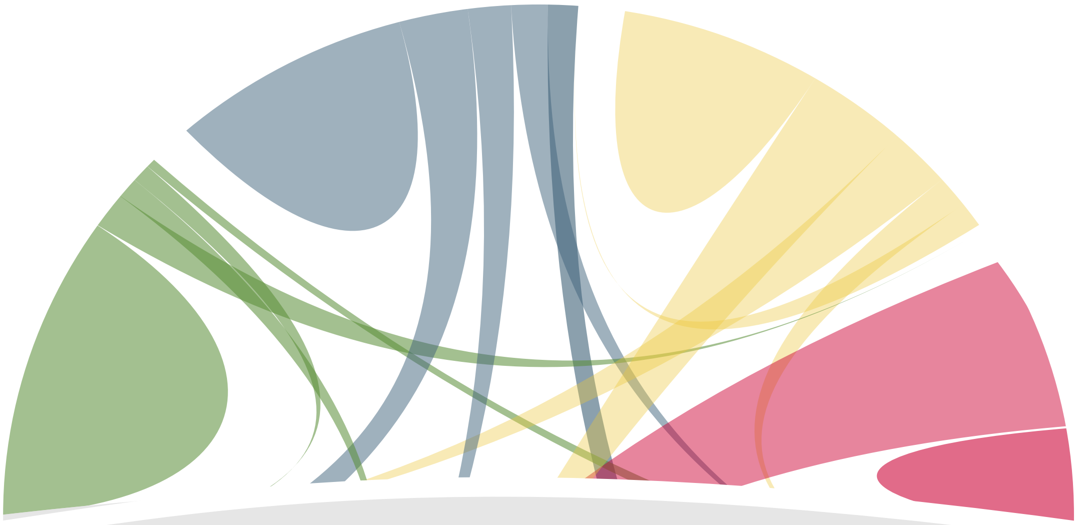

Genome-Wide Source Attribution for Animal and Environmental Pathogen Reservoirs

Nicolas Arning
Supervised by
Daniel Wilson

BIG DATA INSTITUTE

OXFORD INTERDISCIPLINARY BIOSCIENCE DTP

UNIVERSITY OF OXFORD

Table of Contents

Abstract

Gastroenteritis is a food-borne disease commonly caused by the bacterium *Campylobacter jejuni*. Human infection sources can be under-cooked meat, contact with animal faeces or environmental sources like contaminated drinking water. Attributing the source of a gastroenteritis outbreak is important for applying public health regulations. Previous efforts depend on statistical inference through the comparison of genetic sequence between human and source samples. Our aim is to develop a new Machine Learning based application for source attribution to outperform the existing methods and broaden the spectrum of viable input. The choice of Machine Learning as a solution is due to its underuse, albeit reported success, in genomic contexts. This aim was implemented through unbiased choice of tools and use of a k-mer based approach applicable to any form of genetic sequence. This is especially useful with the ever growing amount of whole genome data available. We report a slight increase in accuracy and present a protocol which can be used for input forms unavailable for use with previous methods. Our results confirms the applicability of Machine learning in genomic contexts and can be used as a stepping stone for similar future endeavours.

1. Introduction

1.1 *Campylobacter jejuni* as the Source for Gastro-Enteritis

Gastroenteritis is a food-borne disease accounting for an estimated 2.5 million cases each year in the United States, which is predominantly caused by the bacterium *Campylobacter jejuni* (Sheppard et al., 2009). The human transmission route goes primarily through raw or under-cooked meat or poultry with additional sources of infection through contact with animal faeces or contaminated drinking water (Wilson et al., 2008). Outbreaks in human population are most commonly sporadic with the main source being poultry (Domingues et al., 2012). However, other transmission routes have been reported and the source attribution of an outbreak is of paramount interest to public health regulations (Ravel et al., 2017).

1.2 The Source-Attribution Problem

The two previous protocols established in source-attribution are based on genetic sequence similarity between source-associated samples and human samples (Wilson et al., 2008; Sheppard et al., 2009). Both protocols use Multilocus sequence typing (MLST), a technique established for *C. jejuni* by Dingle et al. (2001). With MLST genetic variation between *C. jejuni* isolates is captured by sequencing seven housekeeping genes. The MLST data from source isolates is compared to human isolates to determine the transmission route through statistical inference. Wilson et al. (2008) developed iSource, which uses an asymmetric island model to assign probability estimates to the source of infection by analysing mutation, migration and recombination events. Sheppard et al. (2009) have used a no admixture model implemented in STRUCTURE (Pritchard et al., 2000) to look at allele frequencies. The application of both models on outbreaks across Europe, Canada and New Zealand (Wilson et al., 2008; Mullner et al., 2009; Sheppard et al., 2009; Strachan et al., 2009; Gras et al., 2012; Kittl et al., 2013; Mossong et al., 2016; Ravel et al., 2017; Rosner et al., 2017; Thépault et al., 2018) shows variation in assigning sources underlining the need for repeated attribution (Thépault et al., 2018).

1.3 Applying Machine Learning to the Source-Attribution Problem

With source-attribution being a recurring obstacle in containing gastroenteritis, this study focuses on developing a new approach using supervised Machine Learning. In supervised Machine Learning an algorithm learns the key attributes of given classes by examining labelled training data (Huang and Yu, 2017). The algorithm can then be applied to unseen data to assign classes given similarity to attributes previously learned. This process called classification is a powerful tool because it can generate predictive models without applying strong assumptions about underlying mechanisms (Angermueller et al., 2016). This is especially useful where data generating mechanisms are exceedingly complex, as in genomics, where Machine Learning is still underused (Schrider and Kern, 2018). We therefore aim to extend and possibly outperform the existing toolbox used for the source-attribution of gastroenteritis using Machine Learning classification.

One way in which Machine Learning can be applied in a genetic context is to break sequences down into overlapping words of length k, called k-mers (as depicted in figure 1). A Machine Learning algorithm can subsequently learn the characteristic composition of k-mers within a class and thereby label unseen data. Machine Learning classification based on k-mer composition have shown to be successful with taxonomic labelling in general metagenomics (Patil et al., 2012; Ounit et al., 2015; Fierst and Murdock, 2017; Jiang et al., 2017) and in bacteria (Wang et al., 2007; Rosen et al., 2008; Vervier et al., 2015; Deneke et al., 2017), making it a promising tool for the source attribution problem. As k-merisation works independently of sequence type, the range of input can be extended from MLST to sequencing reads and whole genome sequences.

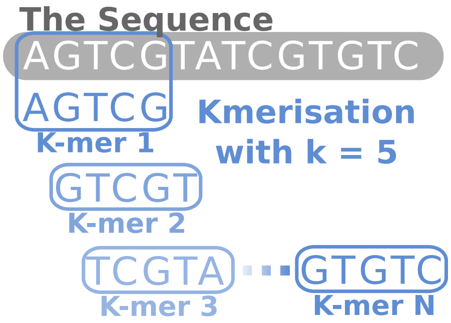

Figure 1: **The K-mer approach** In k-merisation the sequence is broken up into overlapping words of length k. A window of length k is slid across the sequence and every move of the frame produces one k-mer until the end of the sequence is reached.

2. Materials and Methods

2.1 Data Acquisition and Preparation

For any k-mer based approach the value of k is an important choice preceding analysis. A general rule is that with unrelated sequences k-mers should be shorter and for more similar sequences the k should be greater (Wu et al., 2005). As our analysis is aimed at classifying within a species level, a k of 31 was chosen which has also successfully been used in metagenomics (Wood and Salzberg, 2014; Soueidan and Nikolski, 2015) and bacterial genomics (Earle et al., 2016).
Part of the ability of a Machine Learner to perform well on unseen data relies on how much data was seen during training (Domingos, 2012). Therefore we have included a comprehensive data-set collected from MLST samples present in the Public databases for molecular typing and microbial genome diversity (PubMLST, https://pubmlst.org/campylobacter/, Jolley and Maiden (2010)). All samples from possible infection sources containing a high number of replicates (>1000) were chosen for our analysis. The composition of the data-set can be seen in Table 1.

Table 1: **Data-set composition** The table shows the composition of classes within the data-set acquired from PubMLST (Jolley and Maiden, 2010)

Sources of Infection	Number of MLST samples
Chicken	12494
Cattle	2680
Bird	1603
Sheep	1088
Environment	1146
Total	19011

The table from PubMLST contains different alleles for the seven housekeeping genes represented by unique numerical identifiers. The sequence of the alleles was also obtained from the PubMLST webpage and a Python script was written which concatenates allele sequences for every sample. The concatenated sequences are collected into a fasta file and used for k-merisation with DSK (Rizk et al., 2013) (k=31 and default parameters). The k-mer data is stored as a table with samples as rows and all unique k-mers from all samples as columns. The cells contain the presence or absence of each individual k-mer in the sample in binary. The table is converted into a pandas DataFrame (McKinney, 2011) for easy storage and access through Python.

2.2 Optimising and Choosing a Classifier

An unbiased classification through Machine Learning is hindered by an imbalanced data-set, containing a varying number of samples per class (Libbrecht and Noble, 2015). A remedy for imbalance is random under-sampling where only subsets of the more populated classes are taken for training (Huang and Yu, 2017). Here, we limit the size of the subsets to the size of smallest class (sheep = 1088), leading to 5440 samples in total. The Python library imbalanced-learn (Lemaître et al., 2017) was used for random under sampling with a random seed of 0. The random seed ensures that the randomisation is reproducible when repeating our analysis.
In order to gauge the capacity of a classifier to perform on unseen data (called generalising), testing is performed on a labelled data-set not used during training (Domingos, 2012). We therefore split the data-set into training and testing to judge classifier performance. For splitting the data-set and all subsequent Machine Learning related tasks, the Python library scikit-learn (Pedregosa et al., 2011) was used. For the test-training-split the default allocation of 75% training (4080 isolates) and 25 % testing (1360 isolates) was chosen with the random seed of 0.

With trying to introduce as little bias as possible to classifier choice, we will use a comprehensive repertoire of classifiers available from scikit-learn. In the following all included classifiers are described. To facilitate reproducibility the most important adjustable parameters used for parameter optimisation are also named. When using different other Machine Learning tools than scikit-learn, the naming conventions for parameters can differ.

- With the **K-nearest-neighbour** classifier, unseen data is labelled by proximity to the k closest neighbours in feature space (Murphy, 2012). The feature space in our case is spanning all possible k-mers (i.e. all columns) as dimensions. The position of each sample in feature space is defined by the k-mer composition.

  The main parameter is k (not the same as in k-mer), which decides how many neighbours are taken into account when assigning labels to samples.

- The **Ridge Regression** is a form of classification fitting a regression through the datapoints and introducing regularisation. In regularisation the number of parameters stays the same but the magnitude of coefficients vary, which changes the influence of single parameters on the outcome of classification. Regularisation can be used to avoid the use of too many input parameters for decision-making. Including a disproportionate number of k-mers could lead to overfitting, which is adapting a classifier exceedingly well onto a specific set of observations. The classifier is subsequently unlikely to perform well on new data, which is a problem present throughout all of Machine learning.

The main parameter is alpha which denotes to what extend the use of many parameters is penalised.

- A **Support Vector Machine** separates data-points in feature space with a decision boundary (Ben-Hur et al., 2008). The separation is aided by a projection into a higher dimensional space where samples might be more separable. A hyper-plane is fitted through the higher dimensional space as a decision boundary, so that the overall distance to data-points is maximised. The mapping of the data into higher dimensional space is achieved by a kernel.

  The main parameter here is the choice of kernel and C, which is a parameter similar to alpha, penalising excessive feature use. A maximum of 300 iterations of optimising the hyper-plane was used.

- The **Naïve Bayesian** classifier maximises the likelihood of the k-mer composition of a new sample given a certain class (Vervier et al., 2016). The likelihood is defined by the number of samples in a given class similar to the sample examined, divided by the number of members of that class. The frequency of k-mer presence/absence is assumed to be independent for each k-mer within a class. However unrealistic the independence assumption is in the data-set, it can perform well in practice

  The main parameter is alpha which can impose a penalty to reduce overfitting.

- A **Decision Tree** uses succession of binary decisions (like presence or absence of a k-mers) to split the data-set whilst trying maximise the homogeneity within both splits (Weitschek et al., 2014).

  The main parameters are maximum depth, which denotes how many times the data-set is split, and minimum amount of samples allowed per split.

- The **Multilayer Perceptron** is a feed-forward artificial neural network (Korvigo et al., 2018). In a feed-forward network information is only passed from input to output as opposed to nodes being circularly connected. In our case the input is the k-mer composition of the samples and the output is the class labels assigned. The Multilayer Perceptron learns by back-propagation, which compares the predicted label with the true label and makes adjustments to the decision process.

  The main parameter is alpha which can impose a penalty to reduce overfitting.

Our toolbox also contains ensemble classifiers which bundles a number of weak classifiers to improve decision making (Murphy, 2012). This form of meta classification is often combined with randomisation between instances of the ensemble to reduce overfitting.

- A **Random Forest** classifier is made out of an ensemble of decision trees applied to subsets of the data-set (Qiu et al., 2016). By repeatedly and randomly selecting subsets of the data, overfitting is reduced. Through subsampling the columns, the Random Forest algorithm can judge the relative importance of columns to the outcome of classification. This can give an indication of which k-mers are most influential for distinguishing sources of infection.

  The main parameters are maximum depth as in Decision Trees and maximum features which determines the size of the subset of features are that are taken into account in decision making. One thousand decision trees were used as an ensemble for the Random Forest.

- **Extra Trees** is an abbreviation for extra randomised trees and is very similar to a Random Forest classifier, but adds randomisation (Geurts et al., 2006). Instead of finding the best split in one feature of the sub-set implemented in Random Forests, Extra-trees will choose one random split throughout the whole sub-set of features. From all features the one which generates the best homogeneity from the random split is chosen. This is repeated for the maximum depth of the tree. An ensemble of all generated trees is then used for classification.

  The main parameters are the same as in Random Forests.

- **Gradient Boosting** fits a weak learner repeatedly on different sub-samples of the data and tries to optimise a loss function by learning from each past iteration (Friedman, 2002). The loss function assigns a number to quantify the cost of misclassification. In our case the weak learner is a decision tree.

  The main parameters are the same as in Random Forests.

Performance comparison to find the best parameters requires data with known labels exempt from the fitting process. Instead of shrinking the training data further, a 5-fold cross validation is used. In a cross-validation the data is randomly split into 5 subsets of equal size (called folds). 4 of the folds are used for training and 1 for testing, which is repeated until every fold was used for testing. In the cross-validation we try out different parameters for each classifier and find parameters which result in the highest precision. Precision is an estimate of how sure we are that a label is correct once we have assigned it. This is a preferred metric for classification with multiple classes (Libbrecht and Noble, 2015). In the biological context of how our classification is used, precision is also the most desirable. Certainty as to where need to be applied once a prediction has been made is of prime interest.

For all parameters we choose a parameter space depending on limitations in computational resources and what was shown to contain optimal parameters on a subset of the data. There is the possibility of finding an optimal parameter which is actually a local maximum. To avoid assuming a local maximum to be a global maximum, a parameter space is chosen which spans a broad range of values. The parameter space explored for optimisation is shown in table 2.

Table 2: **Parameter Space for Classifier Optimisation** The table shows all paramaters used for every classifier during the classifier optimisation.

Classifier	Parameter 1	Parameter 2
K-nearest-neighbour	k = 1, 2, 3, 4, 5, 10, 15, 20, 30, 50, 100, 200	
Ridge Regression	alpha = 0, 0.001, 0.01, 0.1, 0.3, 0.5, 0.7, 1, 3, 5, 8, 10, 20, 40, 100	
Support Vector Machine	Kernel = Linear function, Radial basis function, Polynomial function, Sigmoid function	C = 1, 2, 3, 4, 5, 7, 10
Naïve Bayesian	alpha = 0, 0.1, 1, 10, 100, 200, 400, 1000, 5000, 8000, 10000, 100000	
Decision Tree	max depth = 3, 4, 5, 10, 15, 20	minimum samples / split = 3, 4, 5, 10, 15
Random Forest	max depth = 5, 10, 15, 20, 30, 50	maximum features = 5, 10, 15, 25, 50, 100
Extra Trees	max depth = 5, 10, 15, 20, 30, 50	maximum features = 5, 10, 15, 25, 50, 100
Gradient Boosting	max depth = 5, 10, 15, 20	maximum features = 5, 10, 25, 100
Multilayer Perceptron	alpha = 0.001, 0.01, 0.05, 0.1, 1, 10, 20, 30, 50, 100	

After optimising the toolbox of classifiers on the training data-set, the classifiers are compared in order to find the one performing best. All classifiers are used to assign labels on the test data-set with known classes and multiple scores are measured to estimate performance. To get more robust scores and estimates of uncertainty we use bootstrapping while testing. In bootstrapping random samples are taken with replacement from the data until a set as big as the initial size is obtained. Some samples are randomly excluded, while others are included various times. In our testing, classifiers are run on 200 bootstrap replicates and scores are computed. This will give prediction of 200 data-sets with slight variation for which average scores are calculated and standard deviation is measured.

To understand how scorers estimate performance, a few scoring concepts have to be understood

- **True positives (TP)**, where the sample is predicted as a certain label and truly has this label.
- **True Negatives (TN)**, where the sample is not predicted as a certain label and truly does not have this label.
- **False Positives (FP)**, where the sample is predicted as a certain label, but does not truly have that label.
- **False Negatives (FN)**, where the sample is not predicted as a certain label, but truly has that label.

Table 3 shows a simple table of these concepts in relation to each other which is called a confusion matrix. In classification of multiple classes there is a confusion matrix for every class.

Table 3: **Example for a Confusion Matrix** The table shows the concepts of True Positives, True Negatives, Fale Positives and False negatives in relation to the true labelling of the data against the predicted labelling of the data

		Truth	
		Has label X	Does not have label X
Predicted	Has label X	True Positive	False Positive
	Does not have label X	False Negative	True Negative

We use multiple scorers to determine the best performing classifier:

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$$

$$Precision\ or\ Positive\ Predictive\ Value = \frac{TP}{TP + FP}$$

$$Recall\ or\ Sensitivity = \frac{TP}{TP + FN}$$

$$Negative\ Predictive\ Value = \frac{TN}{TN + FN}$$

$$Specificity = \frac{TN}{TN + FP}$$

$$F1 = \frac{2 * TP}{2 * TP + FP + FN}$$

Additional to performance scorers, the speed used for testing over bootstrap replicates is measured and scaled over classifiers. The fastest classifier has a value of 1 and the slowest a value of 0. As a best classifier we take into account all measures with an emphasis on precision as previously explained.
The performance of a classifier can also be analysed by the numbers assigned to each case within the confusion matrix (TP, TN, FP, FN). To summarise multi-class-classification the table seen in Table 3 needs to be extended by including every class. The extended confusion matrix shows the misclassification of each class into each other class. A frequent misclassification of infection source A into infection source B could indicate a similar k-mer composition between sources.
The best performing classifier can be used on unseen data (called generalisation) to assign labels denoting the probable source of infection. Generalisation requires new data to be pre-processed as described in our data preparation step. In preparation the only modification is that the columns of the k-mer table need to be the limited (or extended) to the k-mers used in training the classifier.
A flowchart of our work-flow is depicted in Figure 2

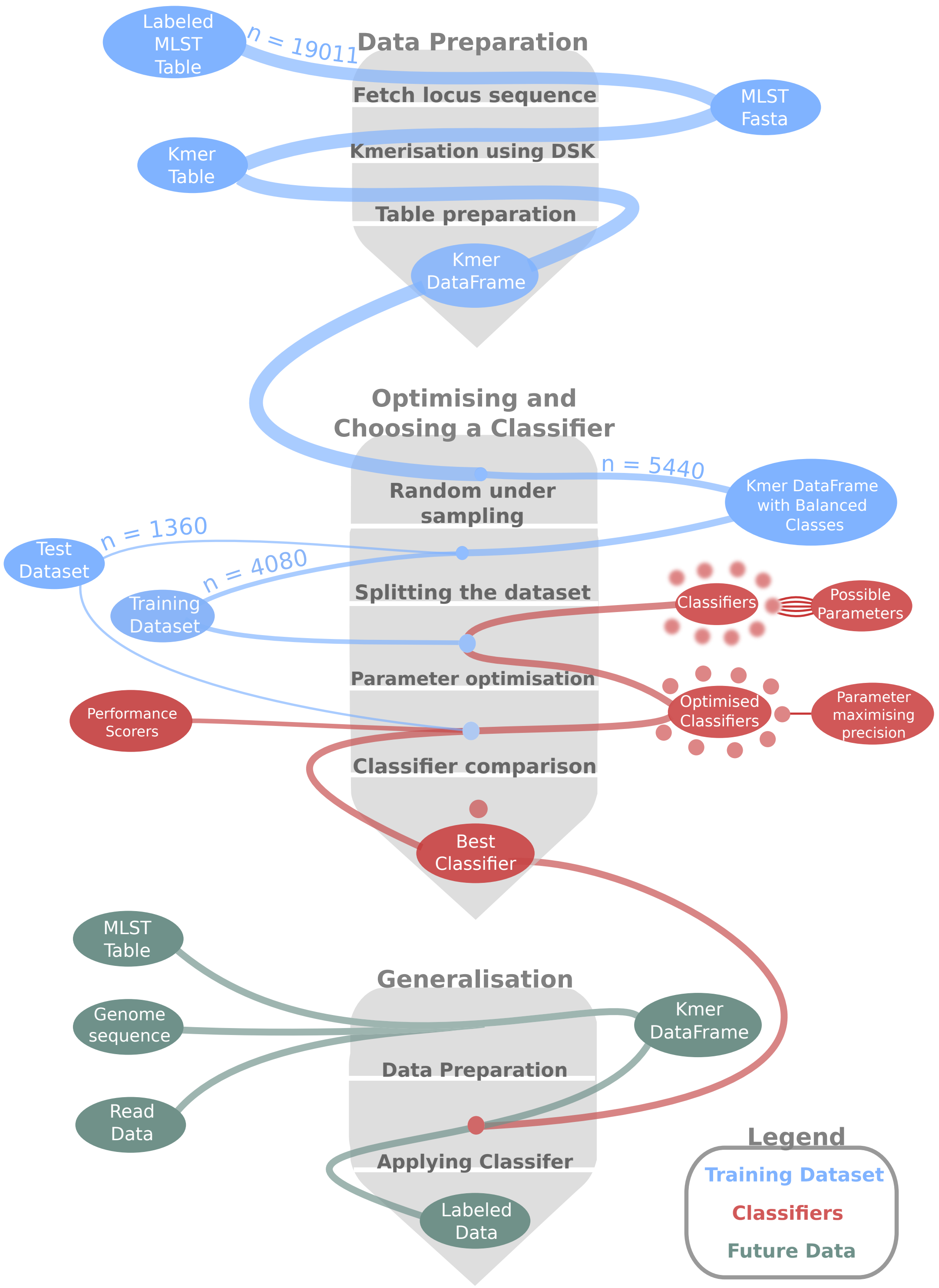

Figure 2: **Work-flow of our Analysis** Here the trajectory of the main components of our analysis: the training data-set, the classifiers and future data are shown through main steps of our analysis: data preparation, optimising and choosing a classifier, and generalisation.

3. Results and Discussion

3.1 Results

The binary table of k-mer presence absence has 19011 rows (samples) and 56235 columns (all k-mers present in the PubMLST data-set). The table is sparse, showing few non-zero values which can be seen in figure 3. After data preparation, balancing the data-set reduces the number of rows to 5440 and train-test splitting produces two tables with 4080 and 1360 rows respectively.

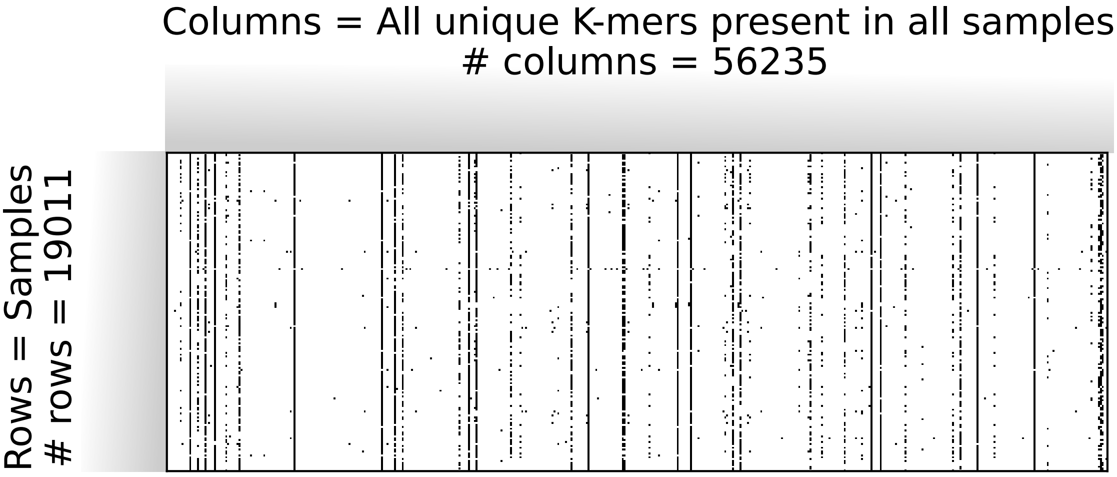

Figure 3: **Sparsity plot** The plot shows the occurrence of non-zero values (black dots) throughout the k-mer table obtained from the data preparation step. The table shows very few non-zero values compared to zero values and can therefore be considered sparse

Exemplary for one of the 10 classifiers, the optimisation process of Extra-Trees classifier can be seen in figure 4. The ascending precision before the optimal value (maximum features = 15 and maximum depth = 5) and descending precision after indicates that the parameter space contains the optimal parameter. However, overall the choice of parameter does not seem to greatly influence the performance of the classifier.

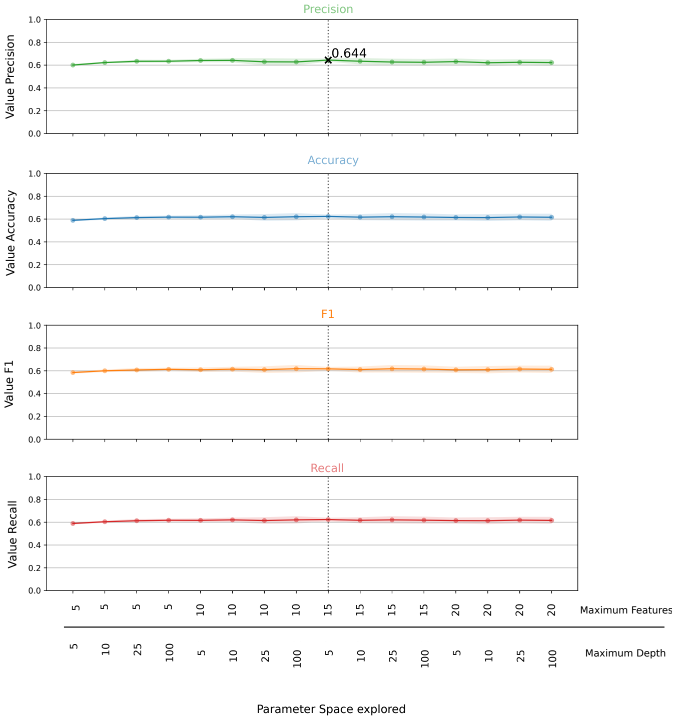

Figure 4: **Optimisation of the Extra-Trees classifier** This table shows the influence of parameters on Precision, Accuracy, F1 and Recall. On the x-axis the parameter space that was explored for the Extra-trees classifier can be seen. On the y-axis the values of the different scorers for classification performance can be seen. Optimal precision is marked by an x which is also plotted as a vertical line through all plots.

The comparison of every classifier is depicted in Figure 5 and Table 4. The Extra-Trees classifier performs best amongst all classifiers and throughout all different scoring categories, except Negative Predictive Value where it is second to the Random Forest classifier. The Extra-Trees classifier is a modification of the Random Forest classifier with added randomisation. The good performance of a tree based ensemble methods like Random Forest and Extra-Trees on genomic problems is not suprising and has been reported (Austerlitz et al., 2009; Deneke et al., 2017; Fierst and Murdock, 2017; Qiu et al., 2016; Remita et al., 2017; Schrider and Kern, 2018; Yan et al., 2011). The success of tree ensemble methods is linked to their ability to handle correlation as well as interaction of features, which is often the case with genomic features (Chen and Ishwaran, 2012).

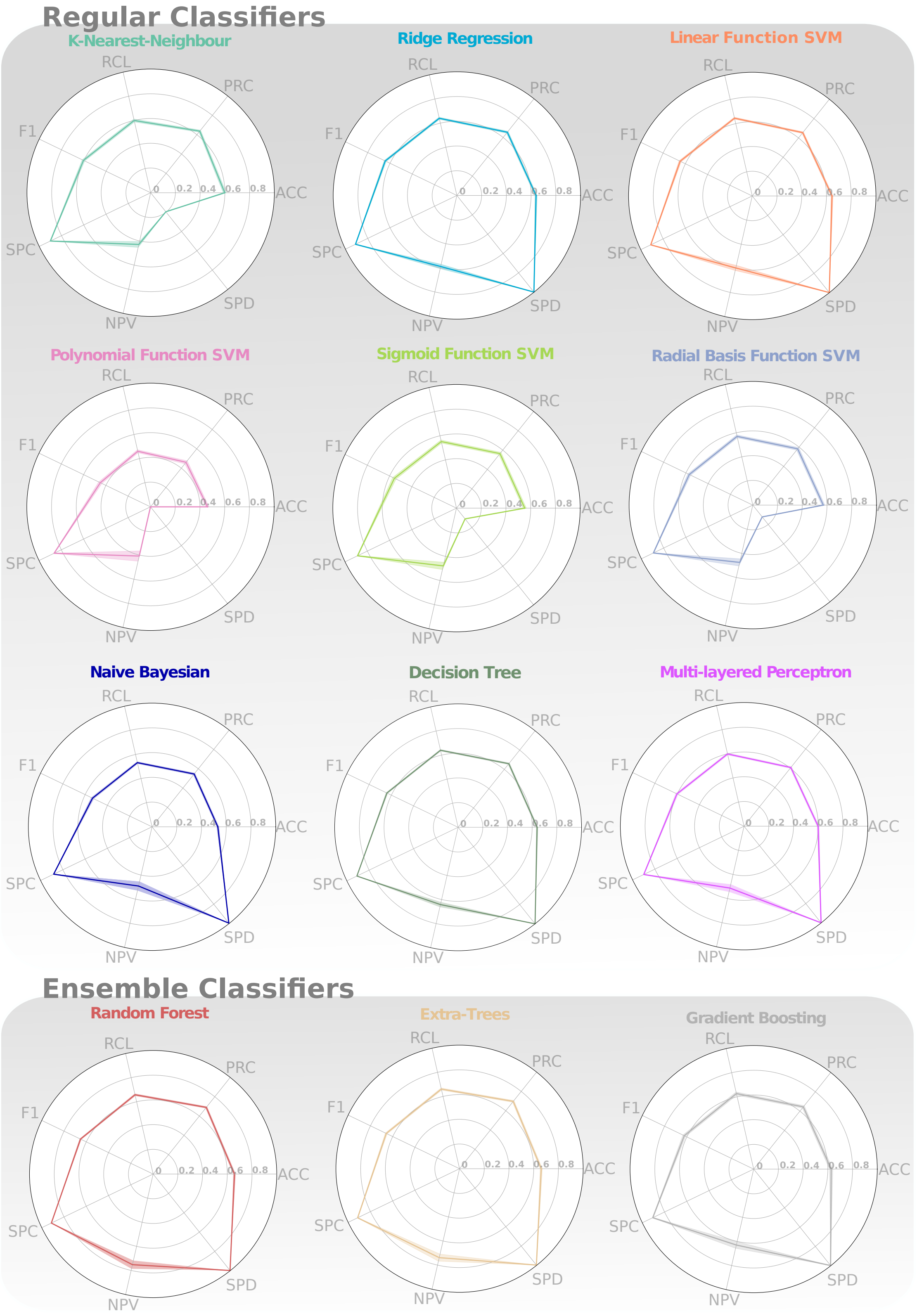

Figure 5: **Radar plot of classifier comparison** The radar plots show the performance of our classifiers as measured by Recall (RCL), Precision (PRC), Accuracy (ACC), Speed (SPD), Negative Predictive Value (NPV), Specificity (SPC) and F1 on a scale from 0 to 1. The line of each radar plot denotes the average of the score whereas the area under it shows the standard deviation. ,SVM = Support Vector Machine

Table 4: **Table of classifier comparison** This table shows the performance of eveRy classifier as measured by Recall, Precision, Accuracy, Speed, Negative Predictive Value, Specificity and F1. Classifiers are coloured as in Figure 5.The Meta-classifiers separated by a horizontal line perform better than all regular classifiers. Extra-Trees which is the best performing classifier is highlighted in grey. NPV = Negative Predicitve Value, SVM = Support Vector Machine

Classifiers	Accuracy	Precision	Recall	F1	Specificity	NPV	Speed
K-Nearest-Neighbour	0.60	0.64	0.60	0.60	0.90	0.43	0.20
Ridge Regression	0.64	0.65	0.64	0.64	0.91	0.59	1.00
Linear SVM	0.65	0.66	0.65	0.65	0.91	0.60	1.00
Polynomial Function SVM	0.46	0.46	0.46	0.45	0.87	0.41	0.00
Sigmoid Function SVM	0.55	0.56	0.55	0.56	0.89	0.48	0.11
Radial Basis Function SVM	0.57	0.58	0.57	0.57	0.89	0.48	0.12
Naive Bayesian	0.53	0.55	0.53	0.53	0.88	0.49	1.00
Decision Tree	0.64	0.66	0.64	0.64	0.91	0.64	1.00
Multilayered Perceptron	0.60	0.61	0.60	0.60	0.90	0.51	1.00
Random Forest	0.66	0.69	0.66	0.65	0.91	0.75	1.00
Extra-Trees	0.66	0.70	0.66	0.66	0.92	0.74	1.00
Gradient Boosting	0.63	0.64	0.63	0.62	0.91	0.63	1.00

When looking at the speed comparison of classifiers, the scaling is inflated for maximum values of one. This indicated that the Polynomial Function Support Vector Machine as the minimum value of the scale is a lot slower than all other classifiers. The scaling should be improved to account for that or the Polynomial Function Support Vector Machine should be excluded from the comparison as it also does not perform well on our data-set.
The confusion matrix of the Extra-Trees classifier (as shown in Figure 6 and Figure 7) shows the most frequent misclassification between the *C. jejuni* samples from cattle and sheep. This has also been reported by Wilson et al. (2008) with cattle and sheep exhibiting the lowest amount of genetic differentiation amongst all sources. A high sequence similarity, and hence similar k-mer composition, could be a reason for the frequent misclassification. Gene flow between *C. jejuni* populations of cows and sheep due to being held in proximity could be an explanation for the high sequence similarity and thus rate of misclassification.

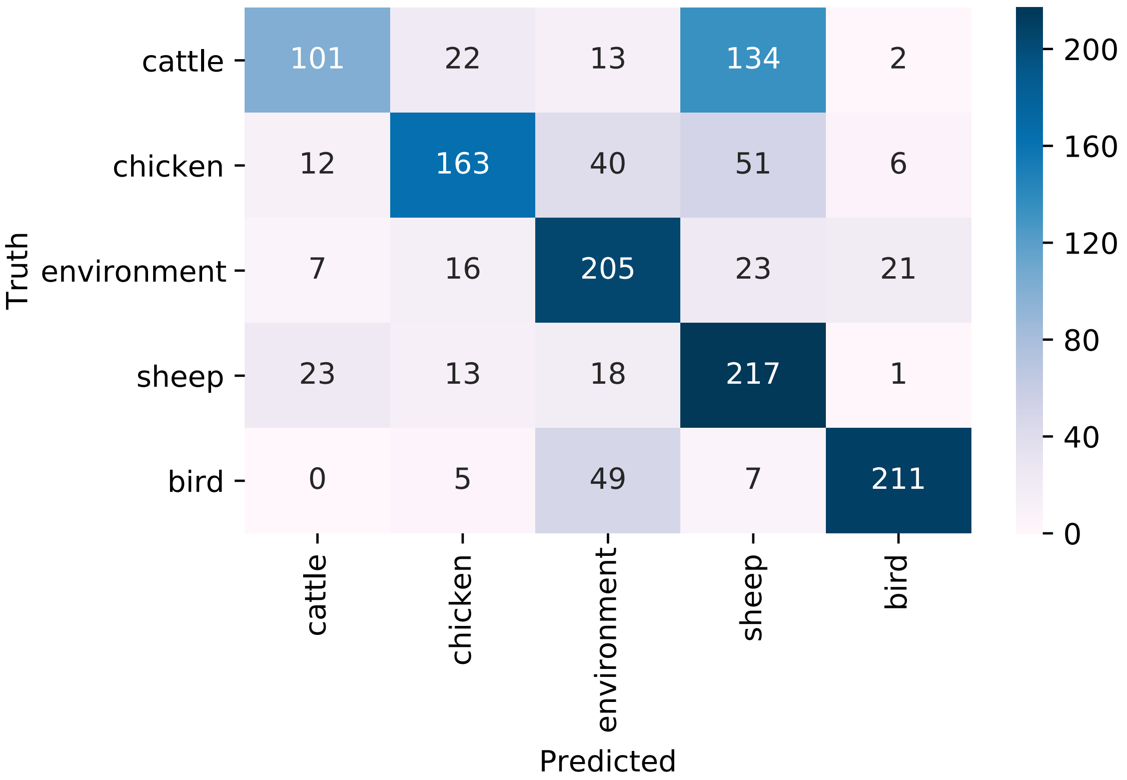

Figure 6: **Heatmap of the Confusion Matrix produced by the Extra-Trees Classifier** The heatmap shows the predicted class labels (x-axis) against the true class labels (y-axis). The colour indicates how many instances are in each case. The diagonal shows correct classifications where truth and prediction coincides. The other cells show the misclassification of each class into every other class.

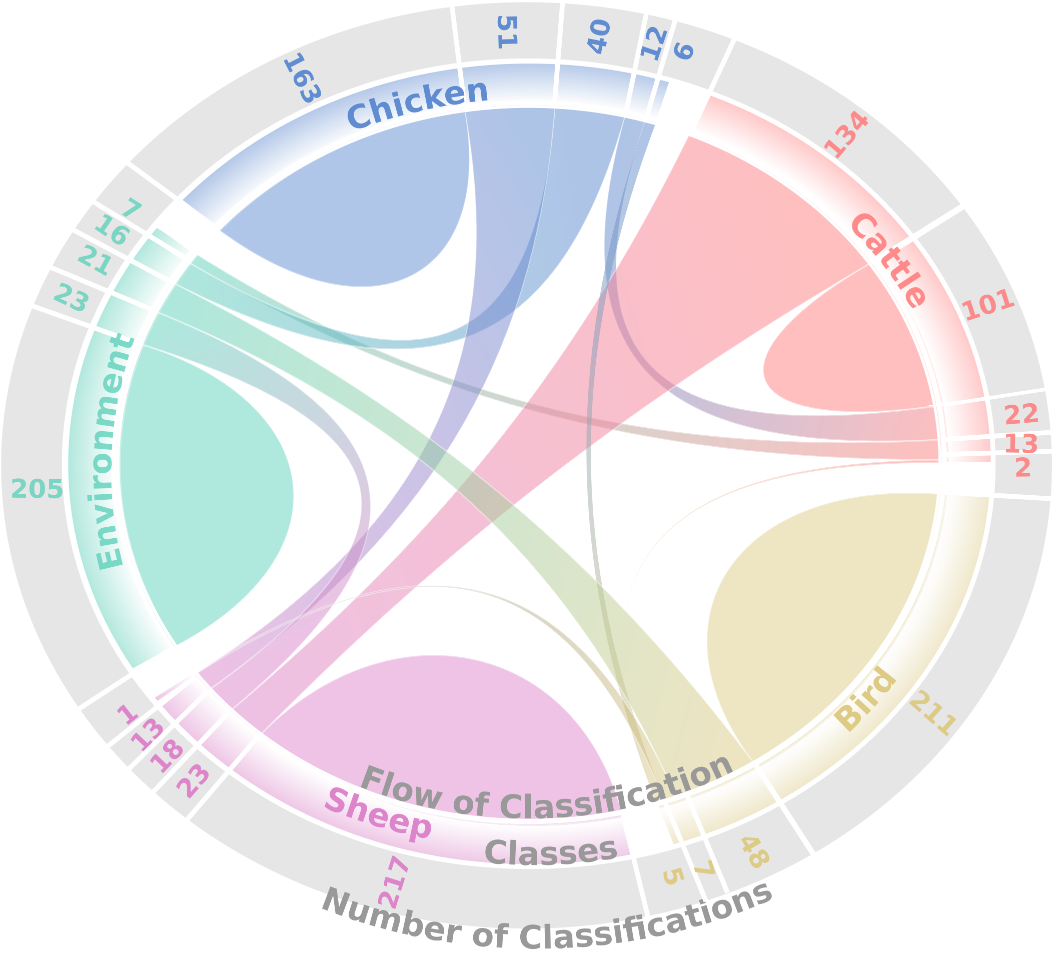

Figure 7: **Chord diagram of Confusion matrix**. The chord diagram is a more visually accessable representation of the confusion matrix depicted in Figure 6. Class labels are shown on the exterior of the circle and interior connections show the proportions of classifications. Connections between two different classes show misclassifications with the thickness indicating how many samples were misclassified. The directionality of connections is away from the class label. I. e. the number under cattle which is above the connection between cattle and sheep denotes how many cattle samples were classified as sheep. The number on the sheep side of the connection denotes how many sheep were classified as cattle. The connections of class labels with itself (semi circles) denote how many samples were correctly assigned.

3.2 Comparison to Previous Methods

In comparison to previous source-attribution methods of Wilson et al. (2008) and Sheppard et al. (2009), who achieved 64% accuracy and 61.5% average accuracy over all classes on their own data-sets, our top classifier performs slightly better at 66% average accuracy. This comparison is far from perfect as the comparison is on different data-sets and training-test-splits (75% 25% here vs. 50% and 50% in both other studies). A comparison would only be viable comparing performance on the same data-sets, scorers and training-test-split, which we have been unable to do due to time constraints.

3.3 Room for Improvement

The data-set used could be modified to improve the performance of the classification. A factor that introduces noise into our data-set is the difference in time and space of collection. Samples from Africa, Asia, Europe, North America, Oceania and South America as well as collections dating as far back as 1980 and as recent as 2018 are included. Due to the rapid evolution and gene transfer exhibited in bacterial genomes (Rocha, 2008) the sequence similarity shared within classes is expected to erode, making classification more difficult. The data-set can be improved by either sub-selecting to decrease the variation in time and collection or generating more MLST samples with similar collection locations and dates. The classification could also be improved by looking only at k-mers which are biologically more likely to vary between host. For example limiting the analysis to k-mers from genomic regions co-evolving with host species could reduce noise in the data-set and improve performance.

The classifiers can also be modified to achieve more accurate classification. With additional computational resources, the complexity of some classifiers could be increased. For example vaster networks can be used for the Multilayered Perceptron or bigger ensembles can be included in ensemble classifiers, which might increase performance. Another improvement could be the choice of kernel for the Support Vector Machine classifier, which can be adjusted towards genetic application. Ben-Hur et al. (2008) report that sequence similarity metrics like BLAST scores and Smith-Waterman algorithm can be used as a basis for kernels which could improve performance as a more meaningful distance for genetic sequences. Again, this was implemented but not carried out due to time constraints.

4. Outlook and Conclusions

4.1 Future Applications

Due to the development of more affordable sequencing, the wealth of bacterial whole genome sequences is growing exponentially, leading to an ever increasing potential to understand bacterial infection (Koonin and Wolf, 2008; Heather and Chain, 2016). The number of available whole genome sequences has grown by a hundredfold between 2004 and 2014 alone with over 30.000 bacterial genomes available in 2015 (Land et al., 2015). With added potential however, comes the caveat of complexity which leads to an increase in computation time necessary to fit complex evolutionary models (Croucher et al., 2015). Previous methods in source attribution, which infer source from evolutionary events (Wilson et al., 2008; Sheppard et al., 2009), are burdened by this caveat. Our method however is not only easily applicable, but after data preparation will need almost no additional time or resources to account for the added complexity of using whole genomes. The only modification is in preparing data, where the k-mer composition has to be limited (or extended) to the k-mers used in training our classifier. After this, the classifier runs on the same amount of columns per sample and thus only scales in time with data-set size and not sequencing type. We can efficiently make use of ever growing potential through whole genome data to analyse *C. jejuni* transmission, which was previously impossible. Additionally, the spectrum of viable genetic input is essentially limitless as long as the sequences are at least of 31 bp length, making it also applicable to raw read data. However true in concept, the applicability of our method to sequence data other than MLST needs to be validated first. This can be done by acquiring *C. jejuni* genomes with source labels, which can be downloaded from the Integrated Microbial Genomes database (Chen et al., 2018). The accuracy of predicted labels against the true labels can then be assessed. This was attempted but has not been carried out at the time of submitting this report.

4.2 Biological Relevance

With the capability of the Random Forest classifier, the relative importance of columns and thus k-mers into the classification can be assessed. As the classes are different hosts of *C. jejuni* the most important k-mers for classification are likely to be responsible for host adaption. Using the most important k-mers and tracing the position in the *C. jejuni* genome could reveal genes, RNAs, or protein domains important for host adaption in *C. jejuni*.

With an unbiased choice of classification and the applicability of the k-mer approach to any sequencing data, our approach can be applied to other genomic classification problems with only few modifications. For example, using different antimicrobial resistances as classes could quickly classify unknown bacteria. A more targeted treatment with antimicrobials could be administered. Through the relative feature importance generated by the Random Forest classifier, the genes responsible for antimicrobial resistance could be uncovered.

4.3 Machine Learning as a Solution to the Source-Attribution Problem

Having outperformed existing methods, and potentially broadened the spectrum of viable input data, this study indicates that Machine learning is highly applicable to the source-attribution problem. With better data-sets and classifiers adjusted towards genetic applications, the performance could be improved further. As Machine Learning is still underused in genomics (Schrider and Kern, 2018) our results seem promising as to what can still be achieved.Through our unbiased selection of classifiers and the wide applicability of the k-mer approach our methods are easily amendable to facilitate the advancement of Machine Learning in genomics.

References

Angermueller, C., Pärnamaa, T., Parts, L., and Stegle, O. (2016). Deep learning for computational biology. *Molecular Systems Biology*, 12(7):878.

Austerlitz, F., David, O., Schaeffer, B., Bleakley, K., Olteanu, M., Leblois, R., Veuille, M., and Laredo, C. (2009). DNA barcode analysis: a comparison of phylogenetic and statistical classification methods. *BMC Bioinformatics*, 10(14):S10.

Ben-Hur, A., Ong, C. S., Sonnenburg, S., Schölkopf, B., and Rätsch, G. (2008). Support Vector Machines and Kernels for Computational Biology. *PLOS Computational Biology*, 4(10):e1000173.

Chen, M. L., Doddi, A., Royer, J., Freschi, L., Schito, M., Ezewudo, M., Kohane, I. S., Beam, A., and Farhat, M. (2018). Deep Learning Predicts Tuberculosis Drug Resistance Status from Whole-Genome Sequencing Data. *bioRxiv*, page 275628.

Chen, X. and Ishwaran, H. (2012). Random Forests for Genomic Data Analysis. *Genomics*, 99(6):323–329.

Croucher, N. J., Page, A. J., Connor, T. R., Delaney, A. J., Keane, J. A., Bentley, S. D., Parkhill, J., and Harris, S. R. (2015). Rapid phylogenetic analysis of large samples of recombinant bacterial whole genome sequences using Gubbins. *Nucleic Acids Research*, 43(3):e15–e15.

Deneke, C., Rentzsch, R., and Renard, B. Y. (2017). PaPrBaG: A machine learning approach for the detection of novel pathogens from NGS data. *Scientific Reports*, 7:39194.

Dingle, K. E., Colles, F. M., Wareing, D. R., Ure, R., Fox, A. J., Bolton, F. E., Bootsma, H. J., Willems, R. J., Urwin, R., and Maiden, M. C. (2001). Multilocus sequence typing system for Campylobacter jejuni. *Journal of Clinical Microbiology*, 39(1):14–23.

Domingos, P. (2012). A Few Useful Things to Know About Machine Learning. *Commun. ACM*, 55(10):78–87.

Domingues, A. R., Pires, S. M., Halasa, T., and Hald, T. (2012). Source attribution of human campylobacteriosis using a meta-analysis of case-control studies of sporadic infections. *Epidemiology & Infection*, 140(6):970–981.

Earle, S. G., Wu, C.-H., Charlesworth, J., Stoesser, N., Gordon, N. C., Walker, T. M., Spencer, C. C. A., Iqbal, Z., Clifton, D. A., Hopkins, K. L., Woodford, N., Smith,

E. G., Ismail, N., Llewelyn, M. J., Peto, T. E., Crook, D. W., McVean, G., Walker, A. S., and Wilson, D. J. (2016). Identifying lineage effects when controlling for population structure improves power in bacterial association studies. *Nature Microbiology*, 1(5):16041.

Fierst, J. L. and Murdock, D. A. (2017). Decontaminating eukaryotic genome assemblies with machine learning. *BMC Bioinformatics*, 18(1):533.

Friedman, J. H. (2002). Stochastic gradient boosting. *Computational Statistics & Data Analysis*, 38(4):367–378.

Geurts, P., Ernst, D., and Wehenkel, L. (2006). Extremely randomized trees. *Machine Learning*, 63(1):3–42.

Gras, L. M., Smid, J. H., Wagenaar, J. A., Boer, A. G. d., Havelaar, A. H., Friesema, I. H. M., French, N. P., Busani, L., and Pelt, W. v. (2012). Risk Factors for Campylobacteriosis of Chicken, Ruminant, and Environmental Origin: A Combined Case-Control and Source Attribution Analysis. *PLOS ONE*, 7(8):e42599.

Heather, J. M. and Chain, B. (2016). The sequence of sequencers: The history of sequencing DNA. *Genomics*, 107(1):1–8.

Huang, H. and Yu, B. (2017). Data Wisdom in Computational Genomics Research. *Statistics in Biosciences*, 9(2):646–661.

Jiang, Y., Wang, J., Xia, D., and Yu, G. (2017). EnSVMB: Metagenomics Fragments Classification using Ensemble SVM and BLAST. *Scientific Reports*, 7(1):9440.

Jolley, K. A. and Maiden, M. C. (2010). BIGSdb: Scalable analysis of bacterial genome variation at the population level. *BMC Bioinformatics*, 11:595.

Kittl, S., Heckel, G., Korczak, B. M., and Kuhnert, P. (2013). Source Attribution of Human Campylobacter Isolates by MLST and Fla-Typing and Association of Genotypes with Quinolone Resistance. *PLOS ONE*, 8(11):e81796.

Koonin, E. V. and Wolf, Y. I. (2008). Genomics of bacteria and archaea: the emerging dynamic view of the prokaryotic world. *Nucleic Acids Research*, 36(21):6688–6719.

Korvigo, I., Afanasyev, A., Romashchenko, N., and Skoblov, M. (2018). Generalising better: Applying deep learning to integrate deleteriousness prediction scores for whole-exome SNV studies. *PLOS ONE*, 13(3):e0192829.

Land, M., Hauser, L., Jun, S.-R., Nookaew, I., Leuze, M. R., Ahn, T.-H., Karpinets, T., Lund, O., Kora, G., Wassenaar, T., Poudel, S., and Ussery, D. W. (2015). Insights from 20 years of bacterial genome sequencing. *Functional & Integrative Genomics*, 15(2):141–161.

Lemaître, G., Nogueira, F., and Aridas, C. K. (2017). Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning. *Journal of Machine Learning Research*, 18(17):1–5.

Libbrecht, M. W. and Noble, W. S. (2015). Machine learning applications in genetics and genomics. *Nature Reviews Genetics*, 16(6):321–332.

McKinney, W. (2011). pandas: a Foundational Python Library for Data Analysis and Statistics. page 9.

Mossong, J., Mughini-Gras, L., Penny, C., Devaux, A., Olinger, C., Losch, S., Cauchie, H.-M., van Pelt, W., and Ragimbeau, C. (2016). Human Campylobacteriosis in Luxembourg, 2010–2013: A Case-Control Study Combined with Multilocus Sequence Typing for Source Attribution and Risk Factor Analysis. *Scientific Reports*, 6:20939.

Mullner, P., Spencer, S. E. F., Wilson, D. J., Jones, G., Noble, A. D., Midwinter, A. C., Collins-Emerson, J. M., Carter, P., Hathaway, S., and French, N. P. (2009). Assigning the source of human campylobacteriosis in New Zealand: A comparative genetic and epidemiological approach. *Infection, Genetics and Evolution*, 9(6):1311–1319.

Murphy, K. (2012). *Machine Learning: A Probabilistic Perspective*. MIT Press, Cambridge, MA.

Ounit, R., Wanamaker, S., Close, T. J., and Lonardi, S. (2015). CLARK: fast and accurate classification of metagenomic and genomic sequences using discriminative k-mers. *BMC Genomics*, 16:236.

Patil, K. R., Roune, L., and McHardy, A. C. (2012). The PhyloPythiaS Web Server for Taxonomic Assignment of Metagenome Sequences. *PLOS ONE*, 7(6):e38581.

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

Pritchard, J. K., Stephens, M., and Donnelly, P. (2000). Inference of Population Structure Using Multilocus Genotype Data. *Genetics*, 155(2):945–959.

Qiu, Z., Cheng, Q., Song, J., Tang, Y., and Ma, C. (2016). Application of Machine Learning-Based Classification to Genomic Selection and Performance Improvement. In *Intelligent Computing Theories and Application*, Lecture Notes in Computer Science, pages 412–421. Springer, Cham.

Ravel, A., Hurst, M., Petrica, N., David, J., Mutschall, S. K., Pintar, K., Taboada, E. N., and Pollari, F. (2017). Source attribution of human campylobacteriosis at the point of exposure by combining comparative exposure assessment and subtype comparison based on comparative genomic fingerprinting. *PLOS ONE*, 12(8):e0183790.

Remita, M. A., Halioui, A., Malick Diouara, A. A., Daigle, B., Kiani, G., and Diallo, A. B. (2017). A machine learning approach for viral genome classification. *BMC Bioinformatics*, 18.

Rizk, G., Lavenier, D., and Chikhi, R. (2013). DSK: k-mer counting with very low memory usage. *Bioinformatics (Oxford, England)*, 29(5):652–653.

Rocha, E. P. C. (2008). The organization of the bacterial genome. *Annual Review of Genetics*, 42:211–233.

Rosen, G., Garbarine, E., Caseiro, D., Polikar, R., and Sokhansanj, B. (2008). Metagenome Fragment Classification Using N-Mer Frequency Profiles. *Advances in Bioinformatics*, 2008.

Rosner, B. M., Schielke, A., Didelot, X., Kops, F., Breidenbach, J., Willrich, N., Gölz, G., Alter, T., Stingl, K., Josenhans, C., Suerbaum, S., and Stark, K. (2017). A combined case-control and molecular source attribution study of human Campylobacter infections in Germany, 2011–2014. *Scientific Reports*, 7(1):5139.

Schrider, D. R. and Kern, A. D. (2018). Supervised Machine Learning for Population Genetics: A New Paradigm. *Trends in Genetics*, 34(4):301–312.

Sheppard, S. K., Dallas, J. F., Strachan, N. J. C., MacRae, M., McCarthy, N. D., Wilson, D. J., Gormley, F. J., Falush, D., Ogden, I. D., Maiden, M. C. J., and Forbes, K. J. (2009). Campylobacter Genotyping to Determine the Source of Human Infection. *Clinical Infectious Diseases*, 48(8):1072–1078.

Soueidan, H. and Nikolski, M. (2015). Machine learning for metagenomics: methods and tools. *arXiv:1510.06621 [q-bio]*. arXiv: 1510.06621.

Strachan, N. J. C., Gormley, F. J., Rotariu, O., Ogden, I. D., Miller, G., Dunn, G. M., Sheppard, S. K., Dallas, J. F., Reid, T. M. S., Howie, H., Maiden, M. C. J., and Forbes, K. J. (2009). Attribution of Campylobacter Infections in Northeast Scotland to Specific Sources by Use of Multilocus Sequence Typing. *The Journal of Infectious Diseases*, 199(8):1205–1208.

Thépault, A., Rose, V., Quesne, S., Poezevara, T., Béven, V., Hirchaud, E., Touzain, F., Lucas, P., Méric, G., Mageiros, L., Sheppard, S. K., Chemaly, M., and Rivoal, K. (2018). Ruminant and chicken: important sources of campylobacteriosis in France despite a variation of source attribution in 2009 and 2015. *Scientific Reports*, 8(1):9305.

Vervier, K., Mahé, P., Tournoud, M., Veyrieras, J.-B., and Vert, J.-P. (2015). Large-scale Machine Learning for Metagenomics Sequence Classification. *arXiv:1505.06915 [cs, q-bio, stat]*. arXiv: 1505.06915.

Vervier, K., Mahé, P., Tournoud, M., Veyrieras, J.-B., and Vert, J.-P. (2016). Large-scale machine learning for metagenomics sequence classification. *Bioinformatics*, 32(7):1023–1032.

Wang, Q., Garrity, G. M., Tiedje, J. M., and Cole, J. R. (2007). Naïve Bayesian Classifier for Rapid Assignment of rRNA Sequences into the New Bacterial Taxonomy. *Applied and Environmental Microbiology*, 73(16):5261–5267.

Weitschek, E., Fiscon, G., and Felici, G. (2014). Supervised DNA Barcodes species classification: analysis, comparisons and results. *BioData Mining*, 7:4.

Wilson, D. J., Gabriel, E., Leatherbarrow, A. J. H., Cheesbrough, J., Gee, S., Bolton, E., Fox, A., Fearnhead, P., Hart, C. A., and Diggle, P. J. (2008). Tracing the Source of Campylobacteriosis. *PLOS Genetics*, 4(9):e1000203.

Wood, D. E. and Salzberg, S. L. (2014). Kraken: ultrafast metagenomic sequence classification using exact alignments. *Genome Biology*, 15(3):R46.

Wu, T.-J., Huang, Y.-H., and Li, L.-A. (2005). Optimal word sizes for dissimilarity measures and estimation of the degree of dissimilarity between DNA sequences. *Bioinformatics*, 21(22):4125–4132.

Yan, R., Boutros, P. C., and Jurisica, I. (2011). A tree-based approach for motif discovery and sequence classification. *Bioinformatics (Oxford, England)*, 27(15):2054–2061.