–647. http://dx.doi.org/10.1001/amajethics.2021.639.

[159] Graham Loomes and Lynda McKenzie. 1989. The use of QALYs in health care decision making. Social Science & Medicine 28(4):299–308. http://dx.doi.org/10.1016/0277-9536(89)90030-0.

[160] Sylvie M. C. Van Osch, Peter P. Wakker, Wilbert B. Van Den Hout, and Anne M. Stiggelbout. 2004. Correcting Biases in Standard Gamble and Time Tradeoff Utilities. Med Decis Making 24(5):511–517. http://dx.doi.org/10.1177/0272989X04268955.

[161] National Council on Disability. 2019. Quality-Adjusted Life Years and the Devaluation of Life with Disability: Part of the Bioethics and Disability Series https://ncd.gov/sites/default/files/NCD_Quality_Adjusted_Life_Report_508.pdf.

[162] Anna Maria Seekatz, Nasia Safdar, and Sahil Khanna. 2022. The role of the gut microbiome in colonization resistance and recurrent *Clostridioides difficile* infection. Therap Adv Gastroenterol 15:17562848221134396. http://dx.doi.org/10.1177/17562848221134396.

[163] Ana Zhu, Shinichi Sunagawa, Daniel R. Mende, and Peer Bork. 2015. Inter-individual differences in the gene content of human gut bacterial species. Genome Biology 16(1):82. http://dx.doi.org/10.1186/s13059-015-0646-9.

[164] Stilianos Louca, Martin F. Polz, Florent Mazel, Michaeline B. N. Albright, Julie A. Huber, Mary I. O'Connor, Martin Ackermann, Aria S. Hahn, Diane S. Srivastava, Sean A. Crowe, Michael Doebeli, and Laura Wegener Parfrey. 2018. Function and functional redundancy in microbial systems. Nature Ecology & Evolution 2(6):936–943. http://dx.doi.org/10.1038/s41559-018-0519-1.

[165] Better Software for Science. https://sloan.org/programs/digital-technology/better-software-for-science.