# Detection of Predictable Temporal Changes in Multidimensional Biological Sequences

A Thesis

submitted to the designated
by the General Assembly of Special Composition
of the Department of Computer Science and Engineering
Examination Committee

by

Nestor Timonidis

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

WITH SPECIALIZATION
IN TECHNOLOGIES - APPLICATIONS

University of Ioannina

July 2017

# Dedication

This thesis is dedicated to my family, which supported me all those years as a student, and whose financial support made the elaboration of this thesis possible.

# Acknowledgements

I would like to thank my thesis advisor Professor Aristidis Likas of Department of Computer Science and Engineering at the University of Ioannina. The remarks of Prof. Likas proved valuable for interpreting the results and comprehending the subject as well as the data. Moreover, this thesis would have not been completed without his insights regarding the subject throughout the whole process, which formed the directions we took and the outcome of this research.

Furthermore, I would like to thank Professor Sacha A.F.T. van Hijum from the Centre for Molecular and Biomolecular Informatics of the Radboud University Medical Centre in Nijmegen, Netherlands, whose expertise in Biostatistics aided us in using Beta Diversity threshold values for detecting spike occurrences in the analysis described at chapter 3.

Author,
Nestor Timonidis.

# Table of Contents

# List of Figures

# List of Tables

# List of Algorithms

# Abstract

Nestor Timonidis, M.Sc. in Computer Science, Department of Computer Science and Engineering, University of Ioannina, Greece, July 2017.
Detection of Predictable Temporal Changes in Multidimensional Biological Sequences.
Advisor: Aristidis Likas, Professor.

This work investigates the predictability of interesting temporal changes between the various states of a longitudinal microbiome dataset, whilst those changes occur at time-points subsequent to the analyzed ones. Predictability has been defined as the generalization performance of an optimal classification system built using a given dataset and tested with a given measure. The temporal dataset used was a longitudinal microbiome dataset containing information about the evolution of the relative abundances of the vaginal microbiome of a number of women. Initially, the analysis focused on the prediction of double changes in microbial composition (named as spikes), given the population relative abundances in previous time instances. The constructed datasets were classified using several methods with accuracy about 70% for the prediction of spikes.

Next we searched for subsets of the datasets being more predictable than the complete dataset. A continuous measure describing the amount of temporal change between consecutive time-points, named spikeness was estimated for all time-points. The dataset examples were ranked based on spikeness and data subsets were created containing top-ranked positive and bottom-ranked negative examples. The classification system used for measuring predictability (called black-box classifier), consisted of a set of various classification models as well as external model parameters and the output classification result for each data subset was obtained from the best performing model.

Based on the above ideas, a new automatic way of detecting predictable temporal changes has been proposed. An approach called rank-based predictability was applied for estimating the predictability of gradually increasing subsets of the dataset, which were selected based on the ranking of the examples. The methodology is based on first transforming the time series into symbolic ones using clustering techniques and then defining patterns of temporal change using a symbolic representation. Then a two-class dataset was constructed given a pattern of temporal changes and its prediction features. As a second step, the rank-based predictability approach was applied to this dataset, as a way of estimating the predictability of temporal patterns. Patterns of temporal changes with predictability greater than a user-specified threshold were considered predictable. The experimental results using four temporal patterns indicated that all temporal patterns were predictable for subsets having a high coverage of the positive examples. Moreover, the results indicated that the predictability of the rank-based subsets was always greater than the average predictability of randomly selected subsets.

# Εκτεταμενη Περιληψη

Νέστωρ Τιμονίδης, Μ.Δ.Ε. στην Πληροφορική, Τμήμα Μηχανικών Η/Υ και Πληροφορικής, Πανεπιστήμιο Ιωαννίνων, Ιούλιος 2017.
Εντοπισμός Προβλέψιμων Χρονικών Μεταβολών σε Πολυδιάστατες Βιολογικές Ακολουθίες.
Επιβλέπων: Αριστείδης Λύκας, Καθηγητής.

Στην εργασία αυτή μελετάται η δυνατότητα πρόβλεψης μεταβάσεων μεταξύ των διαφόρων καταστάσεων ενός χρονικά εξελισσόμενου μικροβιωματικού συνόλου δεδομένων. Ως προβλεψιμότητα ενός συνόλου δεδομένων ταξινόμησης ορίζεται η γενικευτική ικανότητα ενός βέλτιστου συστήματος ταξινόμησης, κατασκευασμένου με την χρήση του συνόλου δεδομένων και αξιολογούμενου με μία καθορισμένη μετρική. Στην εργασία αξιοποιήθηκε ένα μικροβιωματικό σύνολο δεδομένων, το οποίο περιείχε πληροφορίες για την εξέλιξη του κολπικού μικροβιώματος ενός πλήθους γυναικών. Η ανάλυση σε αρχική φάση εστίασε στην πρόβλεψη διπλών μεταβολών στην μικροβιωματική σύνθεση (ορισμένες ως spikes), με δεδομένες τις σχετικές αφθονίες των πληθυσμών σε προγενέστερες χρονικές στιγμές. Τα σύνολα δεδομένων που κατασκευάστηκαν, ταξινομήθηκαν με τη χρήση πολλαπλών μεθόδων ταξινόμησης με περίπου 70% ακρίβεια στην πρόβλεψη των spikes.

Στην συνέχεια ασχοληθήκαμε με τον εντοπισμό υποσυνόλων ενός συνόλου δεδομένων, τα οποία ήταν πιο προβλέψιμα από το αρχικό σύνολο δεδομένων. Ορίστηκε μια συνεχής ποσότητα που ονομάστηκε spikeness, η οποία περιγράφει το μέγεθος των χρονικών μεταβολών μεταξύ των διαδοχικών χρονικών στιγμών. Τα παραδείγματα των συνόλων δεδομένων κατατάχθηκαν με βάση το spikeness και δημιουργήθηκαν υποσύνολα δεδομένων τα οποία περιείχαν κορυφαίας-κατάταξης θετικά και τελευταίας-κατάταξης αρνητικά παραδείγματα. Με τον τρόπο αυτό ορίστηκαν υποσύνολα με ανώτερη προβλεψιμότητα σε σχέση με το αρχικό σύνολο. Το σύστημα

ταξινόμησης που αξιοποιήθηκε για την μέτρηση της προβλεψιμότητας (ονομάστηκε black-box classifier), απαρτίζονταν από ένα σύνολο διαφόρων μοντέλων ταξινόμησης καθώς και εξωτερικών παραμέτρων για τα μοντέλα, ενώ η έξοδος-αποτέλεσμα της ταξινόμησης για κάθε σύνολο δεδομένων λαμβάνονταν από το μοντέλο με την καλύτερη επίδοση.

Με βάση τις παραπάνω ιδέες προτάθηκε μια νέα μέθοδος αυτόματου εντοπισμού προβλέψιμων χρονικών μεταβολών. Ορίστηκε καταρχήν μια προσέγγιση με το όνομα rank-based predictability για τον υπολογισμό της προβλεψιμότητας διαδοχικά αυξανόμενων υποσυνόλων ενός συνόλου δεδομένων ταξινόμησης, τα οποία επιλέχθηκαν με βάση την κατάταξη των παραδειγμάτων. Η προτεινόμενη γενική μεθοδολογία βασίζεται καταρχήν στην διακριτοποίηση των χρονοσειρών σε συμβολικές με την χρήση τεχνικών ομαδοποίησης και έπειτα στον καθορισμό μοτίβων χρονικών μεταβολών με την χρήση μιας συμβολικής αναπαράστασης. Στην συνέχεια, κατασκευάστηκε ένα σύνολο δεδομένων δύο κατηγοριών δοθέντος ενός μοτίβου χρονικών μεταβολών και των χαρακτηριστικών για την πρόβλεψη. Σαν δεύτερο βήμα, η προσέγγιση rank-based predictability εφαρμόστηκε στο σύνολο δεδομένων ούτως ώστε να υπολογίσει την προβλεψιμότητα των χρονικών μοτίβων. Μοτίβα χρονικών μεταβολών με προβλεψιμότητα μεγαλύτερη από ένα κατώφλι καθορισμένο από τον χρήστη, θεωρήθηκαν ως προβλέψιμα. Το πειραματικά αποτελέσματα με την χρήση τεσσάρων χρονικών μοτίβων υπέδειξαν πως όλα τα χρονικά μοτίβα ήταν προβλέψιμα για υποσύνολα με υψηλή κάλυψη των θετικών παραδειγμάτων. Επιπλέον, τα αποτελέσματα υπέδειξαν πως η προβλεψιμότητα των βασιζόμενων σε κατάταξη υποσυνόλων ήταν πάντοτε μεγαλύτερη από την μέση προβλεψιμότητα τυχαία επιλεγμένων υποσυνόλων.

# Chapter 1

# Introduction



## 1.1 Longitudinal Microbiome Data

The human body harbors microorganisms that inhabit surfaces and cavities exposed or connected to the external environment [2]. These microorganisms form complex ecological communities that exist in each body site [3], have mutual relationships with the host [2] and outnumber our own cells by at least a factor of 10 [3]. The aggregate of these microorganisms (also referred to as microbiomes) in the human organism is the human microbiota [4].
Little is known about the temporal dynamics of the microbiome communities. The temporal dynamics are the changes of the microbiome communities through time. A microbiota can change in terms of diversity and dominant taxa.

**Definition 1.1.** Taxon definition: populations of microorganisms like bacteria that form a unit.

The taxa composing the human microbiome (bacteria, archaea and viruses) can be measured by their relative abundances. The relative abundance of a taxon is a value indicating its dominance at the given microbiome. The dominance is measured by

the percentage that the population of the microorganism has over the complete microbiome population.

Temporal variation refers to the difference in the microbiota across time. Interpersonal variation refers to the difference in the microbiota across persons. A longitudinal analysis is based on sampling microbiome data from different time-points and different persons. Therefore, temporal and interpersonal variation are both included in the longitudinal analysis. The inclusion of the two variations can provide insight in the temporal dynamics of the given microbial community [5]. This is because the microbial community of a single subject can be analyzed in different time-points and thus temporal changes can be tracked [6]. At the same time it can be compared with the microbial community of different persons which can provide insight in the dynamics of such community [5]. Downside is that more samples are generated per subject, so fewer subjects are considered in a longitudinal study.

Data were gathered from Gajer's et al. work in [1]. The research goal in [1] was the description of the temporal dynamics in human vaginal microbiota. Specifically, the vaginal bacterial communities of 32 women were analyzed over a period of 16 weeks in terms of their composition. Five major classes characterizing the bacterial communities were identified, and the analysis focused on the temporal rate of change in the communities. There was an identification of both stable and instable communities and the community stability was modeled using a log-linear mixed-effects model. The results indicated that the variation found in microbial composition and high diversity levels were not indicators of dysbiosis. Dysbiosis, as a term is used to describe the imbalance of a microbial community inside or outside a host [7].

The dataset in [1] was derived from self-collected mid vaginal swabs of 32 women, while this procedure occurred twice weekly for 16 weeks using a validated self-collection protocol. A data set of 2,522,080 high-quality classifiable 16S rRNA gene sequences was obtained from 937 samples with an average of $2692 \pm 910$ sequences per sample. The relative abundances of 330 bacterial taxa at family, genus or species levels that compose the vaginal microbiome of the subjects were estimated from the rRNA sequences using the Ribosomal Database Project (RDP) Naive Bayes Classifier [8]. Therefore this table contained information about the 330 relative bacterial abundances of 32 women for approximately 30 time-points per subject.

Therefore, this dataset contained 937 examples and 330 features where:

1. the examples correspond to the microbiome of various subjects at various time-

points per subject,

2. the features correspond to the relative abundances of specific bacteria composing the microbiome,

3. there exists a temporal relationship between examples belonging to the same subject, meaning that they correspond to different time-points of the subject's time-series. Therefore, one could estimate the temporal distance between two examples of the same subject, wherein two consecutive time-point have a distance of 4 days on average. However this approximation is not possible between examples of different subjects, because the microbiome sampling was asynchronous for all subjects.

Besides the temporal relative abundances, additional information about the samples (termed metadata) was also available:

1. The time in study that the sample was taken from.

2. The id of the subject.

3. Race.

4. Age.

5. Total Read Counts.

6. Community State Type. This state type is based on differences in species composition and their relative abundances according to [1]. The community state types were 5 in total and labeled I,II,III,IV-A and IV-B. Each community state type was dominated by a bacteria species. For example, community state type III was dominated by *L. iners*, while community state type I was dominated by *L.crispatus* [1].

The supplementary material of [1] also contained information on how the community state types were estimated. Specifically, a dissimilarity matrix between the microbiome of all time-points of all subjects of the dataset (937 microbiome instances in total) was constructed with the use of the Jensen-Shannon divergence measure (section 2.4). Afterwards, the agglomerative hierarchical clustering (section 2.3) was applied to the proximity matrix. The silhouette measure (section 2.3) was used to measure the

clustering quality for various numbers of clusters. The highest mean silhouette value found was for five clusters. Hence, these five clusters constituted the five community state types which were used to characterize the state of each microbiome at its corresponding time-point.

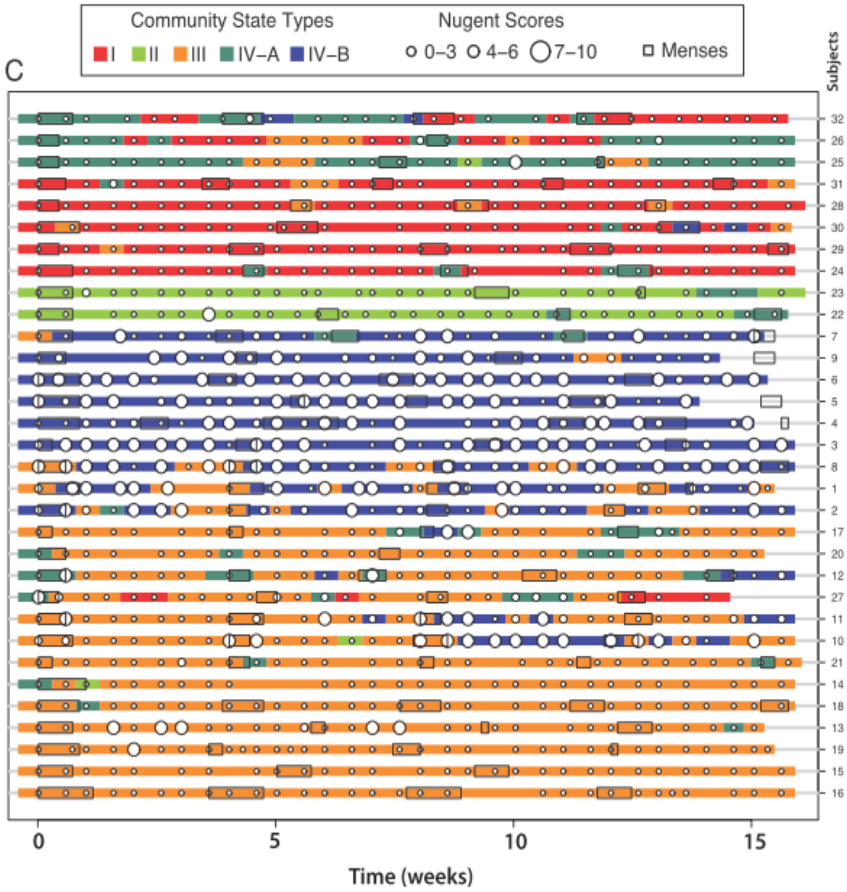


Figure 1.1: Profiles of Community State Types, menses and Nugent Scores (def. 1.2) for a number of subjects over a period of 16 weeks (Taken from [1]). X-axis: time (weeks). Y-axis: subjects. The colors correspond to the 5 different Community State Types as indicated by the legend above the figure. The circles of each time-series correspond to the time-points, at which microbial samples of the equivalent subject were taken for the analysis in [1]. The size of the circle corresponds to its Nugent Score, while the squares of each time-series correspond to the menses of the subjects.

The time-points were numbered in the order they appeared. Therefore, the time-point corresponding to the first sample was labeled as 1, the time-point of the second sample as 2 and so on (fig. 1.1). The supplementary material also contained informa-

tion about the menstrual period of the subjects. More specifically, it contained visual plots for each time-series in which the time-points corresponding with the menstrual period were represented by a square symbol. Moreover, the analysis contained information about the Nugent Score of the subjects for each time-point. The Nugent Score was defined by Nugent et al. in [9], as:

**Definition 1.2.** Nugent Score: score which measures the bacterial vaginosis of a subject.

## 1.2 Thesis Contribution

The goal of this research is the investigation of the predictability of interesting changes between the various states of longitudinal data, whilst those changes occur at time-points subsequent to the analyzed ones.
This thesis aims in not only detecting temporal changes between the states of a time-series, but as a step furthermore on detecting which of these changes are predictable. Predictability is associated with the generalization performance of an optimal classification system. Moreover, a change can be considered predictable if its predictability is greater than a specified threshold value. This goal is based on the hypothesis that there exist temporal changes of discrete states characterizing time-series/longitudinal data, which can be predicted from data available at time-points preceding the changes.
Chapter 2 focuses on the explanation of the basic machine learning methods as well as the dissimilarity measures, used for classification, clustering and evaluation in this research. In chapter 3, the analysis focuses on the prediction of double changes in microbial composition, which have been named as spikes.
In chapter 4, the existence of subsets with predictability greater than the complete dataset is being investigated. The predictability measure is defined as the generalization performance of an optimal classification system built using a given dataset and tested with a given measure. Two different approaches are being compared for detecting subsets with predictability greater than the complete dataset: rank-based and random-based predictability.
Chapter 5 focuses on the detection of predictable temporal changes in the composition of the longitudinal microbiome dataset. The time-series are discretized into symbolic ones with the use of clustering techniques. Afterwards, patterns of temporal changes are being detected with the use of a symbolic representation. Finally, the rank-based predictability approach is applied for estimating the predictability of the

temporal patterns.

Chapter 6 contains the experimental results of chapter 5. Each section of chapter 6 corresponds to the equivalent section of chapter 5, and provides a representation of the results that are derived from the detection of predictable temporal changes in the complete dataset. The chapter closes with the discussion of the experimental results.

The final chapter, chapter 7, has two sections. In the first section, the conclusion of this research is being presented. The second and final section of this thesis contains the future work references which can expand this research into new directions.

# Chapter 2

# Machine Learning Methods



## 2.1 Introduction

This chapter focuses on the presentation of various machine learning methods and distance measures that have been used throughout the thesis. The sections bellow have been structured based on the category of the presented methods, which are: classification methods, clustering methods and distance measures.

## 2.2 Classification Methods

### 2.2.1 Classifier Evaluation

The Cross-Validation method (also referred to as CV method) is a method used for estimating generalization in classification problems. Generalization can be defined as the ability of a supervised learning model (also referred to as classifier) to be able to classify correctly examples that were not used in the training process [10].

The CV method utilizes the accuracy measure for estimating the generalization of a classifier. The accuracy of a classifier is defined as the probability that a randomly selected example of the dataset is classified correctly by the model [11]. Therefore:

$$accuracy = Pr(C(x) == y), \tag{2.1}$$

where C is the classifier and x,y $\in$ X, are the randomly selected example from dataset X and its label. Given the classification of a number of examples, which are labeled as positive or negative, an alternate definition of accuracy is:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}, \tag{2.2}$$

where:

1. TP = the number of positive examples classified correctly as positives (true positives).
2. TN = the number of negative examples classified correctly as negatives (true negatives).
3. FP = the number of negative examples classified incorrectly as positives (false positives).
4. FN = the number of positive examples classified incorrectly as negatives (false negatives).

In cases of imbalanced two-class datasets, where the number of examples belonging to the one category is greater than the other, the f-measure is being used instead of accuracy:

$$f - measure = 2\frac{precision * recall}{precision + recall} \tag{2.3}$$

In the CV method, the dataset is partitioned into K-disjoint subsets (also referred to as folds) with approximately equal size. Let D be considered the dataset and $D_1, D_2, .., D_K$ its disjoint subsets [11].
For i=1,..K:

1. $D_i$ is used as the testing set and $D \backslash D_i$ is used as the training set.
2. $D \backslash D_i$ constructs a classifier using any classification algorithm.
3. $D_i$ is tested to the classifier.

4. $accuracy_i$ is estimated as the classification accuracy of the classifier for $D_i$.

After the procedure has been completed for all K-folds, then the total Cross-Validation accuracy is estimated as:

$$accuracy = \frac{\sum_{i=1}^{K} accuracy_i}{K}, \tag{2.4}$$

which is the average for the accuracy of all folds [11].
The aforementioned procedure is also called as K-fold Cross-Validation. In case that the K = N, where N is the number of examples, then the approach is called leave-one-out Cross-Validation [11]. In leave-one-out (LOT) CV, each example $x \in X$ is the testing set, and for each x, the rest of the examples $X \backslash x$ compose the training set.
In this thesis, the leave-one-out Cross-Validation approach was used due to the small size of the datasets being used.

### 2.2.2 Support Vector Machines

The support vector machines (also referred to as SVM) are a category of supervised learning models used for regression and classification. The original SVM algorithm was invented by Vladimir N. Vapnik and Alexey Ya. Chervonenkis in 1963, but in its present form was published by Vapnik and Corrina Cortes in 1995 [12]. In classification analysis, support vector machines are used for two-class classification problems and can be considered to be derived by linear models of the form:

$$y(x) = w^T \phi(x) + b, x \in R^n. \tag{2.5}$$

$\phi(x)$ refers to a feature map from input space x to a linearly separable space. Therefore SVM can be extended to non-linear classification problems [10].
Linear models such as the aforementioned can present multiple solutions/decision boundaries for class separation, (show proof of multiple local minima by the linear classifiers such as perceptron). SVM on the other hand minimize the generalization error by maximizing the margin dictated by the smallest distance between the decision boundary and the examples. For that reason, the decision boundary is selected based on the margin maximization [10].

**Implementation steps**

In SVM, The maximum margin solution is dictated by the Lagrangian function:

$$L(w,b,a) = \frac{1}{2}|w|^2 + C\sum_{n=1}^{N}\xi_n - \sum_{n=1}^{N}a_n\{t_n y(x_n) - 1 + \xi_n\} - \sum_{n=1}^{N}\mu_n\xi_n, \tag{2.6}$$

where:

1. $a_n \geq 0$ and $\mu_n \geq 0$ are the Lagrange multipliers.
2. $x = (x_1, x_2, ..., x_N)$ is the input space with N features and
3. $w = (w_1, w_2, ..., w_N)$ is a vector of real-valued weights assigned to each of the N features.
4. $\phi(x)$ is the feature mapping of x to a linearly separable space
5. b is the bias.
6. $y(x_n) = w^T\phi(x) + b$ (2.1), is the linear function defining the decision boundary between two defined data classes. Therefore $y(x_n) = 0$ for data points that are on the decision boundary, $y(x_n) > 0$ for data points that belong to the positive class and $y(x_n) < 0$ for data points that belong to the negative class.
7. $t_n \in \{-1, 1\}$ corresponding to the sign of $y(x_n)$, such that $t_n(w^T\phi(x) + b) > 0$ for correct classification of example $x_n$.
8. $\xi_n$ is defined slack variable where:

$$\xi_n = |t_n - y(x_n)|, \quad n = 1, 2.., N. \tag{2.7}$$

   Given the definition: $\xi_n = 0$ for data points that are on the side of the correct margin boundary, $\xi_n = 1$ for data points that are on the decision boundary and $\xi_n > 1$ for misclassified data points.

9. The parameter $C > 0$ exists as a constraint that creates a balance between the minimization of training error and the control of overfitting. The higher the C value, the smaller the margin and therefore the more prone to overfitting the model is. The lower the C value, the higher the margin due to the slack variables and therefore the more prone to training errors the model is [10].

For this constraint optimization problem, the *Karush-Kuhn-Tucker* (also referred to as KKT) conditions must be satisfied:

$$a_n \geq 0, \tag{2.8}$$

$$t_n y(x_n) - 1 + \xi_n \geq 0, \tag{2.9}$$

$$a_n\{t_n y(x_n) - 1 + \xi_n\} = 0, \tag{2.10}$$

$$\mu_n \geq 0, \tag{2.11}$$

$$\xi_n \geq 0, \tag{2.12}$$

$$\mu_n * \xi_n = 0, \tag{2.13}$$

where n = 1,...,N [10].
The problem with the above Lagrangian function is that it has 3 parameters: w,a,b. However by considering the function as a primal one and solving the Lagrangian dual of it, a more simplified problem is obtained:

$$\tilde{L}(a) = \sum_{n=1}^{N} a_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m k(x_n, x_m), \tag{2.14}$$

where $k(x_n, x_m)$ is a kernel function of two data points $x_n$ and $x_m$, with:

$$k(x_n, x_m) = \phi(x_n) * \phi(x_m), n = 1, .., N. \tag{2.15}$$

The two constraints that have been derived from the dual function are:

$$box \quad constraints: \quad 0 \leq a_n \leq C, \quad n = 1, ..., N, \tag{2.16}$$

$$\sum_{n=1}^{N} a_n t_n = 0, \quad n = 1, ..., N. \tag{2.17}$$

Several kernel functions can be used with SVM. The most frequently used are:

1. RBF or Gaussian kernel:

$$k(x_n, x_m) = exp(-\frac{||x_n - x_m||^2}{2\sigma^2}) \tag{2.18}$$

2. Linear kernel:

$$k(x_n, x_m) = x_n^T x_m \tag{2.19}$$

3. Polynomial kernel:

$$k(x_n, x_m) = (x_n^T x_m + c)^d, \tag{2.20}$$

where $c \geq 0$ is a free parameter and d is the polynomial degree.

4. Cosine kernel:

$$k(x_n, x_m) = \frac{x_n^T x_m}{||x_n||_2 ||x_m||_2} \tag{2.21}$$

### 2.2.3 Decision Trees

The decision tree classifier is a category of classification models which are based on decision trees. A decision tree is an acyclic graph in which, given a set of features and a set of class labels, the nodes represent decision splits on features, based on specific criteria, the branches represent possible decision outcomes and the leaf nodes represent the class labels or decisions. The act of building a decision tree is referred to as decision tree induction [13].
The induction of decision trees is a non-parametric approach for the construction of classification models. Therefore, there are no requirements regarding the probabilistic prior distributions of the classes, and the dataset is sufficient for the decision tree construction [13].

#### Decision Tree Induction

A decision tree is defined by three types of nodes: the root node, with only outgoing edges, the internal nodes, with one incoming and two or more outgoing edges, and the leaf nodes with only incoming edges. Given a set of features, there is an exponential number of decision trees that can be constructed [13]. Therefore, there has been developed a number of efficient algorithms with the purpose of constructing an suboptimal tree in a reasonable amount of time [13]. Some of the most efficient ones are: Hunt's algorithm, ID3, C4.5, SLIQ,SPRINT and CART.
Hunt's algorithm is one of the earliest ones and served as a basis of many subsequent decision tree algorithms [13]. Therefore, it's general structure can be an asset for the explanation of decision tree induction.

**Attribute test condition and split measures**

The way that a decision tree induction algorithm applies an attribute test condition depends on the type of features. The possible feature types are four: binary, nominal, ordinal and continuous [13]. In case of binary feature types, meaning that there are two possible outcomes for each feature, a binary split is performed on each node. In case of nominal feature types, the split is either binary or multi-way, meaning that each node will have more than two children depending on the number of possible outcomes. In case that a binary split is selected and there are more than two possible outcomes, then the feature values will be merged in two groups for producing two outcomes. Ordinal features can have binary or multi-way splits as long as there is no violation of the order property of the feature values. Finally in continuous features, a common approach is that all feature values are discretized into intervals which represent the splits, and each new feature value is assigned to one of the possible intervals [13].

Node impurity is measured by the distribution of classes at the children nodes after a split has occurred. The more homogeneous the class distribution is at the children nodes, the more impure the split is. If we consider a binary split at which the class distribution of the first child node is (0,1) while the distribution of the second one is (0.5,0.5), then the node impurity of the first one is zero, while the impurity of the second one is the highest possible [13]. Node impurity serves as a basis for a number of measures employed for finding the best split, with some of them being:

1. $$Gini(t) = 1 - \sum_{i=0}^{c-1}[p(i\backslash t)]^2, \tag{2.22}$$

2. $$Entropy(t) = -\sum_{i=0}^{c-1} p(i\backslash t) log_2 p(i\backslash t), \tag{2.23}$$

3. $$Classification\_error(t) = 1 - \underset{i}{max}[p(i\backslash t)], \tag{2.24}$$

where c is the number of classes and $p(i\backslash t)$ the probability distribution of class i at node t [13].

**Advantages and Disadvantages**

Decision tree induction algorithms have in worst case a time complexity of O(w), with w being the maximum tree depth. Therefore, the decision tree algorithm is fast with even large training datasets. Furthermore, the decision trees models have high interpretability, meaning that a comprehension of the model is easy, especially in small trees [13].

Moreover, while the algorithms are sensitive to noise, pruning techniques have been employed for dealing with overfitting. Pruning techniques reduce the number of tree branches, either by stopping their development during the training phase (pre-pruning) or trimming them after the training phase (pre-pruning) In pruning, a data subset known as validation set is used for measuring the generalization error as a way of validating the technique [13].

As far as disadvantages of the decision tree induction algorithms are concerned, an important one is that they are prone to overfitting on non-useful features, which limits the generalization ability of the model to new data. For that reason, feature selection techniques are useful in selecting the most important features for the training process and increasing the classification accuracy. Furthermore, decision tree classifiers do not have a good generalization to specific boolean problems like the parity function. The parity function is a function whose value is 1 or 0 depending on whether the the boolean features with the value TRUE have an even or odd number.

### 2.2.4 k-Nearest Neighbors

The k-Nearest Neighbors algorithm (also referred to as k-NN) is a non-parametric approach used for classification and regression problems. The k-NN algorithm is based on the nearest neighbors (NN) problem: determination of a data point which is nearest to a given query point. This problem is applied in Geographical Information Systems with the association of data points to geographical locations [14].

In a nearest neighbors classifier, the query point is the testing data point, while the data point for determination belongs to the training dataset. Therefore, the label of a testing data point x* is determined by the label of the training data point nearest to it.

In a k-NN classifier with $k > 1$, the problem is extended to the determination of the k training data points nearest to the testing data point x*, thus considered as k-nearest

neighbors. The class label of x* is determined by the majority vote of the k-nearest neighbors: the class with the majority of the k- nearest neighbors belonging to it will have its label assigned to x* [15].
The NN problem has been considered a special case of Parzen windows problem, where the number of points $k_N = k$ is considered fixed, and the volume around the points x is variable in order to include k points [16].
There are various distance measures which can be applied in the k-NN algorithm. The Euclidean, Mahalanobis, Chebychev and Minkowski distance measures have been used for continuous data, as well as the hamming distance for discrete data [16]. Spearman's rank and Pearson correlation coefficients have also been applied in k-NN at Bioinformatics problems such as gene expression microarray data classification [17]. Finally, Dynamic Time Warping as a distance measure together with the 1-NN classifier has been reported to achieve good results on time-series classification analysis [15]. In terms of selecting the number of neighbors k to be determined, k is usually preferred to be an odd number in order to avoid ties in the majority vote. However, in case that a tie occurs, a usual approach is the selection of the class label belonging to the 1-nearest neighbor.
As far as advantages are concerned, k-NN is very simple as a non-parametric algorithm. There is a lack of need for constructing a classification model with training examples, due to the fact that the testing data points are classified based on their proximity with the training data points [13]. Furthermore, the simplicity of k-NN makes it useful for complex datasets. Finally there is no information loss during classification, which makes k-NN highly interpretable on the results [13].

### 2.2.5 Linear Discriminant Classifier

The linear discriminant algorithm was introduced by Fisher in 1936 in [18], as a linear classification model with dimensionality reduction. Given a n-dimensional dataset x and 2 classes $C_1$ and $C_2$, the idea is the projection of x in one dimension using the linear function: $y = w^T x$. Afterwards, a testing data point x* will be classified in $C_1$ if $y(x*) \geq -w_0$, or otherwise in $C_2$, with $w_0$ being a threshold value [10].
The projection of a n-dimensional dataset into one dimension can lead to loss of information and therefore into an overlap of the two classes. For that reason, the linear discriminant algorithm tries to find the optimal values of the weight component w,

such that the projection into one dimension minimizes the variance within classes and maximizes the variance between classes [10].
The Fisher criterion is defined as the ratio of between class variance to within class variance [10]. Therefore, the optimal weight component w is given by maximizing the ratio.

$$J(w) = \frac{w^T S_B w}{w^T S_W w} = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2}[10], \tag{2.25}$$

where:

1. $$S_B = (m_2 - m_1)(m_2 - m_1) \tag{2.26}$$
   is the between-class covariance matrix.

2. $$S_W = \sum_{n \in C_1} (x_n - m_1)(x_n - m_1)^T + \sum_{n \in C_2} (x_n - m_2)(x_n - m_2)^T \tag{2.27}$$
   is the within-class covariance matrix.

3. $$m_1 = \sum_{n \in C_1} \frac{x_n}{N_1}, \tag{2.28}$$
   is the mean vector of class 1.

4. $$m_2 = \sum_{n \in C_2} \frac{x_n}{N_2}, \tag{2.29}$$
   is the mean vector of class 2.

5. $$s_1^2 = \sum_{n \in C_1} (y_n - m_1)^2, \tag{2.30}$$
   is the within-class variance of class 1 after the projection to one dimension has occurred.

6. $$s_2^2 = \sum_{n \in C_2} (y_n - m_2)^2, \tag{2.31}$$
   is the within-class variance of class 2 after the projection to one dimension has occurred.

7.

$$y_n = w^T x_n \tag{2.32}$$

is the linear function.

By differentiating J(w) with respect to w and solving the system, the following equation is provided:

$$w \propto S_W^{-1}(m_2 - m_1), \tag{2.33}$$

meaning that w is proportional to the difference of the class means and the inverse of the within-class covariance matrix [10].
The algorithm is further known as Fisher's linear discriminant, due to the fact that the projected data construct a discriminant by selecting a threshold value $w_0$ and classifying data points into class 1 or class 2 based on whether $y(x) \geq -w_0$ or not [10].

### 2.2.6 Stacked Autoencoders

#### Introduction

An autoencoder is a representation-learning artificial neural network, used for unsupervised learning. Initially it was proposed by Hinton and Salakhutdinov in [19], as a dimensionality reduction method for high dimensional data with the use of a mutlilayer neural network.
Representation learning is the application of machine learning for discovering the proper representation of data for increasing the performance of machine learning algorithms [20]. Therefore, representation learning aims in finding the most important features and an optimal mapping to a feature space which increases performance.
Deep neural networks deal with representation leaning, with the introduction of multiple hidden layers. Multiple hidden layers allow for representations of a dataset to be expressed by simpler ones. The representations become increasingly more abstract as the hidden layers are closer to the output one [20].

#### Autoencoder structure

The autoencoder is a neural network composed of one input, one hidden and one output layer. The hidden layer, also referred to as h, acts as the representation of

the input [20]. The basic characteristic of the autoencoder that separates it from the other neural networks, is that it seeks in copying its input to the output.
For that reason, the network is consisted of the encoder h = f(x), and the decoder r = g(h) functions. For a perfect copy of the input to the output layer to be possible, g(f(x)) = x. However, due to restrictions placed in autoencoders, g(f(x)) $\neq$ x, which means that a perfect copy is not possible. Since the copy is achieved approximately, the model prioritizes its inputs and selects the most important ones for copy. This procedure leads to the model learning useful information about the data [20].
The encoder function of the autoencoder is given by the equation:

$$h = f_{w,b}(x) = f(w^T x + b), \tag{2.34}$$

The decoder function of the autoencoder is given by the equation:

$$r = g_{v,v_0}(h) = g(v^T h + v_0) \tag{2.35}$$

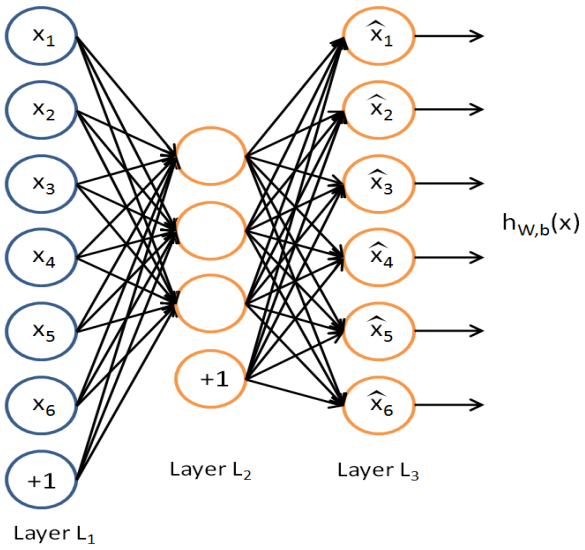


Figure 2.1: Schematic representation of an autoencoder. The three network layers (input - Layer $L_1$, hidden - Layer $L_2$, output - Layer $L_3$), are shown from left to right. Layer $L_2$ acts as the representation of data. The lines connecting $L_1$ to $L_2$ represent the weights w, which together with the input values and the bias $b$ compose the encoder function (eq. 2.34), with f(x) being an activation function. The equivalent lines from $L_2$ to $L_3$ represent the hidden layer weights v, which together with the bias $v_0$ and the encoder values h, compose the decoder function (eq. 2.35), with g(x) being an activation function.

### Sparse Regularized Autoencoders

The minimization of the following loss function, describes the autoencoder's learning procedure:

$$L(g(f(x)), x) = ||g(f(x)) - x||_2^2 \tag{2.36}$$

is the mean squared error between g and x, and n the input layer size.
While mean squared error is not the only loss function used, it is a frequently used one [20]. Regularized autoencoders scale the encoder and decoder capacity based on the complexity of the input distribution [20]. This scaling includes additional model properties, like sparsity and robustness to noise or missing values [20].
An approach for autoencoder regularization is through the addition of the sparsity penalty to the loss function:

$$L(x, g(f(x))) + \Omega(h), \tag{2.37}$$

where

$$\Omega(h) = \beta \sum_i |h_i| \tag{2.38}$$

is the sparsity penalty and $\beta$ is the sparsity parameter.
Sparse autoencoders result in a large proportion of the input neurons having zero output value for any given input. Therefore, as the sparsity constraint gets higher, the network gets more selective in choosing an important proportion of the data input and discarding the rest. Further approaches in regularized autoencoders include the addition of the $L_2$ regularization to the loss function:

$$L(x, g(f(x))) + \beta \sum_i |h_i| + \lambda w^T w, \tag{2.39}$$

where $w^T w$ is the $L_2$ regularization, and $\lambda$ is the regularization parameter.

### Stacked Autoencoders

A frequently used deep neural network model for classification is the stacked autoencoder [21]. In stacked autoencoders, many sparse autoencoders are trained in succession, with the characteristic the the hidden layer size gets smaller as the hidden layers are closer to the output layer [21].
As a next step, the softmax activation function is used for training a supervised neural

network between the final hidden layer, acting as the data representation, and the output layer having a softmax function:

$$y_j = \frac{e^{z_j}}{\sum_{k=1}^{C} e^{z_n}}, \quad where \quad n = 1, .., C \tag{2.40}$$

with z being an C-dimensional vector and C being the number of classes. The softmax activation function converts a C-dimensional vector with real values, to a C-dimensional probability vector [10].

As a last step, with the weights optimized from the consecutive sparse autoencoders and the training between the final hidden layer and the output one, the complete neural network is now trained for a further optimization of the weights [21].

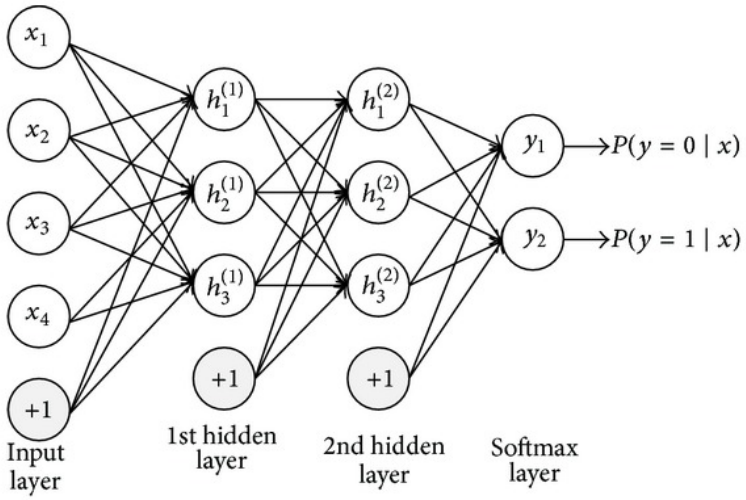


Figure 2.2: Schematic representation of a stacked autoencoder. The first layer (Input layer) is the input one. The next two hidden layers ($1^{st}$ hidden layer and $2^{nd}$ hidden layer) are the two sparse autoencoders stacked in succession. The final layer (Softmax layer) is the output one, which together with the $2^{nd}$ hidden layer compose a two-layered neural network trained with softmax activation function. The +1 neuron shown in all layers corresponds to the bias term b.

## 2.3 Clustering Methods

### 2.3.1 K-means

K-means is considered to be one of the most important clustering algorithms. On general terms, K-means is a prototype-based, partitional clustering algorithm with

the goal of finding a specified number of clusters, which are represented by their centroids. In K-means, the centroid is the mean of a group of data belonging to the same cluster [13].
K-means is usually applied to data belonging to a continuous n-dimensional space. Furthermore, as in all prototype-based clustering algorithms, all data are considered to belong to the same level of hierarchy [13].

**Implementation steps**

The K-means clustering technique is simple and can be broken down to 5 basic steps:

1. K data points from the n-dimensional space are selected randomly as initial centroids. K is a user defined parameter and refers to the number of desired clusters.
2. Each data point is assigned to its closest centroid in terms of distance. All points assigned to a centroid compose a cluster and thus all K clusters are formed.
3. In each cluster, the mean of the data points assigned to it is estimated and constitutes to the updated centroid of the cluster.
4. Step 2 is repeated.
5. The above procedure is continued until there is no change in the centroids [13].

**Distance Measure and Objective function**

In K-means, the notion of closest is being quantified with the use of a distance measure. The distance measure varies with the different types of data being applied to the algorithm. The most frequently used distance measures are the euclidean distance, for data points in euclidean space, and the cosine similarity for document data. However, Manhattan distance is also used for euclidean data, while the Jaccard measure is used on document data [13].
The purpose of the objective function in K-means is the measurement of clustering quality. The objective function depends on the distance measure between the points and the centroids and thus is dependent on the different types of data [13]. The two

main data categories in K-means are data in Euclidean space and document data. For data in Euclidean space, the objective function of data belonging to the Euclidean space, and having the Euclidean as a distance measure, is the Sum of Squared Error (also referred to as SSE). SSE is defined as the total sum of the Euclidean distances between all points and their corresponding centroids.

$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist(c_i, x)^2, \tag{2.41}$$

with dist being the standard Euclidean distance between two points in Euclidean space:

$$dist(c_i, x) = ||c_i - x||_2. \tag{2.42}$$

$c_i$ is the centroid of the $i^{th}$ cluster defined as:

$$c_i = \frac{1}{m_i} \sum_{x \in C_i} x. \tag{2.43}$$

The main advantage of K-means is its computational simplicity. Its time complexity is: O(NKq), where q is the number of iterations necessary for convergence, N is the number of data and K is the number of clusters. Due to the fact that q and K are fairly smaller than N, K-means is preferred for big data clustering.
Its main disadvantage however is that if we consider SSE as its objective function, SSE is influenced to a great extent by the random initialization of the clusters. Different initializations can lead to different cluster results and therefore finding the optimal initial cluster centers is the basic problem of K-means clustering [13].

### 2.3.2 Silhouette Measure

The silhouette measure was introduced by Rousseeuw in 1987 [22], as a way of measuring clustering quality, by comparing the dissimilarities within clusters and between clusters.
Given a clustering solution, the silhouette measure creates silhouette values for all data points [22]. For each data point i, a corresponding silhouette value has been defined as s(i), and given by the following definition:

$$s(i) = \frac{b(i) - a(i)}{max\{a(i), b(i)\}}, \tag{2.44}$$

where

1. a(i) = the average proximity between data point i and all data points belonging to the same cluster A.

2. $$b(i) = \min_{C \neq A} \quad d(i, C), \tag{2.45}$$

   where d(i,C) = the average proximity between data point i and all data points belonging to cluster C, where C is a different cluster from A.

Given the definition above and as far as the range of values of s(i) is concerned, s(i) $\in [-1, +1]$. In cases where a cluster consists only of one data point, then s(i) is set to zero due to the uncertainty of definition of a(i). The value -1 represents the complete misclassification of data point i, while +1 indicates the assignment of i to a very appropriate cluster [22].

The silhouette measure can be used to validate and interpret a clustering approach. For that reason, the average silhouette value for all silhouette values is taken:

$$average(s) = \frac{\sum_{i=1}^{n} s(i)}{n}, \tag{2.46}$$

where n is the total number of data points.

A main advantage of the silhouette measure, is that as a measurement method is not dependent on the clustering approach, but depends on the final partition of the data points. Therefore, it can be used for optimization of the cluster analysis results, by comparing the average silhouette of various clustering approaches applied to the dataset.

A major disadvantage of silhouettes is its instability on extreme cases. In cases where there is only one cluster, b(i) cannot be defined for a silhouette value. Finally, as mentioned before, a(i) is set to zero in cases where a cluster consists of only one data point, due to uncertainty. In cases where each data point constitutes a cluster, then with a(i) = 0, according to the silhouette equation 2.44:

$$s(i) = \frac{b(i) - 0}{max(b(i), 0)} = 1. \tag{2.47}$$

The silhouette value of 1 indicates a well-clustered data partition. However, this judgment is not objective due to the fact that there is a lack of information on whether the clustering quality can be improved, which in most cases of singleton clusters is the case.

### 2.3.3 Agglomerative hierarchical clustering

Agglomerative hierarchical clustering is a category of clustering techniques belonging to the broader category of hierarchical clustering. The agglomerative is the most common of the two basic approaches of hierarchical clustering, with the other one being the divisive hierarchical clustering [13].
Their main difference is that divisive hierarchical clustering starts with a single all-inclusive cluster and splits the clusters based on a given criterion. Agglomerative hierarchical clustering follows the reverse approach and starts with all data points as individual clusters and merges them based on a criterion until all data points belong to a single cluster.
A hierarchical clustering can be displayed graphically with the use of a dendrogram or linkage tree, which is a diagram that contains the order in which the clusters were merged or split, depending on the approach.

**Implementation steps**

1. At first, a proximity matrix is computed. In the beginning where each data point composes a cluster, the proximity matrix is usually either a distance matrix or a similarity matrix (matrix containing information about the similarity of all pairs of data points). On the latter steps, where there exist clusters with more than one data points, there are multiple approaches used and the most common of them will be analyzed in the next subsection.

2. The closest two clusters based on the proximity matrix are being merged.

3. The proximity matrix is being updated based on the proximities of the clusters after merging.

4. Step number 2 is repeated.

5. The termination condition for the above repetition is the existence of a single cluster.

**Proximity between clusters**

There are many approaches regarding the proximity between clusters. The most commonly used are: MIN,MAX, Group Average, Centroid and Ward.

In the MIN clustering technique, proximity between two clusters is being measured by the proximity of the closest two data points that belong to different clusters. In the MAX clustering technique, proximity between two clusters is being measured by farthest two data points that belong to different clusters. In Group Average clustering technique, proximity between two clusters is being defined as the average proximity between all pairs of data points from different clusters [13].
In the centroid method, the proximity between two clusters is defined by the distance between their centroids. Finally, Ward's method measures the proximity between two clusters by measuring the increase of SSE that results from merging the clusters. Therefore, at each iteration it merges the cluster-pair that minimize the SSE. In both Centroid and Ward's method, the clusters are represented by their centroids like in K-means [13].

#### Advantages and Disadvantages

The main advantage of the agglomerative hierarchical clustering techniques is that there is no requirement of the data points themselves and therefore a proximity matrix is sufficient to be given as input. Furthermore, there is no constraint regarding the type of distance or similarity measure that can be used to construct the initial proximity matrix. Finally, they contain information about the hierarchical structure of data which is important for problems like taxonomy creation.
The main weakness of the algorithm is its expense. The overall time complexity of the agglomerative hierarchical clustering algorithm is O($m^2 log(m)$), where m is the number of data points. Therefore, this algorithm is not applied in big datasets. Furthermore, all merges occurring during the algorithm are final, which can provide bad clustering for high dimensional data [13].

### 2.3.4 Dip-dist criterion

The dip-dist criterion tests the empirical density distribution of a given dataset for multi-modality. For a cluster to be assumed as unimodal, its empirical density should have a single mode. A single mode is being defined as a region with a maximum empirical density, while data observations have a decreasing density as they belong further away from the mode. This assumption enables unimodal distributions such

as Gaussian or Student-t to generate clusters which can be identified with that test [23].
The dip-dist criterion, takes as input a distance matrix created from the distances between all data points belonging to the cluster, whose unimodality is to be tested. The typical distance measure used is the euclidean one. The output of the criterion is a dip value, estimated by applying the hartigan's dip test to the dataset, and a p-value of the hypothesis that the cluster's distribution is unimodal [24].

### 2.3.5 Agglodip

Agglodip is an agglomerative clustering algorithm, based on the hartigan's dip-test [24], with the purpose of finding an optimal number of clusters and the creation of clusters which are meaningful to the data. This algorithm is heavily influenced by the dip-means algorithm [23], and tries to achieve the same result with a different approach.
While dip-means gradually divides the clusters based on the unimodality test, until all clusters are considered unimodal, agglodip reverses the approach: initially partitions the data to a large number of initial clusters and iteratively merges pairs of clusters based on the unimodality test, until all clusters are considered unimodal.

**Implementation steps**:

1. Initially the data are being partitioned to a large number of clusters, with the use of a parametric clustering algorithm like hierarchical clustering or k-means. Since the algorithm is agglomerative, the initial number of clusters must be greater or equal than the optimal one. For that reason the initial number of clusters must be as large as possible. Alternatively, the number of initial clusters can be equal to the number of data points, which means that each data point composes a cluster, which is more computationally expensive than the previous option.

2. Each possible pair of clusters ($\frac{K*(K-1)}{2}$, with K being the current number of clusters), is being selected for merge.

3. Each one of the selected pairs is being tested with the dip-dist criterion. From

all possible tested pairs, only one will be merged. For a pair to be merged, the p-value estimated by the dip-dist criterion, of the hypothesis that the cluster is unimodal, must be greater than zero. Furthermore, for all pairs that satisfy that criterion, the one with the lowest dip value will be merged.

4. Step 2 is being repeated.

5. The above procedure is being repeated until there is an iteration where all possible merging pairs have been tested and there is no pair satisfying the aforementioned criterion.

## 2.4 Distance Measures

### 2.4.1 Kullback-Leibler Divergence

The Kullback-Leibler divergence measure or relative entropy, also referred to as KL divergence, was first introduced by Kullback and Leibler in 1951 [25], as a way of measuring the relative entropy of two distributions. In entropy terms, given a distribution p(x) which is used for encoding information of x for transmission, the KL divergence is defined as the additional amount of information required for decoding x, which is encoded by a second distribution q(x) instead of p(x) [10]. On more general terms, the KL divergence measures the divergence of a probability distribution P(x) from a different one Q(x), given by:

$$D_{KL}(P\backslash\backslash Q) = \sum_{x} P(x) ln \frac{P(x)}{Q(x)}, \quad x = x_1, ..., x_n, \quad n \geq 0. \tag{2.48}$$

Given the definition 2.43, KL divergence represents the mean of the log fold change between P and Q with respect to P [25].
Relative entropy is used in computer science, statistics and microbiology [25]. However, it has a number of limitations:

1.

$$D_{KL}(P\backslash\backslash Q) \neq D_{KL}(Q\backslash\backslash P), \tag{2.49}$$

meaning that there is a lack of symmetry in the measure [25].

2. In cases where

$$P(x) \neq 0 \quad and \quad Q(x) = 0, \tag{2.50}$$

$D_{KL}$ is infinite [25].

3. The smaller the P(x) and Q(x) values are, the less reliable the log fold change estimates are [25].

### 2.4.2 Jensen-Shannon Divergence

The Jensen-Shannon divergence is a dissimilarity measure, introduced in [26] with the purpose of overcoming the limitations of KL divergence.
Given two probability distributions P(x) and Q(x), where $x = x_1, ..., x_n$ and $n \geq 0$, the measure is defined as:

$$D_{JS}(P, Q) = \frac{D_{KL}(P, M) + D_{KL}(Q, M)}{2}, \tag{2.51}$$

where

$$M = \frac{P + Q}{2}, \tag{2.52}$$

is the average of P and Q. Furthermore, Lin in [26] proved the mathematical connection between Jensen-Shannon divergence and Shannon's entropy with the following equation:

$$D_{JS}(P, Q) = H(\frac{P + Q}{2}) - \frac{H(P) + H(Q)}{2}, \tag{2.53}$$

where H(X) is the Shannon's entropy measure given by:

$$H(X) = -\sum_j P(x_j) log P(x_j). \tag{2.54}$$

As far as symmetry is concerned,

$$D_{JS}(P, Q) = D_{JS}(Q, P). \tag{2.55}$$

This is due to the fact that, given the Jensen-Shannon divergence definition, the dissimilarity between P and Q is not measured directly, but by the relative entropy between each of the two distributions and their average distribution M, and afterwards by averaging the two relative entropies [26].
Furthermore, there are no cases where $D_{JS}$ is infinite. In KL divergence, infinite

values were produced by cases with $P(X) \neq 0$ and Q(X) = 0, due to the division with zero in (2.43). Let's consider the case of a variable $x_i$ where $P(x_i) \neq 0$ and $Q(x_i)$ = 0.

$$M(x_i) = \frac{P(x_i) + Q(x_i)}{2} = \frac{P(x_i)}{2} \neq 0 \quad (2.47). \tag{2.56}$$

Therefore:

$$log\frac{P(x_i)}{M(x_i)} \neq inf \Rightarrow D_{KL}(P\backslash\backslash M) \neq inf \tag{2.57}$$

and

$$log\frac{Q(x_i)}{M(x_i)} \neq inf \Rightarrow D_{KL}(Q\backslash\backslash M) \neq inf \tag{2.58}$$

The union of equations (2.52) and (2.53) lead to:

$$D_{JS}(Q\backslash\backslash M) \neq inf. \tag{2.59}$$

The same proof is valid for cases where $P(x_i) = 0$ and $Q(x_i) \neq 0$ [26].

### 2.4.3 Bray-Curtis Dissimilarity

Bray-Curtis dissimilarity was introduced by Bray and Curtis in [27], as a dissimilarity measure between the composition of two different communities. Given two samples u and v, the Bray-Curtis dissimilarity measure has been defined as:

$$BC_{uv} = 100\frac{\sum_i |u_i - v_i|}{\sum_i |u_i + v_i|} \tag{2.60}$$

or

$$BC_{uv} = \frac{\sum_i |u_i - v_i|}{\sum_i |u_i + v_i|}, \tag{2.61}$$

where:

- $u_i \geq 0$ is the relative abundance of the $i^{th}$ taxon (def. 1.1) present in sample u,
- $v_i \geq 0$ is the relative abundance of the $i^{th}$ taxon present in sample v.

Both of the definitions have been used to define the Bray-Curtis dissimilarity measure and their difference is in scaling. A sample can be considered as anything with the form of a group or community, while the term relative abundance refers to the presence or absence of a taxon in a community.
The Bray-Curtis dissimilarity measure is frequently used by ecologists and environmental scientists [28]. Specific characteristics regarding the measure are:

1. The measure takes the value zero when $u = v$.

2. The measure takes its maximum value (1 or 100 depending on the definition) when the two samples have no taxa in common.

3. The dissimilarity between two samples is not affected by the inclusion of a third sample in the analysis.

4. The dissimilarity between two samples is not affected by the inclusion or exclusion of taxa absent in both samples.

5. The dissimilarity between two samples is not affected by a simple change in scale of the samples.

6. The measure combines the information of both the total change and relative change between the composition of two samples [28].

A major problem of the measure is its instability when there is a sparsity in the samples. In cases where two samples have only one present taxon (its value is greater than zero), their dissimilarity can vary between 0, if they are from the same species, and 1 or 100 depending on the definition, if they are from different species. Furthermore, in case that all taxa are absent and therefore the two samples contain only zero values, the measure cannot be defined due to the division with zero according to the definition [28].

# Chapter 3

# Spike Prediction



## 3.1 Spike Definition

Before the explanation of the term spike, the clarification of the terms beta diversity and changes in microbial composition is important.

Beta diversity has been defined as the difference in species and genus composition between two communities or sites [29]. The microbial composition or makeup is the microbial genus and species composition of the microbiome at a specific time-point, dictated by the presence and absence of specific bacterial genera and species. Therefore, changes in microbial composition are dictated by changes in the presence and absence of bacteria belonging to the microbiome.

The dynamic behavior of the microbiome indicated cases where the microbial composition changed to a new one and afterwards retreated to the previous one. Double changes of that nature were categorized as spikes. Given two microbial compositions a and b, the spike could be described by the pattern: a-b-a.

Initially, changes in microbial composition were quantified with the use of the beta

diversity. The purpose of this quantification was to identify double changes which could be considered as spikes. The Bray-Curtis dissimilarity (eq. 2.61) was used to measure the beta diversity between two consecutive time-points of the microbiome. Therefore, given two consecutive time-points named $t_1$ and $t_2$, where $t_1 < t_2$, beta diversity was estimated as: beta diversity$_{t_2}$ = $BC_{t_1,t_2}$, where BC is the Bray-Curtis dissimilarity between $t_1$ and $t_2$. In case that a time-point was the first of a time-series, its beta diversity value was zero due to the fact that there was a lack of previous time-point for diversity estimation.

According to the beta diversity approach, the spike was defined by statistical terms. Specifically, beta diversity values were considered as either baseline or outliers (Fig. 3.1). A spike was dictated by a baseline (def. 3.2) beta diversity at a time-point defined as t-1, followed by a rapid increase in the beta diversity of the microbiome on outlier levels (def. 3.1) at time-point t, succeeded by a preservation to outlier levels at time-point t+1 and drop to the baseline levels at time-point t+2. The reason for the outlier restrictions at time-points t and t+1 was due to that the beta diversity indicated dissimilarity. Hence, a change from composition a to b and then back to a again, would be associated with a dissimilarity above baseline levels.

While the above approach indicated changes in composition above baseline and later retreat to baseline levels, it was still not evident whether the retreat was at composition a or a new composition c. In case that it changed to a completely new composition, there could be no interpretation regarding the microbiome dynamics. For that reason the retreat to the initial composition a was assumed.

The aforementioned hypothesis about the changes in beta diversity interpreted as changes in microbial composition, was named as the beta diversity spike hypothesis and served as a way of identifying spikes.

**Definition 3.1.** Outlier criterion for time-point t:
$beta\_diversity_t \leftarrow outlier$ if $beta\_diversity_t > median + SD$.

**Definition 3.2.** Baseline criterion for time-point t:
$beta\_diversity_t \leftarrow baseline$ if $beta\_diversity_t \in [median - SD, median + SD]$.

**Definition 3.3.** Beta Diversity Criterion for spike occurrence at time-point t:
$beta\_diversity_{t-2} \leftarrow baseline \cap beta\_diversity_{t-1} \leftarrow baseline \cap beta\_diversity_t \leftarrow outlier \cap beta\_diversity_{t+1} \leftarrow outlier \cap beta\_diversity_{t+2} \leftarrow baseline$

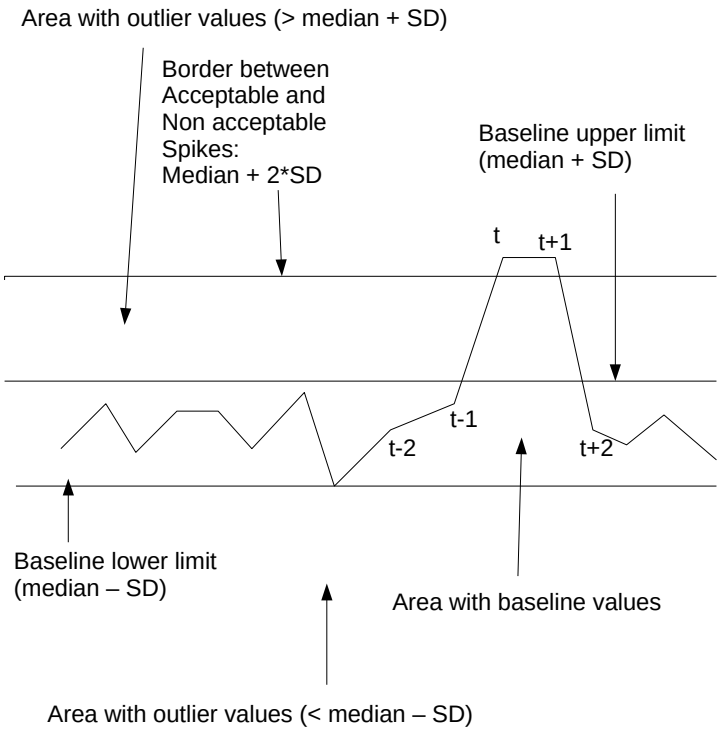


Figure 3.1: Detection of Spikes with the Beta Diversity Spike Hypothesis. X-axis: time-points values. Y-axis: beta diversity values. The first two horizontal lines from the bottom indicated the borders between the baseline and the outlier areas. The baseline area was defined as: [median - SD, median + SD ] (SD: standard deviation). The third horizontal line from bottom indicated the border between spikes and non-spikes.

For a value to be considered as a spike (fig. 3.1), it must have been regarded as an outlier (definition 3.1). Cut-off values of that nature were commonly used in the micro-array data analysis [30]. The time-point in which a spike occurred was referred to as time-point t. The previous time-points were referred to as time-points t-1, t-2, t-3, etc. Similarly the next time-points were referred to as time-point t+1, t+2, etc. As figure 3.1 indicated, acceptable spikes were spikes belonging to a subset of time-points ranging from time-points t-2 until t+2 and satisfying the beta diversity spike criterion (definition 3.3).

## 3.2 Research Goal

The goal of the research was the investigation of whether future spike occurrences could be predicted. This goal was based on the hypothesis that there existed bacteria from the human vaginal microbiome, whose relative abundances could be used to predict spikes.

The term prediction implied a temporal relationship between the predictor and predicted variables. Therefore, for a spike occurrence to be predicted by bacteria values, the values had to be present at an earlier time-point than the spike. In case that in a given time-series a spike occurrence was at time-point t, the abundances at earlier time-points t-1, t-2 and t-3 were considered as possible candidates for testing the prediction hypothesis. The average temporal distance between two time-points of the dataset was 4 days. Therefore, time-points t-1,t-2 and t-3 indicated an approximate time-frame of maximum 12 days before the spike. This time-frame was considered suitable for prediction, as a time-frame greater than 12 days was considered too distant for any correlation.

The occurrence of spikes was identified with the beta diversity spike hypothesis (section 3.1). Moreover, machine learning classification methods (section 2.2) were used for investigating the spike prediction hypothesis. Therefore, classifiers were built and applied a dataset, consisted of a number of bacterial relative abundances at various time-points (t-1, t-2 and t-3).

## 3.3 Dataset

As mentioned in chapter 1.1, the initial longitudinal dataset contained information about the evolution of the vaginal microbiome of 32 subjects for 16 weeks, sampled at approximately 30 time-points per subject. Therefore, by considering as an example the microbiome of a given subject at a given time-point, the dataset contained 937 examples in total. For each example, the microbiome consisted of the relative abundances of 330 bacteria. Therefore the dataset features were 330 in total (section 1.1). A number of steps were taken for pre-processing the aforementioned dataset and investigating the beta diversity spike hypothesis:

1. Subset selection step:

   Each subset of 5 consecutive time-points from each time-series was labeled as positive or negative depending on if it met the beta diversity spike criterion or not ( 3.3). In the time-series of figure 3.1 for example, the subset of time-points $\in [t-2, t+2]$ would be labeled as positive due to the fact that it met the beta diversity spike criterion. On the other hand, every other subset from that figure would be included in the negative category.

2. Overlap removal step: the subset selection step was performed with a sliding window approach, meaning that all subsets of time-points were categorized consecutively, with the second time-point t-2 of a given subset to be time-point t-3 of the next one. Therefore, there existed cases with various subsets of the same class belonging to close time-frames and thus having similar values. For that reason a time-constraint was placed between subsets of the same class. Subsets being closer than 6 time-points from chronologically older subsets were removed. The same was applied for negative class subsets being close to positive class subsets, in order to avoid cases where the time-points used for prediction in the negative class were associated with a spike.

3. Time-point selection step: From each five consecutive time-points selected, the first corresponded to time-point t-2, the second to time-point t-1, the third to time-point t, the fourth to time-point t+1, the fifth to time-point t+2. Therefore, the spike positive/negative time-point was the third one. For predicting the spike, the data points used for analysis would have to belong to a time-point earlier than the third one. There were was a selected time-frame $\in [t-1, t-3]$ which corresponded to how much back in time was the prediction analysis desired to be performed. Therefore, in this step the desired time-point for analysis was selected.

4. Feature selection step: The complete dataset was very sparse, which was due to the absence of the majority of bacteria at a given time-point. The bacteria constituted the features of the dataset. For that reason, from each dataset created at the previous step, there was always a number of features whose values were

zero for all examples of the dataset. Therefore these features, were removed from the analysis.

5. Classification step: Given the two-class dataset, constituted from the previous steps, the next step was the application of classification algorithms for the prediction of spikes. The classification algorithms used in the analysis were: decision trees, k-nearest neighbor, linear discriminant classifier and support vector machines (section 2.2). The classifiers were evaluated with the leave-one-out Cross-Validation method (eq. 2.4), while the classification measure used was the accuracy measure, since the datasets were balanced with respect to classes.

## 3.4 Experimental Results

Each step mentioned in section 3.3 for the analysis of spike prediction, was analyzed separately as shown in the next subsections. The results contain visual representations of the analysis, followed by a discussion of the results.

### 3.4.1 Subset selection step:

In this step, subsets of time-series were selected and categorized as positive or negative, based on the beta diversity spike criterion (def. 3.3), as mentioned in section 3.1. Figures 3.1 and 3.2 contained examples of positively categorized examples.

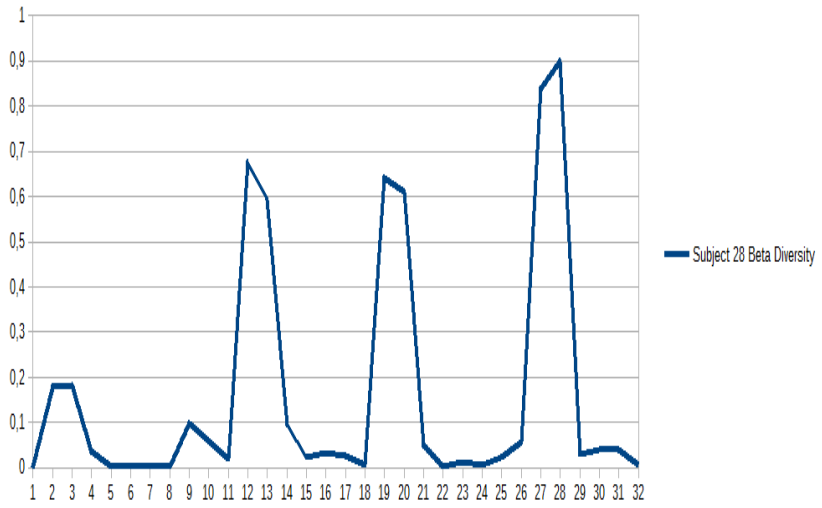


Figure 3.2: Indication of the beta diversity spike hypothesis in subject 28. X-axis: time-points. Y-axis: beta diversity values. The beta diversity values have either a normal deviation from the mean value (also referred to as a baseline level) or a deviation exceeding the mean value and the standard deviation (also referred to as an outlier 3.1), leading to a spike-like figure which was labeled as a spike. Furthermore, the spike criterion (fig. 3.1) is met three times (time-points 10-14, 17-21, 25-29), due to the fact that in at least two time-points before and after each spike, the values have a normal deviation from average. An example is the sequence of time-points 9-13 in both panels. Subsets like these were considered as positive ones and were labeled in the subset selection step.

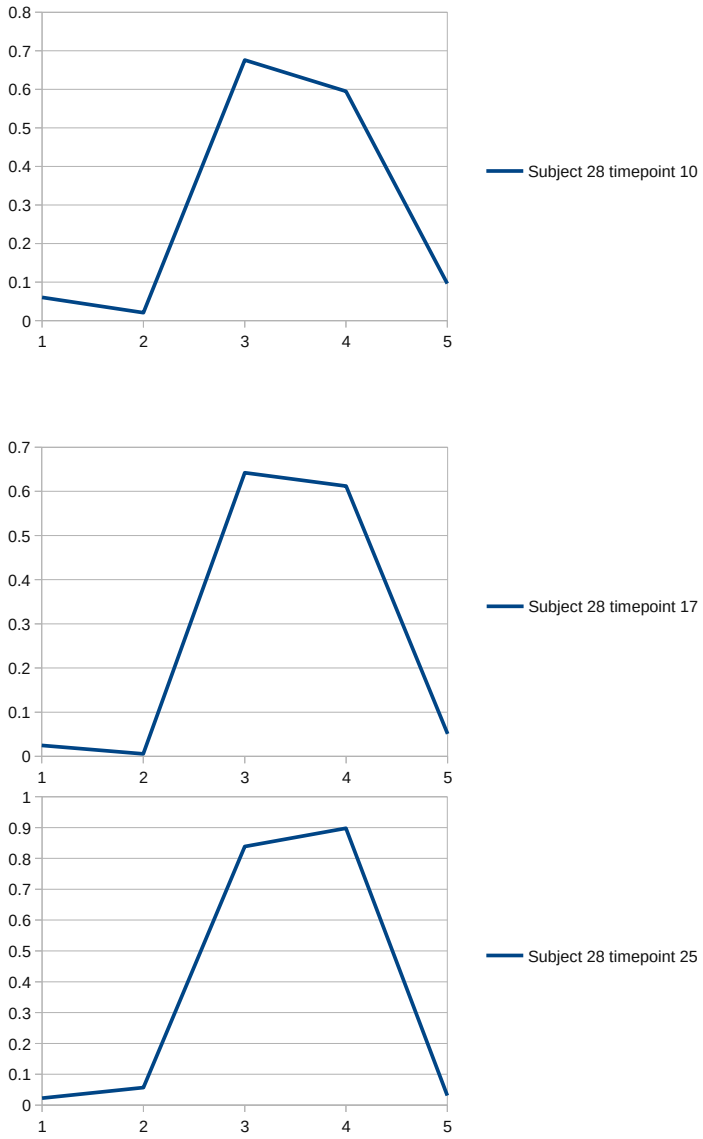


Figure 3.3: Selected time-series subsets of subject 28. X-axis (in all panels): time-points. Y-axis (in all panels): beta diversity values. The consecutive time-points meet the beta diversity spike criterion and thus are labeled as positive. The subsets, as referenced by the labels on the right are identified by their subject id and a time-point number corresponding to tp1 on the x-axis. This number refers to two time-points before the spike and thus is time-point t-2. For example in the first panel, time-point 10 of subject 28 is time-point t-2 and thus the spike occurs at time-point 12.

### 3.4.2 Overlap removal step:

As mentioned in section 3.3, selected subsets that were close to previous subsets less than 6 time-points were removed. From the initial selected subsets of the dataset, the remaining subsets after removal were 72. From the 72 examples, the 37 were labeled as positives and the rest 42 as negatives.

### 3.4.3 Time-point selection step

For each selected subset, the third one was the spike positive or negative one. The spike positive/negative ones was named as time-point t (section 3.1). A time-frame $\in$ [t-3,t-1] was selected since it indicated approximately 12 days before the spike occurrence. Earlier time-points were considered two early for an objective prediction to be possible. Therefore, each of the three time-points were assembled from all subsets and constructed 3 separate datasets. Since the subsets were 79 in total, each of the three datasets contained 72 examples, with 37 positive and 42 negative ones. Those datasets were named as: $D_{t-3}$, $D_{t-2}$ and $D_{t-1}$.

### 3.4.4 Feature selection step:

From the total 330 features of the three datasets, the remained ones after the feature selection step were:

1. $D_{t-3}$: 179 features,
2. $D_{t-2}$: 182 features,
3. $D_{t-1}$: 188 features.

### 3.4.5 Classification step:

The following classification models (section 2.2) were applied to the datasets $D_{t-3}$, $D_{t-2}$ and $D_{t-1}$:
*Decision tree*,
*k-NN* with the Euclidean distance measure,
*k-NN* with the Jensen-Shannon divergence measure,

*k-NN* with the Bray-Curtis dissimilarity measure,

*Linear discriminant classifier*,

*SVM* with RBF kernel function,

*SVM* with linear kernel function,

*SVM* with Jensen-Shannon kernel function:

$$k(x_n, x_m) = exp(-\frac{JS(x_n, x_m)}{\sigma}), \tag{3.1}$$

where $\sigma$ is the scaling factor and $x_n$, $x_m$ are two data points of the dataset,

*SVM* with Bray-Curtis kernel function:

$$k(x_n, x_m) = exp(-\frac{BC(x_n, x_m)}{\sigma}), \tag{3.2}$$

where $\sigma$ is the scaling factor and $x_n$, $x_m$ are two data points of the dataset.

*Stacked-Autoencoders* with two autoencoders.

Bellow are the results of the classification performance per dataset, for all aforementioned applied classification models:

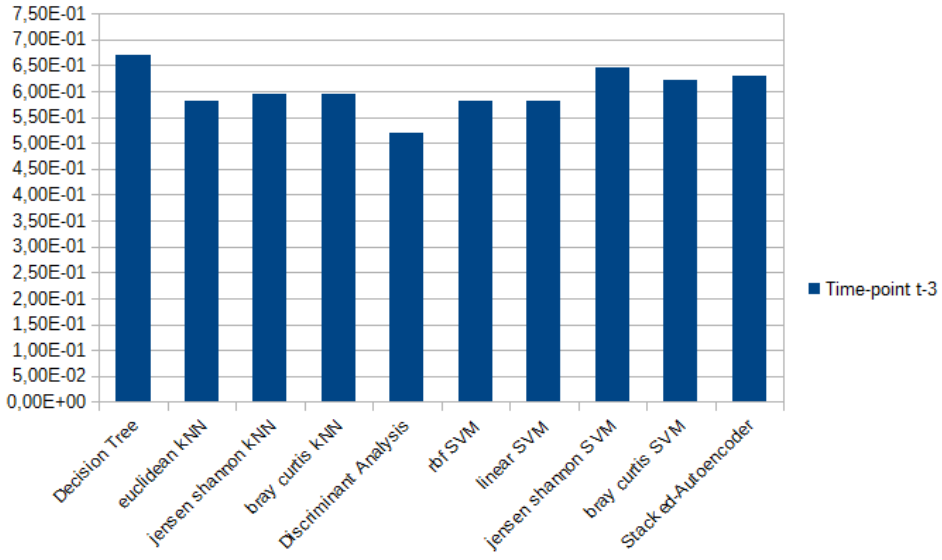


Figure 3.4: Performance of a number of classifiers for the spike prediction at t-3. X-axis: names of the classifiers. Y-axis: LOT average accuracy. The generalization performance of the models was evaluated with the leave-one-out CV method. The highest classifier in terms of performance is the decision tree with 6.71e-01 accuracy.

The optimal parameters applied to each model used in figure 3.4 were:
*Decision tree*: minimum size of leaves = 6, minimum size of parent nodes: 1,
*Euclidean k-NN*: nearest neighbor number = 1,
*Jensen-Shannon k-NN*: nearest neighbor number = 3,
*Bray-Curtis k-NN*: nearest neighbor number = 3,
*Linear discriminant classifier*: discriminant type = diagonal covariance matrix,
*RBF SVM*: box constraint = 10, kernel scale = 6,
*Linear SVM*: box constraint = 10, kernel scale = 10,
*Jensen-Shannon SVM*: box constraint = 1000, kernel scale = 8,
*Bray-Curtis SVM*: box constraint = 100, kernel scale = 6. *Stacked-Autoencoders*: number of hidden layers = 2, number of hidden neurons (hidden layer 1) = 16, number of hidden neurons (hidden layer 2) = 15 , encoder activation function (both hidden layers) = saturating linear, decoder activation function (both hidden layers) = saturating linear, L2 regularization = 10e-04, sparsity regularization $\beta$ = 10e-04, sparsity proportion h = 0.05.

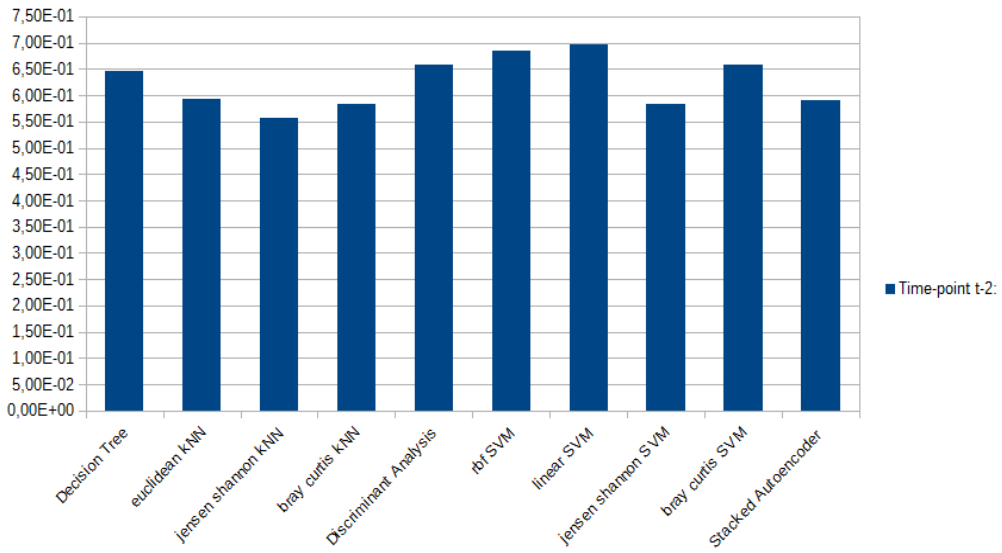


Figure 3.5: Performance of a number of classifiers for the spike prediction at t-2. X-axis: names of the classifiers. Y-axis: LOT average accuracy. The generalization performance of the models was evaluated with the leave-one-out CV method. The highest classifier in terms of performance is SVM with linear kernel function having 6.96e-01 accuracy.

The optimal parameters applied to each model used in figure 3.5 were:
*Decision tree*: minimum size of leaves = 8, minimum size of parent nodes: 22,
*Euclidean k-NN*: nearest neighbor number = 9,
*Jensen-Shannon k-NN*: nearest neighbor number = 1,
*Bray-Curtis k-NN*: nearest neighbor number = 3,
*Linear discriminant classifier*: discriminant type = diagonal covariance matrix,
*RBF SVM*: box constraint = 1, kernel scale = 7,
*Linear SVM*: box constraint = 1, kernel scale = 1.0e-05,
*Jensen-Shannon SVM*: box constraint = 1, kernel scale = 1,
*Bray-Curtis SVM*: box constraint = 1, kernel scale = 1,
*Stacked-Autoencoders*: number of hidden layers = 2, number of hidden neurons (hidden layer 1) = 10, number of hidden neurons (hidden layer 2) = 8, encoder activation function (both hidden layers) = saturating linear, decoder activation function (both hidden layers) = saturating linear, L2 regularization = 10e-04, sparsity regularization $\beta$ = 10e-02, sparsity proportion h = 0.5.

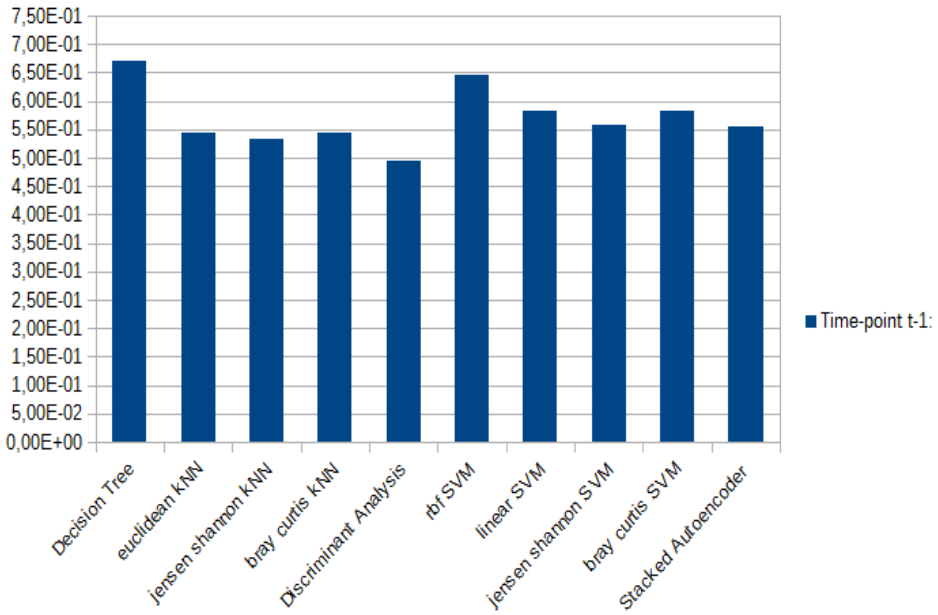


Figure 3.6: Performance of a number of classifiers for the spike prediction at t-1. X-axis: names of the classifiers. Y-axis: LOT average accuracy. The generalization performance of the models was evaluated with the leave-one-out CV method. The highest classifier in terms of performance is the decision tree having 6.71e-01 accuracy.

The optimal parameters applied to each model used in figure 3.6 were:
*Decision tree*: minimum size of leaves = 8, minimum size of parent nodes: 18,
*Euclidean k-NN*: nearest neighbor number = 1,
*Jensen-Shannon k-NN*: nearest neighbor number = 1,
*Bray-Curtis k-NN*: nearest neighbor number = 8,
*Linear discriminant classifier*: discriminant type = diagonal covariance matrix,
*RBF SVM*: box constraint = 1000, kernel scale = 7,
*Linear SVM*: box constraint = 1000, kernel scale = 10,
*Jensen-Shannon SVM*: box constraint = 1, kernel scale = 1,
*Bray-Curtis SVM*: box constraint = 1000, kernel scale = 3,
*Stacked-Autoencoders*: number of hidden layers = 2, number of hidden neurons (hidden layer 1) = 36, number of hidden neurons (hidden layer 2) = 35, encoder activation function (both hidden layers) = saturating linear, decoder activation function (both hidden layers) = saturating linear, L2 regularization = 10e-03, sparsity regularization $\beta$ = 15e-02, sparsity proportion h = 10.

Table 3.1: Best classifiers and accuracies per dataset.

| Dataset | Best Model Accuracy | Model Name |
|---|---|---|
| $D_{t-1}$ | 6.71e-01 | decision tree |
| $D_{t-2}$ | 6.96e-01 | linear-kernel SVM |
| $D_{t-3}$ | 6.71e-01 | decision tree |

Table 3.1 indicated for all three datasets the highest classification accuracy close to 0.7. Therefore, according to the results there was an optimal prediction accuracy close to 0.7 for time-points t-3, t-2 and t-1 before the spike-positive/negative time-point t. Furthermore, the optimal prediction accuracies for t-3,t-2 and t-1 were close to each other (0.67, 0.69 and 0.67), with the highest one corresponding to time-point t-2. The classification results indicated a sub-optimal prediction of spikes. For that reason, new ways of measuring changes in microbial composition were investigated, as shown at the next chapter.

# Chapter 4

# Detecting Predictable Subsets



## 4.1 Introduction

Following the results of chapter 3 (table 3.1), a new research question arose. The research question was whether there exist subsets of a given dataset, whose classification performance for a given classifier is greater than that of the complete dataset. In case that such a subset exists, the examples should not have been selected randomly for an interpretation of the results to be possible. Therefore, the identification of a deterministically selected subset with improved classification performance over the complete dataset, implies the existence of a selection criterion that differentiates this subset from the rest of the dataset.

Furthermore, for a deterministically selected subset to exist, its selection criterion

must be based on a property of the examples. A possible approach would be ranking the dataset examples based on such property and assembling a subset with the most important ones, either positive or negative. Initially however, this measurable criterion had to be defined. For that reason, new ways of characterizing spikes were investigated as shown at the next section.

## 4.2 Spikeness measure

The spike was defined in binary terms, so a given time-series subset was either considered to have a spike or not. But what about a time-series subset whose composition was not clear on whether it changed significantly or not? For that reason spikeness was defined as a continuous ordinal measure which measured the amount of change in microbial composition at a given time-series subset.

Continuous values could provide a more sensitive measure for estimating changes in the microbial composition than a threshold value. Therefore, spikeness could be considered as a more sensitive approach to defining spikes than using thresholds based on dissimilarity indices. Hence, high spikeness values were high indicators of a spike occurrence. On the other hand prediction of low spikeness values could still reveal important information, given the fact that even a small decline from the baseline diversity levels could be indicative of changes in microbial composition.

There are were two possible approaches by which spikeness could be defined. The first was the use of absolute differences between two consecutive time-points as a measure of change, with the use of a dissimilarity measure like Bray-Curtis (eq. 2.61) or Jensen-Shannon (eq. 2.51).

The second way was the use of relative differences based on a baseline value. In this approach the absolute compositional difference between each time-point and its previous one, were taken as a first step and represented the diversity at each time-point. At the second step, the standard deviation of the diversity was estimated through a local sliding window. Therefore, for sliding window size equal to n, the standard deviation of time-point t was estimated by taking into account time-points in the range $\in [t-n-1, t+n-1]$. The sliding window approach was preferred over estimating the total standard deviation of a time-series, due to the fact that it took into account

local changes in diversity which were lost when taking all time-points into account. The final step was the division of all diversities with their local standard deviation to achieve normalization.

At the initial analysis, the second approach was preferred because the relative differences provided information of difference with regards to the surrounding differences. A value of 3 indicates a diversity which exceeds 3 times the standard deviation and is in direct contract to a value of 0.5 which indicates a diversity close to the average. The final step was common in both approaches. By having the diversities of time-points estimated in absolute or relative terms, spikeness was estimated as the minimum value from the diversities of two consecutive time-points. Therefore, spikeness was given by the following definition:

$$spikeness = min(\frac{diversity_t}{SD}, \frac{diversity_{t+1}}{SD}), \tag{4.1}$$

in case of relative differences or

$$spikeness = min(diversity_t, diversity_{t+1}) \tag{4.2}$$

in case of absolute differences. The reason why two time-points were used in the process instead of one, was due to the fact that a spike was defined by a change in a new state, quickly followed by the retreat to the old one. In diversity terms, it was defined as a notable increase in high diversity which indicated the change to a new state, followed by high diversity one time-point later which indicated the change to the old state (section 3.1). Therefore, for a spike to be quantified, the lowest of the two values had to be higher than the baseline levels. By comparing spikeness to the quantification of spikes with the beta diversity spike criterion (definition 3.3, section 3.1), the two time-points used for spikeness estimation would represent the time-points t and t+1. Pairs of time-points with high spikeness value were expected to be associated with spike occurrences and pairs of time-points with low spikeness value were expected not to be associated with spikes.

## 4.3 Rank-based example selection

Rank-based example selection was defined as the process of selecting a subset of the dataset based on a ranking that has been applied on its examples.

The measure used to rank the dataset was spikeness. The ranking was at descending order, hence the highest ranking examples were considered as top examples, while the lowest ones were considered as bottom examples.

Two assumptions were made following the ranking of the dataset based on spikeness (eq. 4.1). The first assumption was that examples which were close or equal to the highest ranked one, had a significantly different feature distribution than the ones which were closest or equal to the lowest ranked one, and could be classified more effectively than other subsets of a dataset given a classifier. Since spikeness implied a rate of change, examples with low spikeness were considered to have a significantly different microbiome from the ones with high spikeness. The same assumption implied that examples which were ranked in a middle zone, had similar feature distribution and could be classified more ineffectively than the rest. A dataset subset which included the examples with a medium ranking was named as the gray-zone.

The second assumption was that examples labeled as positives would be ranked higher than the negative ones. This assumption was due to the fact that spike-positive examples indicated high spikeness and vice versa. If this assumption was false, then it would imply an incorrect categorization of the examples. However, the categorization made was based on an approximate approach (section 3.1) and hence was prone to errors. For that reason, there was expected to exist some sort of overlap between the two classes, with examples of the positive class being ranked lower than examples of negative ones.

The combination of the two assumptions lead to many possible scenarios. The worst case scenario was that the the top examples of the positive class had lower or equal spikeness value on average than the top examples of the negative class. In this scenario, the classes of the dataset were considered to be similarly distributed and thus a satisfiable classification was not possible.

The best case scenario was that all top examples of the positive class had higher spikeness value than all top examples of the negative class. Moreover, all bottom examples of the negative class had lower spikeness value than all bottom examples of the positive class. Therefore, the bottom positive and top negative examples were expected to have a closer or equal ranking to the middle one than the highest and lowest ones, thus being situated in the middle of the ranking spectrum. In this scenario, difficulty in classification between the two classes was expected to be, if any, in

the examples belonging to that middle area. The middle area with the mixed positive and negative examples was, in this case, the gray-zone.
As far as the methodology for rank-based selection is concerned, given a ranking threshold $\alpha$ and a dataset D with two classes $C_1$ and $C_2$:

1. rank the examples for $C_1$ and $C_2$ separately,
2. select the top $\alpha$% examples of the $C_1$ and the bottom $\alpha$% examples of $C_2$,
3. construct a new dataset $D_{selected}$, based on the selected examples of the previous step and their corresponding labels,
4. classify $D_{selected}$ given a classification model,
5. evaluate the classification performance of the model.

In case that the classification performance of a model was considered unsatisfactory, meaning that it did not exceed a desired threshold, then: either the percentage of selected ranked examples included examples from the gray-zone, or the feature distribution between the two classes was highly similar.

## 4.4 Black-Box Classifier

A black-box classifier has been defined as a classification system consisted of a set of various classification models and external model parameters, in which the classification result for a given dataset is obtained from the best performing model.
A black-box classifier was different than ensemble classifiers. In ensemble classifiers, classification has been performed by using an ensemble of different parameters, different subsets of training examples and different classification methods, and the final prediction of an unknown example has been given by the majority vote of all trained models of the ensemble [31]. In black-box classification, classification was performed by using a set of different parameters, all training examples and different classification methods, and the final prediction of an unknown example was given not from voting but by the trained model with the best classification accuracy.
A black-box has been defined as a system that given an input, provided an output without observability of the internal procedures. The same definition was applied to the black-box classifier. While a number of different classification methods with

external model parameters were being applied to a given dataset, the only output provided was the optimal classification accuracy, without knowledge of the classification method or the parameters used. Hence, the system was a black-box classifier. The following steps describe the process being used for black-box classification:

**Algorithm 4.1** Black-Box Classification

**Input**: $X$: Dataset of size NxM.

$y$: labels of X.

Models: Vector of size KxL, where $Models_{ij}$ corresponds to the $i^{th}$ classification model applied to the system $j^{th}$ combination of parameters for model i.

$LOT\_CV$: leave-one-out Cross-Validation measure.

**Output**: $Generalization\_Performance$: generalization performance of the Black-Box classifier.

$accuracy \Leftarrow \emptyset$

**for** $i \leftarrow 1; i \leftarrow K; i \leftarrow i + 1$ **do**

  **for** $j \leftarrow 1; j \leftarrow L; j \leftarrow i + 1$ **do**

    $accuracy \cup LOT\_CV(models_{ij}, X, y)$, meaning that the $i^{th}$ model with the $j^{th}$ combination of parameters is applied to the dataset (X,y) and evaluated with $LOT\_CV$.

  **end for**

**end for**

$Generalization\_Performance \Leftarrow max(accuracy)$, meaning that the maximum leave-one-out accuracy found in the analysis is considered as the Generalization_Performance of the Black-Box classifier.

The main advantage of the black box classifier was that, given the limitations of the applied dataset and classifiers, the classification performance would be as high as possible. This was due to the use of different and varying classification methods, and the exhaustive trial of different possible parameters.

The main disadvantage was its heavy computational cost and the time complexity that the method had. This was obvious due to the sum of the time complexities and computational expenses of all models participating at the ensemble. The main reason why this approach was preferred for classification in the analysis despite its disadvantage, was that the datasets applied were small ($< 500$ examples and $< 300$

features), and thus the method did not slow down the experimentation process.

## 4.5 Predictability

Predictability was defined as the generalization performance of an optimal classification system built using a given dataset $(X, y) \in D$, where X is the dataset matrix and y is the label vector, and tested with a given evaluation measure. Since there was not a deterministic approach of constructing an optimal dataset, it could only be constructed approximately. The inclusion of more classification methods with more possible tested parameters per method, would increase the probability of a better generalization ability, leading to a more optimal system and a more objective predictability.

In terms of practical application, predictability was tested on the dataset with the use of the black-box classifier. The black-box classifier was designed, constructed and tested based on the idea of predictability, serving as an approximation of the optimal classification system. Furthermore, the measure used for estimating the generalization ability was accuracy, while the evaluation measure used was leave-one-out (LOT) Cross-Validation (eq. 2.4).

There was no prior knowledge regarding the predictability of a dataset and the only way of estimating it was by applying it to a classification system like the black-box classifier. The black-box classifier provided a heuristic approach for estimating predictability, while a more optimal approach was not possible. For that reason, the black-box classifier was assumed to estimate the highest possible predictability for a dataset. Given that assumption, the focus shifted on finding datasets with high predictability.

Since predictability implied an optimal classification performance, it was used as a measure for investigating the research question of section 4.1: whether there exist subsets of a given subset, whose predictability is higher than that of the complete dataset.

## 4.6 Rank-based vs random-based predictability

Predictability was used as a measure for investigating the research question of section 4.1. The question would be investigated by the partition of the dataset to a number of subsets, estimation of their predictability and comparison of their results with the complete dataset predictability. There existed multiple ways of selecting subsets for the approach. However, as mentioned in section 4.1, the subset selection of a given dataset could be either random or by a selection criterion. For the second case, rank-based selection with the spikeness measure was used as a criterion (eq. 4.1). Therefore, the two possible ways for subset selection were named as: random-based and rank-based predictability.

In random-based predictability, the idea was the selection of multiple subsets from a given dataset, by choosing random examples for each subset. The subsets would contain an equal number of positive and negative examples for it being balanced as a dataset. The predictability of each subset would be estimated with the black-box classifier. Finally, each subset would be compared with the complete dataset in terms of predictability. As mentioned in section 4.1, the problem with a random-based predictability was the lack of interpretation in the results. Even if the dataset had a lower predictability than a randomly selected subset, the results did not provide an answer to the question. This was due to that a proper interpretation for the quantitative difference in the results could not be given. Also, a possible explanation for the existence of such a subset was a highly noisy dataset that provided no value to the analysis.

On the other hand, rank-based predictability provided a selection criterion that ranks the examples. Therefore, in case that such a subset existed and was selected with the rank-based approach, then the results could be contributed to the ranking of the selected examples. Last but not least, its predictability must have been equal or higher than randomly-selected ones, in order to be considered significant. In the opposite case, the results were not significant for the reasons mentioned in the previous paragraph. In case that it did exceed the randomly-selected ones, then the ranking provided a possible interpretation to the results.

One more important information about rank-based predictability, was that the size of predictable subset was associated with its importance. This was due to the fact that big subsets made up a larger proportion of the dataset than smaller ones. Therefore, big

predictable subsets indicated that a large proportion of the dataset was predictable. In the case that the complete dataset was not considered predictable, subsets like the aforementioned could still provide quantitative information regarding its predictability, which would be lost if a rank-based selection approach was not applied.
In rank-based predictability, the examples of a given dataset were ranked based on their spikeness. Afterwards, the most important ones given selected and assembled a subset, whose predictability was estimated with the black-box classifier. Since there was not a strict definition on example importance, their importance was defined by their ranking. In section 4.3, the idea of the gray-area was introduced which was composed by a number of examples, which had a highly similar feature distribution and thus were difficult in being classified. In this analysis, the gray-area was assumed to be composed by examples having a ranking close to the mean. Examples with a ranking closer or equal to the highest and lowest ones, were named as the top and bottom examples and assumed to be classified more easily than the gray-area ones. Their difference in classification lead to a difference in predictability, since predictability was associated with classification performance. For that reason, top and bottom examples were considered more important in rank-based predictability.
However, it was still not evident which threshold could define the separation of top and bottom examples from the gray-zone. The problem was that such a threshold was highly dependable on the feature distribution of each dataset. For that reason, an incremental approach was applied:

1. the examples for $C_1$ and $C_2$ were ranked based on spikeness with descending order, where $C_1$ and $C_2$ are the two classes of the dataset.

2. A value k was defined. The k-highest ranked examples of the positive class were considered as top, and the k-lowest ranked examples of the negative class were considered as bottom. The rest of the examples were considered as the gray-zone.

3. The top and bottom examples constructed the subset k, $D_k$.

4. The predictability of $D_k$ was estimated as the optimal classification accuracy of the black-box classifier, where subset k was given as input to it.

5. if $D_k$ was greater or equal to the generalization threshold, where generalization threshold was given as user input, then the algorithm proceeded to step 7.

6. if predictability of $D_k$ was less than the generalization threshold, then the algorithm halted.

7. k = k + 1

8. Step number 3 was repeated.

9. The iterations continued until k was equal to the number of positive examples.

With the aforementioned approach, the predictability of a big number of subsets could be estimated. For big datasets however, this approach was computational and time demanding. Hence, the predictability threshold was used for filtering out insignificant subsets. A subset having less predictability than the complete dataset could not provide an answer to the research question of 4.1 and was thus considered insignificant. Moreover, there was a lack of a measure which quantified predictability in terms of the size of the dataset. Two measures were included in the analysis for that purpose, named as total coverage and positive coverage. Total coverage was defined as the proportion of total examples in the dataset which were included on the largest predictable subset $D_k$. Positive coverage was defined as the proportion of positive examples in the dataset which were included on the largest predictable subset $D_k$. Positive and total coverage provided a link between predictability and dataset size. Therefore, besides the information on the number of predictable subsets detected, the coverages provided information regarding the predictable portion of the dataset. Hence, the two coverages along with predictability were returned as outputs of the rank-based predictability approach.
Finally, the incremental approach was based on the assumption of the gray-zone and its estimation was provided by the algorithm: the algorithm started with a small number of examples belonging to the extreme ends of the ranking spectrum and incrementally included more examples being closer to the mean than the highest and lowest one in terms of ranking. As soon as a subset was considered as insignificant, the latest examples included were considered to be part of the gray-zone. Therefore, the gray-zone had been found. In case that the maximum number of positive examples had been reached, then two possible hypotheses existed:

1. there was no gray-zone,

2. the gray-zone existed, but it was composed by negative examples not being

selected by the algorithm. These examples were ranked higher than the higher selected ones, and their inclusion would cause an imbalanced dataset.

As shown at the next chapter, the $2^{nd}$ hypothesis was investigated by the inclusion of more negative examples, creating an imbalanced dataset and new approaches were used for its optimal classification.

## 4.7 Experimental Results

**Spikeness**:

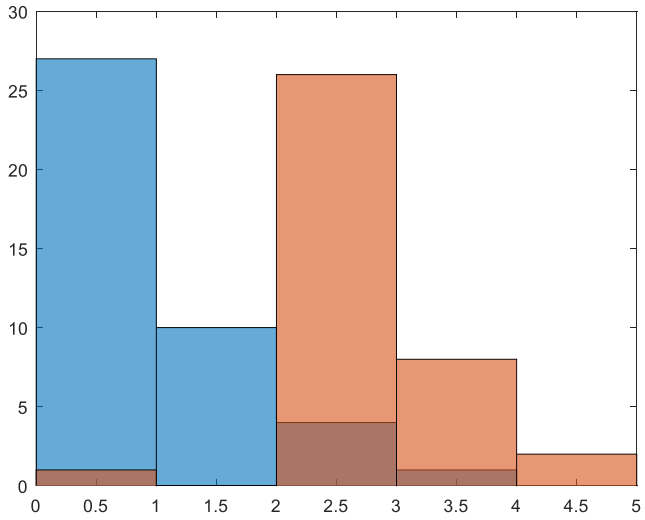


Figure 4.1: Histogram of spikeness values for $D_{t-2}$. X-axis: number of examples with a spikeness value in a given range. Y-axis: spikeness values. The blue color corresponds to the negative class. The red color corresponds to the positive class. There is a small portion of positive examples overlapping with negative ones at the first bin, with the lowest spikeness examples, and there is a moderate overlap at the middle bin. Hence, the gray zone is quite big on that dataset and indicates a non-optimal data categorization.

**Rank-based example selection**:

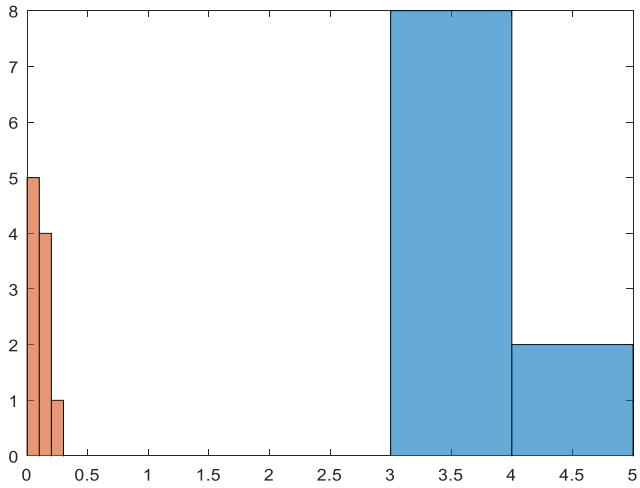


Figure 4.2: Histogram of spikeness values for $D_{t-2}$ - top 10 positive and bottom 10 negative selected examples. X-axis: number of examples with a spikeness value in a given range. Y-axis: spikeness values. The blue color corresponds to the positive class. The red color corresponds to the negative class. There is no overlap in the spikeness between the two classes, which indicates a better data categorization than the equivalent one of 4.1.

The optimal parameters applied to each classifier (classifiers from section 2.2) used in rank-based example selection were:
*Decision tree*: minimum size of leaves = 4, minimum size of parent nodes: 1,
*Euclidean k-NN*: nearest neighbor number = 5,
*Jensen-Shannon k-NN*: nearest neighbor number = 1,
*Bray-Curtis k-NN*: nearest neighbor number = 7,
*Linear discriminant classifier*: discriminant type = diagonal covariance matrix,
*RBF SVM*: box constraint = 1.0e-05, kernel scale = 9,
*Linear SVM*: box constraint = 1.0e-05, kernel scale = 0.1,
*Jensen-Shannon SVM*: box constraint = 1, kernel scale = 1,
*Bray-Curtis SVM*: box constraint = 1.0e-05, kernel scale = 1.
*Stacked-Autoencoders*: number of hidden layers = 2, number of hidden neurons for hidden layer 1 = 10, number of hidden neurons for hidden layer 2 = 8, sparsity regularization $\beta$ = 1000, L2 regularization = 1000, sparsity proportion h = 1, encoder activation function (for both hidden layers) = saturating linear function, decoder

activation function (for both hidden layers) = saturating linear function.
The selection of the top 10 positive and bottom 10 negative examples of dataset $D_{t-2}$, indicated a lack of overlap between the spikeness values of two classes (fig. 4.2). The classification performance of various classifiers on the subset (fig. 4.3) was higher in all cases than the complete dataset one (fig. 3.5), with a notable difference between their optimal accuracies (0.69 at the complete dataset and 0.85 at the subset). This result indicated a correlation between the ranking of a subset's selected examples and classification performance. For that reason, predictability was used for measuring the performance of the selected subsets.

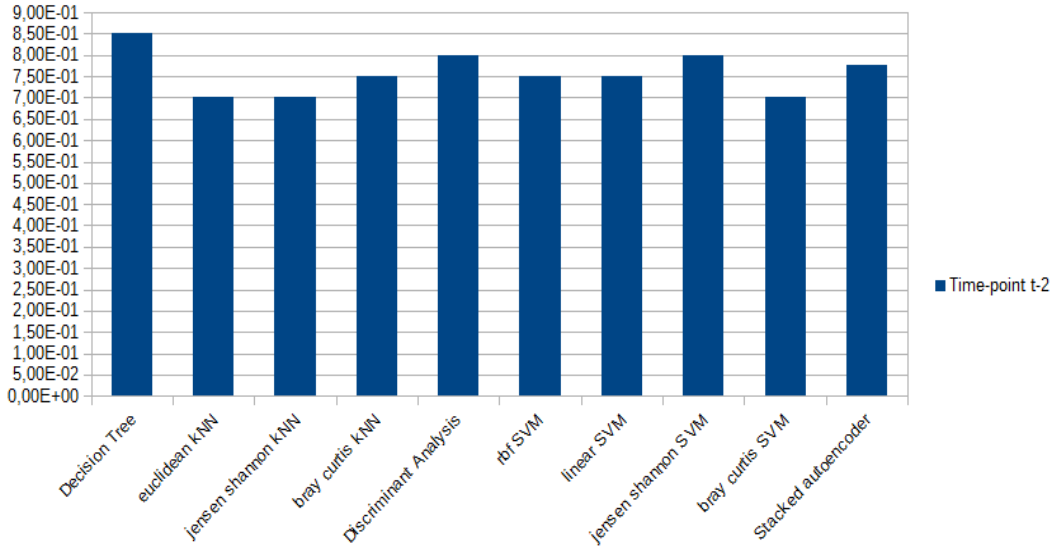


Figure 4.3: Performance of a number of classifiers for the spike prediction at t-2 - top 10 positive and bottom 10 negative selected examples. X-axis: names of the classifiers. Y-axis: LOT average accuracy. This dataset is the result of rank-based selection using the spikeness measure and selecting the top 10 positive and bottom 10 negative examples. The generalization performance of the models was evaluated with the leave-one-out CV method. The highest classifier in terms of performance is the decision tree with 0.85 accuracy.

**Black-box classifier and Predictability**:
Figures 3.4, 3.5 and 3.6 were indicative examples of a black box-classifier. In case that the shown classifiers operated inside a black-box system, with the dataset as input

and the optimal accuracy as output, then all these figures could indicate the internal processes of a black-box classifier. One more detail was that such a system must have also been experimented with as many different combinations of external model parameters as possible, and keep the model with the best performance. Furthermore, in section 4.5 it was mentioned that predictability was estimated with the use of a black-box classifier as its output. Therefore, by considering the classifiers of figures 3.4, 3.5 and 3.6 as internal processes of a black-box classifier with datasets $D_{t-3}$, $D_{t-2}$ and $D_{t-1}$ as input, then the predictability for each case would be: 6.71e-01, 6.96e-01 and 6.71e-01. The classification models applied in rank-based example selection for $D_{t-2}$ were used to construct the Black-box classifier.

**Rank-based vs random-based predictability**:

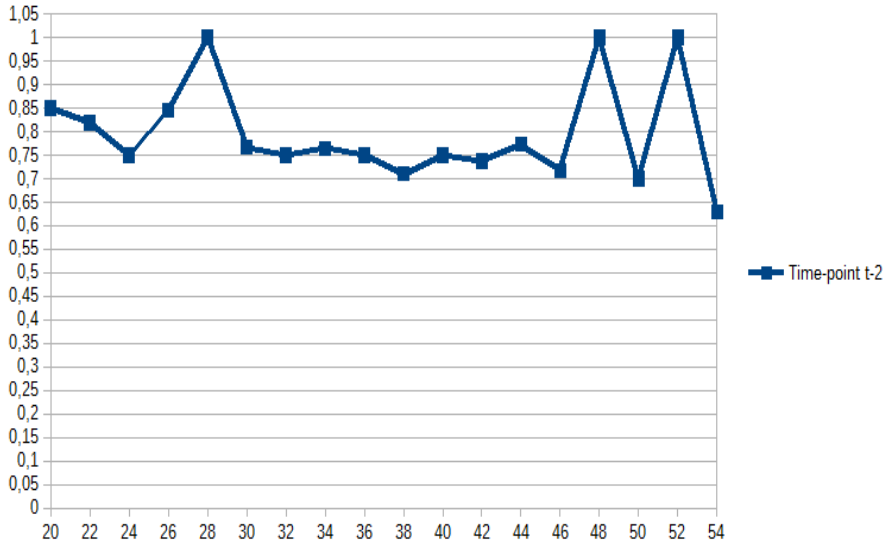


Figure 4.4: Rank-based predictability of time-point t-2 (dataset $D_{t-2}$). X-axis: number of selected examples. Y-axis: predictability. The number of selected examples composing the dataset contains an equal number of top-ranked selected examples from the positive class and bottom-ranked selected examples from the negative class. Therefore, in the case with 20 selected examples, the top-ranked selected positive examples are 10 and the bottom-ranked selected negative examples are 10. The threshold value used is 0.7. At the final subset with 54 examples, the predictability (0.629) is lower than 0.7 and for that reason the rank-based selection approach did not continue to include further examples at the subset.

Figure 4.4 displayed the results of random-based predictability for dataset $D_{t-2}$. The

maximum number of examples with predictability higher than 0.7 was 52, composed by the 26 top positive and 26 bottom negative examples. As far as the dataset coverage was concerned:
**Total Coverage**: 0.65
**Positive Coverage**: 0.70

The positive coverage indicated that almost 3 out of 4 positive examples were predictable. The total coverage was smaller than the positive coverage. However, this was reasonable due to that the maximum number of positive predictable examples was 70% of the total ones. Since the positive examples were less than the negative ones, the percentage of negative examples included in the biggest predictable subset was 61%. This did not necessary indicate that more negative examples were not predictable, but since the maximum number of positive examples had been reached, there was no point in included more negative examples.

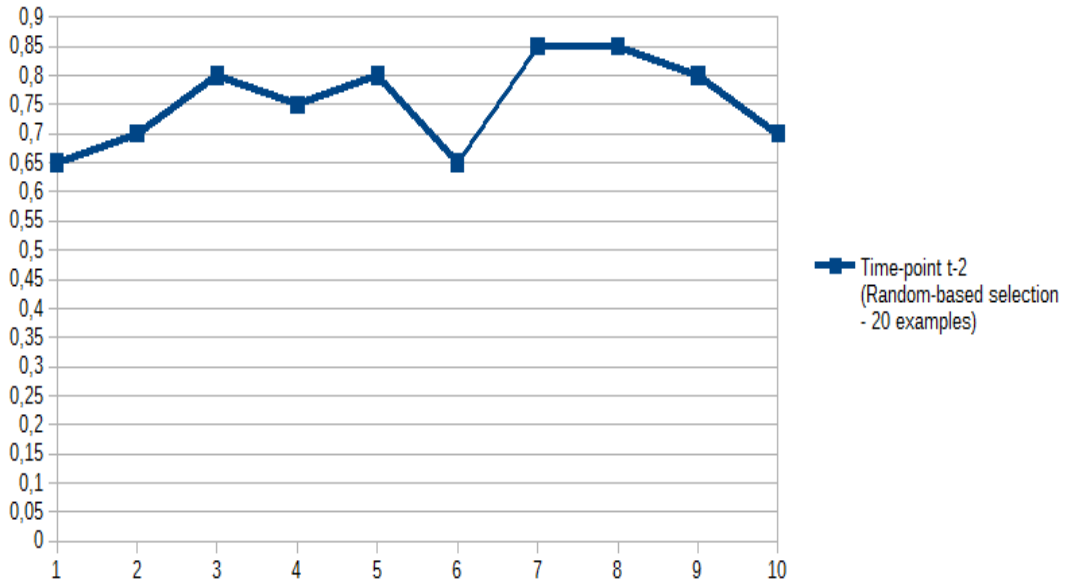


Figure 4.5: Random-based predictability of time-point t-2 (dataset $D_{t-2}$), for 20 selected examples. X-axis: under-sampling repetitions. Y-axis: predictability. The number of selected examples composing the subset contains an equal number of the positive and negative class.

As shown in figure 4.5, an experiment was performed for comparing the predictability between the rank-based and random-based predictability approach for a given dataset. The dataset $D_{t-2}$ was applied to the random-based approach. The random-based approach was applied 10 times with 10 randomly selected examples from the positive and 10 randomly selected examples from the negative class. The reason why 20 examples were selected for the experiments was because as shown at figure 4.2, were highly pure in terms of distribution between the two classes.
The predictability was estimated with the black-box classifier. Therefore, the 10 undersampling repetitions estimated 10 predictability values (fig. 4.5), which were compared with the equivalent one of the rank-based predictability approach for 20 selected examples (fig. 4.4). At the rank-based approach, the predictability for 20 examples was 0.85. From the random-based approach, only 2 repetitions had 0.85 predictability, with the rest being bellow that value and the average predictability being 0.75.
Hence, the rank-based approach produced higher predictability than the average predictability of the random-based approach. Moreover, the highest value estimated by the random-based approach was equal to the rank-based one. The experiments indicated that the subsets of $D_{t-2}$ ranging from size 20 until 2K, where K was the number of positive examples, were significant (section 4.6), and that the spikeness ranking provided a possible interpretation to the results of that chapter.

# Chapter 5

# Detecting Predictable Changes



## 5.1 Problem Definition

Temporal changes between various states present in time-series data, were simply referred to as changes. A state was considered to be a discrete or symbolic characterization of the data distribution at a given time-point. A change was considered predictable if it could be predicted by feature values at time-points previous than the ones at which the change occurred. Therefore, the detection of predictable changes was defined as the detection of temporal changes between states in time-series data, capable of being predicted at previous time-points.

There have existed multiple types of temporal changes that could be present in time-series data. One of these possible types was the spike. The limitations of the approach used for identifying spikes, was that the criterion for separating examples into positives and negatives, was based on the beta diversity hypothesis that a spike occurred according to specific fluctuations in the beta diversity of specific time-series subsets

(section 3.1). Moreover, the beta diversity spike hypothesis (def. 3.3) was specifically adapted to the spike pattern: a-b-a, where a and b were different states of the time-series data. But what about the patterns of change a-b-b or a-b-c? For that reason, a general approach had to be applied for change identification. However, since changes refer to changes in state, the states of a given time-series dataset had to be defined first.

## 5.2 Discretization through clustering

Since a state was considered to be a discrete (symbolic) characterization of the data distribution at a given instance, the various states of a given dataset were identified through discretization. The discretization of the dataset was achieved with clustering. The advantage of clustering as an approach for data discretization, has been that all examples of the dataset could be represented by the cluster label at which they belong to. The cluster representations lead to a discrete form of the dataset. Given three examples, named example 1, example 2 and example 3, and the three corresponding classes, named a,b and c, at which they belonged to, the examples could be characterized as: a,b,c, meaning that they have been discretized. Therefore, each cluster label has become the state that represented all its example-members. As mentioned in section 1.1, metadata were available at the supplementary material of [1], from which the dataset was parsed. An important metadata was the Community State Types (referred to CST for abbreviation), which were based on differences in species composition and their relative abundances. The CST were 5 in total and characterized every time-point of the complete dataset. In other words, the CST constituted the initial discretization of the data that could be used for the analysis.
The Community State Types, as the clustering work of [1] could eliminate the need of any further discretization of the dataset. However, any form of clustering was prone to errors. For that reason, multiple clustering approaches were used with the intent of identifying states which the community state type approach might erroneously did not. The idea of multiple clustering approaches, was to implement a form of ensemble clustering, based on the assumption that: temporal changes in state not identified by one clustering might have been identified by the rest. For that reason, many clustering approaches were used. Since there was no knowledge about the optimal number

of clusters, non-parametric clustering algorithms were preferred. The final clustering approaches applied were:

1. agglomerative hierarchical clustering with the Jensen-Shannon divergence measure and silhouette measure for estimating the optimal number of clusters,
2. agglomerative hierarchical clustering with the Bray-Curtis dissimilarity measure and silhouette measure for estimating the optimal number of clusters,
3. agglomerative hierarchical clustering with the euclidean distance measure and silhouette measure for estimating the optimal number of clusters,
4. agglodip clustering with the Jensen-Shannon divergence measure,
5. agglodip clustering with the Bray-Curtis dissimilarity measure,
6. agglodip clustering with the euclidean distance measure.

While the algorithms were two in total, the different proximity measures constructed different proximity matrices and thus produced different results. Therefore, 6 clustering approaches were applied in total.
The next step was the matching of the clustering labels. Every clustering approach produced a number of clusters, each with its own labels. In order for a comparison between the results of the various clustering approaches to be possible, there must have been knowledge regarding the correspondence between their labels. This was achieved with the use of a simple algorithm for matching the clustering labels. The CST labels were used as the reference ones, since they were initially used in the analysis at [1] where the dataset was taken from. Therefore all clustering approaches matched their labels with the Community State Type ones. The Community State Type clustering contained categorical values : I,II,III,IV-A,IV-B. Since the clustering labels contained nominal numerical values, the first step was the conversion of the Community State Type values to nominal ones:

1. The Community State Type labels were converted to: 1,2,3,4,5, where 1 represented I, 2 represented II, 3 represented III, 4 represented IV-A and 5 represented IV-B.
2. The next steps were similar for all different approaches. Given the labels assigned to the entire dataset by a clustering approach, a cost matrix was constructed

between the community state type clustering and the clusters of the approach. The matrix size was nxm, where n was the total number of CST clusters, and m the total number of clusters for the approach. The $cost_{ij}$ , where i referred to a cluster from the approach and j referred to a cluster from CST, was estimated as the number of times a data point/example was assigned to cluster i in the approach and cluster j in CST. The pair i-j with the highest score was matched, meaning that cluster i was considered to match with cluster j. Therefore the label of i was updated: $label_i = label_j$.

3. The pair i-j was removed from the matching algorithm. Furthermore, since cluster i was matched, all data points belonging to cluster i were removed from both CST and the approach. Step number 2 continued until all clustering labels were matched.

4. There existed cases where the number of clusters for both approaches was not equal. In those cases, the matching was valid for only k clusters, where k is the number of clusters of the approach with the least clusters between the two. The unmatched clusters retained their original labels.

A number of techniques were applied for measuring the clustering solutions. The first technique was the normalized mutual information (also referred to as NMI). The NMI was applied between the final clusters of each applied clustering approach, and the CST clusters. The higher the NMI values were, the closer to the CST ones were considered to be. While this was not a pure indicator for clustering quality, it added another parameter to the clustering evaluation. The information that many clustering approaches provided similar clusters, strengthened the hypothesis that the clustering solutions were good.
Finally, a similarity matrix was between all clustering approaches was constructed, for testing the aforementioned hypothesis. The similarity matrix was constructed with the use of the Needleman-Wunsch algorithm [32], which is frequently used in Bioinformatics analysis for aligning amino-acid and nucleotide sequences.

## 5.3 Patterns of Temporal Changes

After the matching of the various clusterings, which represented states of the dataset, the next step was the detection of changes in state. Temporal changes in state were considered as patterns. Hence, a spike was one possible temporal pattern. The exhaustive search for every possible pattern on the dataset was time-demanding and thus not preferred. For that reason, the patterns searched in the analysis were predefined.

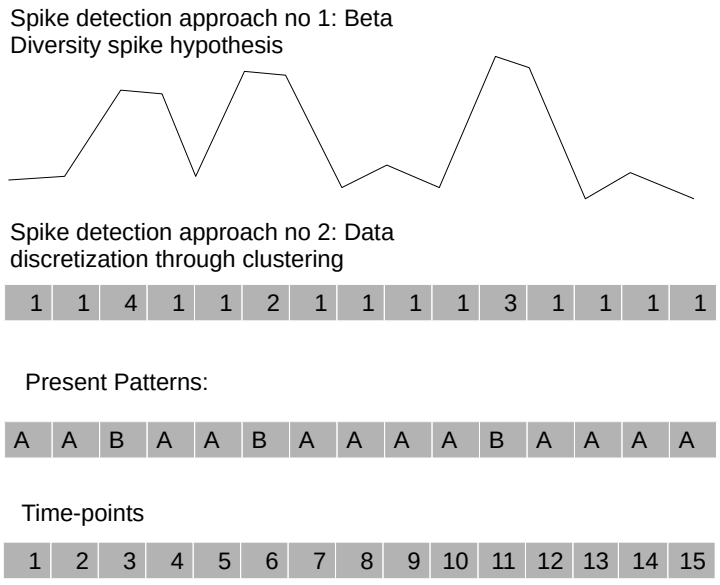


Figure 5.1: Comparison of the two different ways for detecting Spikes, given a time-series with 15 time-points. The diagram drawn at the top represents the beta diversity spike hypothesis, where each time-point is represented by a beta diversity value. The $1^{st}$ number sequence bellow the diagram represents the discretization approach, where each time-point is represented by its cluster or state. The number sequence titled as Present Patterns, represents the patterns that can be detected on that time-series, where symbol A represents similarity to the previous state and B difference from the previous state. The first number sequence from the bottom, titled as Time-points, represents the time-points of the time-series

Furthermore, the patterns were not dependent on specific states, but on the differences between consecutive states through time. An example has been the pattern a-b-c, given three consecutive time-points $t_1$,$t_2$ and $t_3$. This pattern, has not indicated that a state named a was succeeded by a state named b, which was succeeded by a state named c. Contrary to that its actual interpretation has been that: (state of $t_1$) $\neq$ (state of $t_2$) $\neq$ (state of $t_3$). Similarly, a-b-a could be interpreted as: (state of $t_1$) $\neq$ (state of $t_2$), (state of $t_2$) $\neq$ (state of $t_3$) and (state of $t_1$) = (state of $t_3$).

The identification of a given pattern at the dataset was performed for each clustering approach separately. Figure 5.1, presents a visual contrast between the discretization through clustering and the beta diversity approach (def. 3.3) for detecting spikes. Moreover, it presents the correspondence of states or beta diversity values, to the patterns of temporal changes present in a time-series. In comparing the two approaches, the discretization approach was less prone to errors than the first one, since the errors could only be accounted to bad clustering quality, which could be corrected by better clustering or a clustering ensemble like in section 5.2. The threshold values of the first approach were more prone to errors, since they were dictated by the median and standard deviation of the local region of the time-series. The local region of the time-series, whose size was dictated by a sliding window approach (section 3.3), was considered to take into account the local fluctuations in diversity and was thus preferred over the complete time-series. The problem with the local region was the lack of a way of estimating an optimal length that could take into account the local changes. Therefore, the selection of an optimal window size was done by a visual observation of the time-series. For that reason, a bad window size selection could lead to a bad labeling of the datasets, which could explain the non-optimal results for spike prediction (table 3.1, section 3.4).

Therefore, the detection of patterns of temporal changes could be described as: the process of matching a symbolic representation of time-points to patterns of temporal changes between those representations, and detecting the pattern of interest. The following algorithm describes the steps taken for pattern detection:

1. The clustering labels were assigned in time-series, meaning that the clustering label of a given example was assigned to the time-series at the time-point at which the example belonged to.

2. For each time-series:

(a) a sliding window was applied for scanning the time-series, with window size equal to the length of the pattern. For example, pattern a-b-a had length 3 and therefore all feature vectors of 3 consecutive time-points were scanned.

(b) Each subset of time-points fitting to the window size, was checked on whether it matches the pattern or not. The matching was performed with the following way: the first cluster label appearing at the sequence was assigned to the first symbol of the pattern and so on. Afterwards, all cluster labels were matched to the symbols they were assigned to. For example, given the pattern a-b-a and the states 2-3-2, 2 would be assigned to a and 3 would be assigned to b. Therefore, 2-3-2 would get matched to a-b-a. In the case of 2-3-3, the subset would get matched to a-b-b, which is different than a-b-a.

(c) In case that a subset of time-points did match the pattern, then the middle time-point was considered as the pattern positive one, and the label 0 was assigned to the feature vectors of the three previous time-points (t-1,t-2,t-3), meaning that their future time-point (1,2 or 3 time-points later respectively) was associated with the pattern. In case that it did not match the pattern, then the same procedure happened with the difference that the label 1 was assigned to the feature vectors of time-points t-1,t-2 and t-2, meaning that their future time-point was not associated with the pattern.

(d) Furthermore, the examples/data points corresponding to time-points t-1,t-2 and t-3 were stored for constructing the dataset.

(e) The spikeness of each pattern positive/negative example was estimated. This time, the absolute differences approach was preferred over the relative differences approach for investigating possible changes in the analysis results. The Jensen-Shannon (eq. 2.51) and Bray-Curtis (eq. 2.61) measures were both used as possible measures for estimating spikeness (eq. 4.1). The spikeness value was also stored for the construction of the dataset.

3. After the time-series of all time-points were scanned, three datasets were constructed. The stored examples corresponding to time-points t-1, t-2 and t-3 constructed three separate datasets, $X_{t-1}$, $X_{t-2}$ and $X_{t-3}$, with each one corresponding to the temporal proximity between the examples and the pattern posi-

tive/negative ones. Examples having the same position/row in all three datasets, had the same class label, since they belonged to time-points t-1,t-2 and t-3 before the same temporal change.

4. Since the spikeness for all pattern positive/negative time-points was estimated, a vector with all spikeness values was constructed. This vector had equal size to the number of examples and would be used for further analysis.

Since this pattern identification algorithm was applied to the dataset for each clustering solution separately, there existed a different label vector for each solution. Therefore, the next step was the ensemble of the clustering solutions. The ensemble would be used to define the final label vector y of the dataset. The ensemble solution was similar to the one used in ensemble classification [31], where a majority vote is used for classifying a testing example.
For i=1,..,n, where n is the number of examples in the dataset, and given a voting threshold $voting_{thr}$

1. $positive_i$ was estimated, where $positive_i$ refers to the percentage of 0 (positive class) values assigned to the $i^{th}$ example for all label vectors.

2. if $positive_i \geq voting_{thr}$, then: $y_i = 0$,
   else: $y_i = 1$.

The output result of the methodology was the label vector y that contained the final label value. Similar to the methodology used for predicting spikes in section 3.3, the overlap removal step was applied to the data. The difference from the previous approach was that according to the methodology of detecting predictable changes, the time-points t-1,t-2 and t-3 used for prediction were gathered and assembled datasets before the removal of overlapping examples. Furthermore, a time-constraint of 2 time-points was applied for removing examples being closer to chronologically older ones from the same time-series and the same class on the dataset. The reason for a less strict constraint was that the 6 time-points used in the spike prediction analysis (chapter 3) produced a very small dataset with 79 examples in total (section 3.4) and thus a larger dataset was desired. While the removal of overlapping examples did not provide a dataset as pure as the spike prediction one, a temporal-proximity of 2 time-points ensured that all time-points of the same class would be at least 8 days distant from each other.

Hence, the detection of temporal changes lead to the following output results used for further analysis:
$(X_{t-3}, y) \in D_{t-3}$, $(X_{t-2}, y) \in D_{t-2}$ and $(X_{t-3}, y) \in D_{t-1}$ and spikeness.

## 5.4 Predictability of Temporal Changes

Given the definitions of predictability (section 4.5) and temporal changes (section 5.3), predictability of temporal changes was defined as: the generalization performance of an optimal classification system built using a given dataset and tested on a given evaluation measure.
Since an optimal classification system could only be approximately constructed (section 4.5), it was assumed that the results of a classification system were optimal. A dataset was considered predictable if its predictability exceeded a desired threshold (section 4.5). Therefore, the analysis focused on detecting predictable subsets of the dataset, since the experimentation results of section 4.7 indicated the existence of such subsets. The most important reason for finding predictable subsets, was that predictable subsets that made up a significant proportion of the dataset, provided quantitative information regarding its predictability (section 4.7).
As far as methodology was concerned, the classification system applied was the black-box classifier (section 4.4). The evaluation measure used was leave-one-out CV (eq. 2.4). The detection of predictable subsets was achieved with the rank-based predictability approach. For the rank-based selection (section 4.3), the ranking measure used was spikeness (eq. 4.1).
The datasets used for analysis were: $(X_{t-3}, y) \in D_{t-3}$, $(X_{t-2}, y) \in D_{t-2}$ and $(X_{t-1}, y) \in D_{t-1}$, as well as the spikeness vector.
The steps taken for constructing the dataset, from the initial detection of cluster-states to the application of the rank-based predictability approach, were similar to the ones described for the prediction of spikes in section 3.3. The steps taken for discretization of the dataset and the detection of changes in state were equivalent to the subset selection and time-point selection steps. The rank-based predictability step was equivalent to the classification step. The overlap removal step was applied with the less strict time-constraint of 2 time-points. Finally, the feature selection step was applied for features whose total value in a given dataset was zero. Therefore,

despite the differences in the approaches used, the methodology for the detection of predictable changes at the given longitudinal dataset could be considered as an automated way of predicting spikes and similar changes, contrary to the spike prediction approach where changes were detected based on a hypothesis (section 3.1).

## 5.5 Detailed method description

**Algorithm 5.1** Detecting Predictable Changes

**Input**: $X_{initial}$: Initial Dataset of size NxM.

$Ts_X$: longitudinal time-series correspondent to the dataset with L time-series, Y time-points per time-series and W features per time-point where: LxYxW = NxM.

$cluster\_approaches$: matrix with K clustering approaches, $K \in N$.

$voting_{thr}$ = cluster voting threshold.

$Input\_Pattern$: Pattern whose predictability is to be estimated.

$dissimilarity\_measure$: measure that will be used for estimating spikeness.

$Black\_Box$: black box classifier.

$generalization_{thr}$: generalization threshold of predictability.

$lower\_bound$ = lower bounds on rank-based example selection.

$upper\_bound$ = upper bounds on rank-based example selection.

**Output**: $Predictability$, where Predictability is a vector of size $upper\_bound - lower\_bound$, for all dataset subsets selected in the process.

$Total\_Coverage$: Percentage of total predictable examples of the dataset.

$Positive\_Coverage$: Percentage of positive predictable examples of the dataset.

Discretization through clustering phase

**for** $i \Leftarrow 1$; $i \leq K$; $i \Leftarrow i + 1$ **do**

  $cluster\_labels_i \Leftarrow cluster\_approach_i(X_{initial})$

  $discrete\_TS_i \Leftarrow$ passing the labels of $cluster\_labels_i$ to $Ts_X$

  $class\_label_i \Leftarrow \emptyset$

**end for**

$windowsize \Leftarrow size(Input\_Pattern)$

$X_{t-1}, X_{t-2}, X_{t-3} \leftarrow \emptyset$

Patterns of temporal changes detection phase

**for** Time-series in $discrete\_TS_i$ **do**

  **for** $time - point \Leftarrow 3; time - point \leq size(Time - series) - windowsize; time - point \Leftarrow time - point + windowsize$; **do**

    $pattern\_tp \leftarrow time - point + windowsize/2$, is the time-point within the sliding window frame where the temporal change is expected to happen.

**for** $i \leftarrow 1; i \leq K; i \leftarrow i + 1$ **do**

  $subset \leftarrow discrete\_TS_i(time - point) \cup discrete\_TS_i(time - point + 1), .., \cup discrete\_TS_i(time - point + windowsize)$

  $matching \Leftarrow$ match subset with pattern.

  **if** matching is correct **then**

    $class\_label_i \cup 0$, 0 indicates pattern detection

  **else**

    $class\_label_i \cup 1$, 1 indicates non-pattern detection

  **end if**

**end for**

$X_{t-1} \cup X(pattern\_tp - 1)$

$X_{t-2} \cup X(pattern\_tp - 2)$

$X_{t-3} \cup X(pattern\_tp - 3)$

**if** pattern is associated with a single change in state **then**

  $spikeness \cup dissimilarity\_measure(pattern\_tp - 1, pattern\_tp)$

**else**

  $spike_1 \leftarrow dissimilarity\_measure(pattern\_tp - 1, pattern\_tp)$

  $spike_2 \leftarrow dissimilarity\_measure(pattern\_tp, pattern\_tp + 1)$

  $spikeness \cup min(spike_1, spike_2)$

**end if**

**end for**

**end for**

$class\_labels \Leftarrow \emptyset$

$y \Leftarrow \emptyset$

**for** $i \leftarrow 1; i \leq K; i \leftarrow i + 1$ **do**

  $class\_labels \cup class\_label_i$

**end for**

**for** $i \leftarrow example \in class\_labels$ **do**
  $positive_i \leftarrow \frac{\#[class\_labels(i)=0]}{K}$
  **if** $positive_i \geq voting_{thr}$ **then**
    $y \cup 0$
  **else**
    $y \cup 1$
  **end if**
**end for**
Predictability of temporal changes phase
$D_{t-1} \leftarrow (X_{t-1}, y)$, $D_{t-2} \leftarrow (X_{t-2}, y)$ and $D_{t-3} \leftarrow (X_{t-3}, y)$
$D_{sel} \leftarrow$ selection between $D_{t-1}, D_{t-2} \quad and \quad D_{t-3}$
partition $D_{sel}$ into $C_1 \leftarrow$ positive class examples, and $C_2 \leftarrow$ negative class examples.
$C_1 \leftarrow rank(spikeness, C_1)$, $C_2 \leftarrow rank(spikeness, C_2)$ , $C_1$ and $C_2$ are ranked on descending order
$Predictability \leftarrow \emptyset$
**for** $k \leftarrow lower\_bound; k \leq upper\_bound; k \leftarrow k+1$ **do**
  $Top\_Positive \leftarrow$ the first k examples from $C_1$
  $Bottom\_Negative \leftarrow$ the last k examples from $C_2$
  $D_k \leftarrow Top\_Positive \cup Bottom\_Negative$
  $predictability_k \leftarrow Black\_Box(D_k)$
  **if** $predictability_k \geq generalization_{thr}$ **then**
    Break - the detection of predictable subset ceases.
  **else**
    $Predictability \cup predictability_k$
  **end if**
**end for**
$Positive\_Coverage \leftarrow \frac{k}{size(C_1)}$
$Total\_Coverage \leftarrow \frac{k}{size(D_sel)}$

# Chapter 6

# Experimental Results



## 6.1 Introduction

The methodological steps described in the algorithm 5.1 of chapter 5, were followed for experimentation and analysis. Each section of this chapter corresponds to the equivalent section of chapter 5, where various plots are used for visualizing the results.

## 6.2 Discretization though clustering

As mentioned in section 5.2, six different clustering approaches were applied to the complete dataset, which were named in short as: Jensen-Shannon Agglodip, Bray-Curtis Agglodip, Euclidean Agglodip, Jensen-Shannon Agglomerative, Bray-Curtis Agglomerative and Euclidean Agglomerative.
The criterion for estimating the optimal number of clusters at the agglomerative hierarchical approaches was the mean silhouette value (section 2.3) for all data points

of the dataset. The criterion for estimating the optimal number of clusters at the Agglodip approaches was the dip-dist criterion (section 2.3), which in order to be utilized optimally was applied to all possible pairs of data points. Therefore, in the Agglodip approaches all possible pairs of data points were tested with the dip-dist criterion from the beginning.

| **Time-series 1:** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | **20** | **21** | **22** | **23** | **24** | **25** | **26** | **27** | **28** | **29** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bray-Curtis Agglodip | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 4 | 3 | 5 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 4 | 3 | 3 | 3 | 3 | 3 | 3 |
| Jensen-Shannon Agglodip | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 1 | 3 | 3 | 3 | 3 | 3 | 3 |
| Euclidean Agglodip | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 5 | 3 | 5 | 5 | 5 | 5 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 5 | 5 | 3 | 3 | 3 | 3 | 3 | 3 |
| Bray-Curtis Agglomerative | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 4 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 4 | 4 | 3 | 3 | 3 | 3 | 3 | 3 |
| Jensen-Shannon Agglomerative | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 4 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 4 | 4 | 3 | 3 | 3 | 3 | 3 | 3 |
| Euclidean Agglomerative | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 5 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 5 | 5 | 3 | 3 | 3 | 3 | 3 | 3 |
| Community State Type | 4 | 3 | 3 | 3 | 3 | 3 | 3 | 4 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 4 | 4 | 3 | 3 | 3 | 3 | 3 | 3 |

| **Time-series 10:** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | **20** | **21** | **22** | **23** | **24** | **25** | **26** | **27** | **28** | **29** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bray-Curtis Agglodip | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 5 | 5 | 5 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 5 | 5 | 5 | 3 | 5 |
| Jensen-Shannon Agglodip | 1 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 | 1 | 1 | 3 | 1 | 5 | 5 | 5 | 1 | 3 | 3 | 3 | 1 | 3 | 3 | 1 | 5 | 5 | 1 | 1 | 5 |
| Euclidean Agglodip | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 5 | 5 | 5 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 5 | 3 | 5 | 5 | 5 |
| Bray-Curtis Agglomerative | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 5 | 5 | 5 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 5 | 3 | 3 | 4 | 3 |
| Jensen-Shannon Agglomerative | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| Euclidean Agglomerative | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 5 | 3 | 5 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 5 | 3 |
| Community State Type | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 4 | 3 |

Figure 6.1: Cluster labels assigned to time-points of two time-series for different clustering approaches. Top panel: time-series 1. Bottom panel: time-series 10. X-axis (all panels): labels per clustering approach. Y-axis (all panels): time-points. The names of the clustering approaches are on the first column of each panel. The labels of each approach are presented after having been matched with the equivalent ones of Community State Type (section 5.2).

The cluster labels were matched with the equivalent ones from Community State Type (CST), since CST was used as the reference clustering (section 5.2). The CST labels in nominal form were: 1,2,3,4 and 5. The clustering results for each approach, after the matching with CST, were:

1. Jensen-Shannon Agglodip:
   4 clusters with the labels 1,2,3 and 5.

2. Bray-Curtis Agglodip:
   5 clusters with the labels 1,2,3,4 and 5.

3. Euclidean Agglodip:
   4 clusters with the labels 1,2,3 and 5.

4. Jensen-Shannon Agglomerative:
   5 clusters with the labels 1,2,3,4 and 5.

5. Bray-Curtis Agglomerative:
   5 clusters with the labels 1,2,3,4 and 5.

6. Euclidean Agglomerative:
   4 clusters with the labels 1,2,3 and 5.

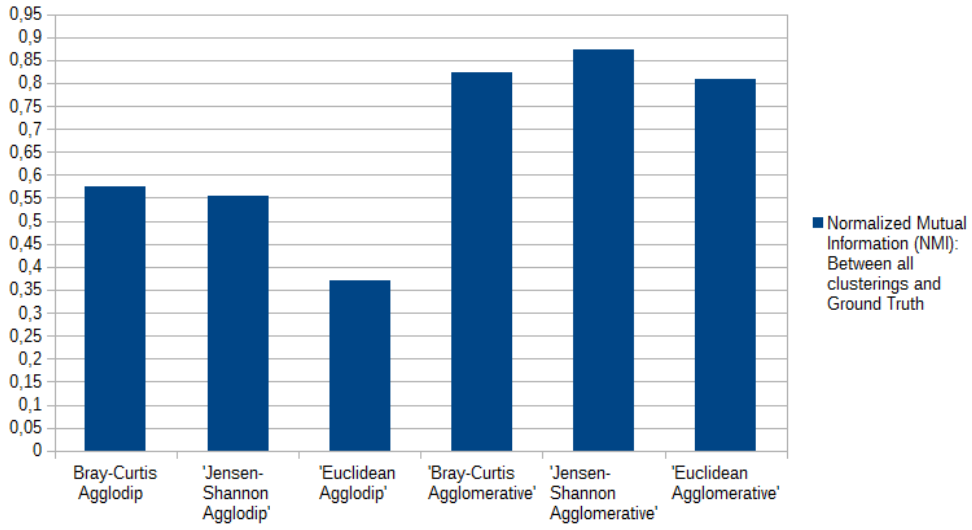


Figure 6.2: Normalized Mutual Information (NMI) between a number of clustering approaches and a reference clustering. X-axis: NMI values. Y-axis: names of the clustering approaches.

Figure 6.1 presented examples of cluster label from all clustering approaches and after having been matched with CST, assigned to their equivalent time-points for two indicative time-series. The reference clustering for the NMI at figure 6.2, was the CST. The highest NMI in figure 6.2 was given by the Jensen-Shannon agglomerative approach, which was reasonable given that the reference clustering was obtained with the same approach. The reason why both clusterings were not completely identical, with an NMI of 1, was that the CST clustering was performed on the rRNA gene sequence data. The analysis at the gene sequence data on [1], produced the relative abundances used to construct the dataset (section 1.1) of this research. Therefore in the transition between rRNA gene sequences and relative abundances there was information loss leading to the deviation between the two clustering approaches.

| **Similarity Matrix** | Bray-Curtis Agglodip | Jensen - Shannon Agglodip | Euclidean Agglodip | Bray-Curtis Agglomerative | Jensen-Shannon Agglomerative | Euclidean Agglomerative |
|---|---|---|---|---|---|---|
| Bray-Curtis Agglodip | 29,28125 | 24,96875 | 19,875 | 24,53125 | 23,15625 | 24,15625 |
| Jensen-Shannon Agglodip | 24,96875 | 29,28125 | 19,84375 | 24,375 | 22,75 | 24,46875 |
| Euclidean Agglodip | 19,875 | 19,84375 | 29,28125 | 19,9375 | 18,5 | 21,125 |
| Euclidean Agglomerative | 24,53125 | 24,375 | 19,9375 | 29,28125 | 27,46875 | 26,8125 |
| Euclidean Agglomerative | 23,15625 | 22,75 | 18,5 | 27,46875 | 29,28125 | 26,03125 |
| Euclidean Agglomerative | 24,15625 | 24,46875 | 21,125 | 26,8125 | 26,03125 | 29,28125 |

Figure 6.3: Similarity matrix between a number of clustering approaches. X-axis and y-axis correspond to pairs of clustering approaches. Given a pair i-j, the value corresponding to i-j is the global alignment score between clustering approaches i and j

The alignment algorithm used in figure 6.3 was the Needleman-Wunsch algorithm. Since the solutions did not correspond to amino-acids or nucleotides, a simple identical matrix was given as input. The identical matrix was used for assigning a high

score, considered as alignment, between pairs of the same solution and a lower score, considered as misalignment, between pairs of different solutions, which would be the same for all possible pairs. Furthermore, since the Needleman-Wunsch algorithm has taken amino-acids or nucleotides as input, each cluster label was represented by a unique amino-acid symbol. After the conversion of cluster labels to amino-acids, the alignment was applied.

The solutions were aligned for each time-series separately, and the final alignment was estimated by averaging the similarity matrices of all time-series. The alignment was time-series wise and not for the complete dataset, since the changes of states of time-points inside each time-series had to be taken into account for the similarity estimation.

The solution with the lowest alignment scores (fig. 6.3) was Euclidean Agglodip. For that reason, this solution was removed from further analysis. The alignment scores for the rest of the solutions (fig. 6.3) were highly identical, which strengthened the hypothesis of a good clustering quality. Hence, the five remaining solutions were used for further analysis since their scores indicated clustering similarity.

## 6.3 Patterns of Temporal Changes

A number of patterns of temporal changes were tested on whether they could be detected for all time-series and for all clustering solutions used. The patterns tested were: A-B-A, A-B-B, A-B-A-A and A-A-B-A-A.

Figure 6.4 displayed the results on pattern detection for pattern A-B-A at two indicative time-series. An indicative example of pattern A-B-A was the subset 7-8-9 of time-series 1 (fig. 6.4, top panel). The clustering labels of this subset for the aforementioned approaches were: 3-4-3, 3-5-3 or 3-1-3 (fig. 6.1). The classification labels of those time-points for the aforementioned approaches in figure 6.4 were: 1-0-1, indicating that the pattern was detected correctly. Similarly, all pattern positive time-points for all time-series were detected correctly.

***Pattern A-B-A:***

| **Time-series 1:** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | **20** | **21** | **22** | **23** | **24** | **25** | **26** | **27** | **28** | **29** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bray-Curtis Agglodip | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| Jensen-Shannon Agglodip | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Bray-Curtis Agglomerative | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Jensen-Shannon Agglomerative | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Euclidean Agglomerative | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Voted Labels | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| **Time-series 10:** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | **20** | **21** | **22** | **23** | **24** | **25** | **26** | **27** | **28** | **29** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bray-Curtis Agglodip | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 |
| Jensen-Shannon Agglodip | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Bray-Curtis Agglomerative | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |
| Jensen-Shannon Agglomerative | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Euclidean Agglomerative | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 |
| Voted Labels | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 |

Figure 6.4: Class label vectors assigned to time-points of two time-series for different clustering approaches. Top panel: time-series 1. Bottom panel: time-series 10. X-axis (all panels): class labels per clustering approach - dictated by pattern A-B-A. Y-axis (all panels): time-points. The names of the clustering approaches are on the first column of each panel. The final row in all panels corresponds to the labeling that has been derived from voting. The voting threshold is 40%. The labels for each time-point are binary values representing a pattern positive (0) or negative (1) time-point, where the pattern is the A-B-A.

Pattern A-B-B

| **Time-series 1:** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | **20** | **21** | **22** | **23** | **24** | **25** | **26** | **27** | **28** | **29** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bray-Curtis Agglodip | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| Jensen-Shannon Agglodip | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| Bray-Curtis Agglomerative | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| Jensen-Shannon Agglomerative | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| Euclidean Agglomerative | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| Voted Labels | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |

| **Time-series 10:** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | **20** | **21** | **22** | **23** | **24** | **25** | **26** | **27** | **28** | **29** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bray-Curtis Agglodip | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 |
| Jensen-Shannon Agglodip | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 |
| Bray-Curtis Agglomerative | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| Jensen-Shannon Agglomerative | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Euclidean Agglomerative | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Voted Labels | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 |

Figure 6.5: Class label vectors assigned to time-points of two time-series for different clustering approaches. Top panel: time-series 1. Bottom panel: time-series 10. X-axis (all panels): class labels per clustering approach - dictated by pattern A-B-B. Y-axis (all panels): time-points. The names of the clustering approaches are on the first column of each panel. The final row in all panels corresponds to the labeling that has been derived from voting. The voting threshold used is 40%. The labels for each time-point are binary values representing a pattern positive (0) or negative (1) time-point, where the pattern is the A-B-B.

Figure 6.5 displayed the results on pattern detection for pattern A-B-B at the same time-series as figure 6.4. An indicative example of pattern A-B-B was the subset 8-9-10 of time-series 1, for all approaches except Bray-Curtis Agglodip (fig. 6.5, top panel, second row). The clustering labels of this subset for the aforementioned approaches were: 1-3-3, 4-3-3 or 5-3-3. The classification labels of those time-points for

the aforementioned approaches in figure 6.5 were: 1-0-1, indicating that the pattern was detected correctly. Similarly, all pattern positive time-points for all time-series were detected correctly.

The same experiments were performed for all time-series and for all patterns. The voting threshold used for voting in all clustering approaches whether a pattern detection was true or not, was 0.4. The classification labels of all time-series were merged and constituted the classification vector label y. Finally, the spikeness measure (eq. 4.1) for all examples was estimated with the absolute differences approach using both the Jensen-Shannon (eq. 2.51) and Bray-Curtis (eq. 2.61) measures (figs. 6.6 and 6.7).

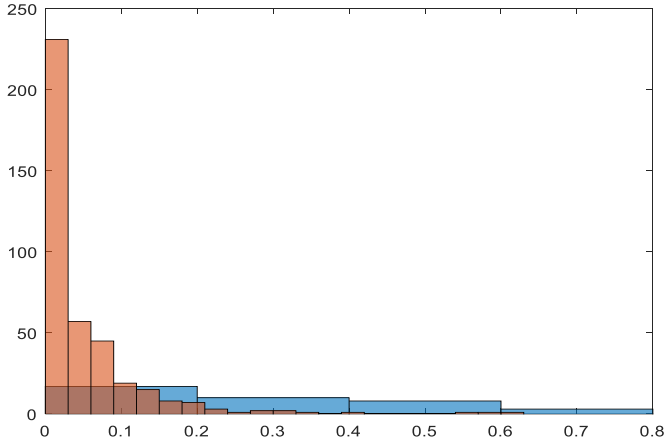


Figure 6.6: Histogram of spikeness values for pattern A-B-A, estimated with the Jensen-Shannon divergence measure. X-axis: number of examples with a spikeness value in a given range. Y-axis: spikeness values. The blue color corresponds to the positive class. The red color corresponds to the negative class. There is a small portion of positive examples less than 20, overlapping with negative ones in the range 0-0.2. There is another portion of positive examples less than 20, overlapping with negative ones in the range 0.2-0.4. Further few negative examples overlap with positives in the range 0.55-0.65. Hence, there is impurity between the two classes but the portion of overlapping examples is small.

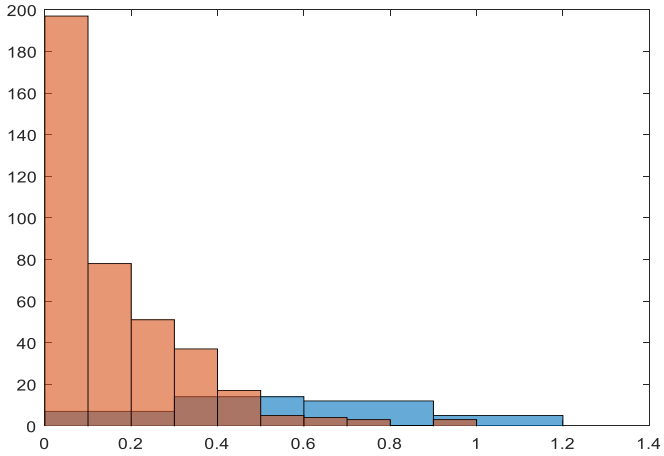


Figure 6.7: Histogram of spikeness values for pattern A-B-A, estimated with the Bray-Curtis dissimilarity measure. X-axis: number of examples with a spikeness value in a given range. Y-axis: spikeness values. The blue color corresponds to the positive class. The red color corresponds to the negative class. There is a small portion of positive examples less than 20, overlapping with negative ones in the range 0-0.3. There is another portion of positive examples less than 20, overlapping with negative ones in the range 0.3-0.6. Further few negative examples overlap with positives in the range 0.6-0.68 and 1. Hence, there is impurity between the two classes but the portion of overlapping examples is small.

## 6.4 Predictability of Temporal Changes

The detection of temporal changes resulted in 3 datasets, $(X_{t-3}, y) \in D_{t-3}$, $(X_{t-2}, y) \in D_{t-2}$ and $(X_{t-1}, y) \in D_{t-1}$ (section 5.4), and two spikeness vectors (one for the Bray-Curtis and one for the Jensen-Shannon measure), for each pattern separately leading to 12 datasets in total used for experimentation. After the removal of overlapping examples and selection of non-zero features, the number of examples and features remaining for each dataset was:

**Pattern: A-B-A**

$X_{t-3}$ : 433 examples, 268 features.

$X_{t-2}$ : 433 examples, 251 features.

$X_{t-1}$ : 433 examples, 272 features.

positive examples: 38

y : 433 labels, spikeness : 433 values.

**Pattern**: **A-B-B**

$X_{t-3}$ : 455 examples, 267 features.

$X_{t-2}$ : 455 examples, 266 features.

$X_{t-1}$ : 455 examples, 272 features.

positive examples: 64

y : 455 labels, spikeness : 455 values.

**Pattern**: **A-A-B-A**

$X_{t-3}$ : 421 examples, 261 features.

$X_{t-2}$ : 421 examples, 256 features.

$X_{t-1}$ : 421 examples, 258 features.

positive examples: 32

y : 421 labels, spikeness : 421 values.

**Pattern**: **A-A-B-A-A**

$X_{t-3}$ : 400 examples, 262 features.

$X_{t-2}$ : 400 examples, 257 features.

$X_{t-1}$ : 400 examples, 258 features.

positive examples: 22

y : 400 labels, spikeness : 400 values.

The estimation of spikeness for pattern A-B-B was different than the rest 3 patterns. Pattern A-B-B implied that there existed a single change in state contrary to the double change found in spikes. For that reason, spikeness was estimated as: $diversity_t$, where $diversity_t$ was either Jensen-Shannon or Bray-Curtis dissimilarity between t and t-1, and time-points t-1 and t corresponded to A and the first B from A-B-B. The rest of the patterns were estimated with the absolute differences approach as mentioned in section 4.2.

The classification models (section 2.2) used to construct the Black-Box classifier were the ones used in section 4.7. Three experiments were conducted, which tested the predictability of the 4 patterns for the three datasets. At the first experiment, named balanced predictability, rank-based predictability was applied to all datasets for all patterns with spikeness estimated by both Jensen-Shannon and Bray-Curtis measures. The subsets selected contained initially 10 top positive and 10 bottom negative examples and gradually increased in size until they contained K top positive and K

bottom negative examples, were K = number of positive examples in the dataset.
At the second experiment, named imbalanced predictability, the above procedure was repeated, with the difference that the initial subsets selected contained K top positive examples and K+1 bottom negative examples. The negative examples gradually increased in size, 1 new bottom ranked example added at a time, until predictability was lower than 0.7 or the total number of negative examples had been reached. Since, the class distribution was imbalanced the problem was corrected in LOT CV (section 2.2) with the following way: for each testing example, there existed 11 different under-samplings of the training data that produced balanced training sets. Each under-sampled training set classified the testing example and produced a label-output. The majority class label from all 11 different outputs was the one assigned to the testing example.
At the final experiment conducted, random-based predictability was applied to a dataset for k positive and negative examples, for multiple repetitions, and the average predictability was compared with the rank-based predictability.

### 6.4.1 Balanced-Predictability

Initially, the dataset $D_{t-2}$ was selected for testing the predictability of examples representing the feature vector at time-point t-2 for all patterns. Moreover, spikeness was measured with the Jensen-Shannon divergence index.
Figures 6.8, 6.9, 6.10 and 6.11 indicated the rank-based predictability of the dataset containing examples present 2 time-points before the temporal change occurrence, for the 4 patterns and with Jensen-Shannon as a measure for estimating spikeness. For all patterns and for all tested rankings, the predictability was higher than the 0.7 threshold value which was also used at the predictability experiments of section 4.7. Moreover, the highest predictability found in all patterns was 1, indicating a zero generalization error. Furthermore, in all cases the predictability of a pattern with top K positive and bottom K negative was $>= 0.7$. Therefore, the results indicated that the selection of positive examples and a number of bottom ranked examples equal to the positive ones for training, created subsets which predicted the tested patterns.

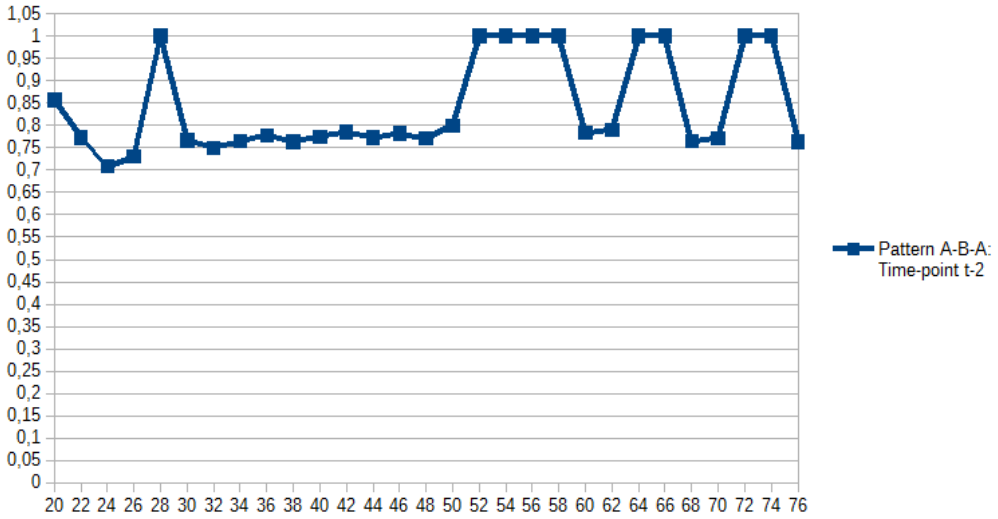


Figure 6.8: Rank-based predictability for pattern A-B-A at time-point t-2. X-axis: number of selected examples. Y-axis: predictability. Spikeness was estimated with the Jensen-Shannon divergence measure. The number of selected examples composing each subset contains an equal number of top-ranked selected examples from the positive class and bottom-ranked selected examples from the negative class. Therefore, in the subset with 20 selected examples, the top-ranked selected positive examples are 10 and the bottom-ranked selected negative examples are 10. The threshold value used is 0.7.

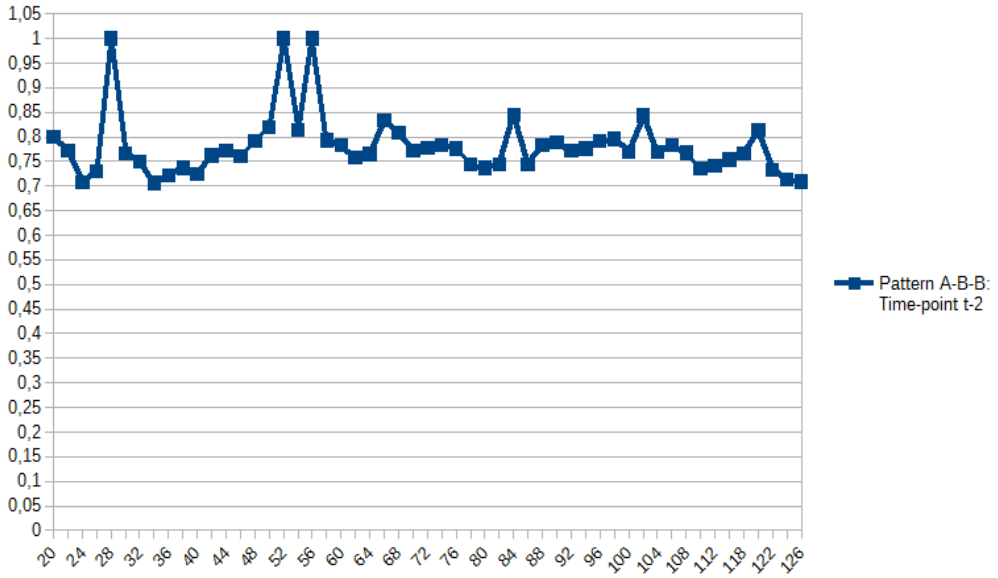


Figure 6.9: Rank-based predictability for pattern A-B-B at time-point t-2. X-axis: number of selected examples. Y-axis: predictability. Spikeness was estimated with the Jensen-Shannon divergence measure. The number of selected examples composing each subset contains an equal number of top-ranked selected examples from the positive class and bottom-ranked selected examples from the negative class. Therefore, in the subset with 20 selected examples, the top-ranked selected positive examples are 10 and the bottom-ranked selected negative examples are 10. The threshold value used is 0.7.

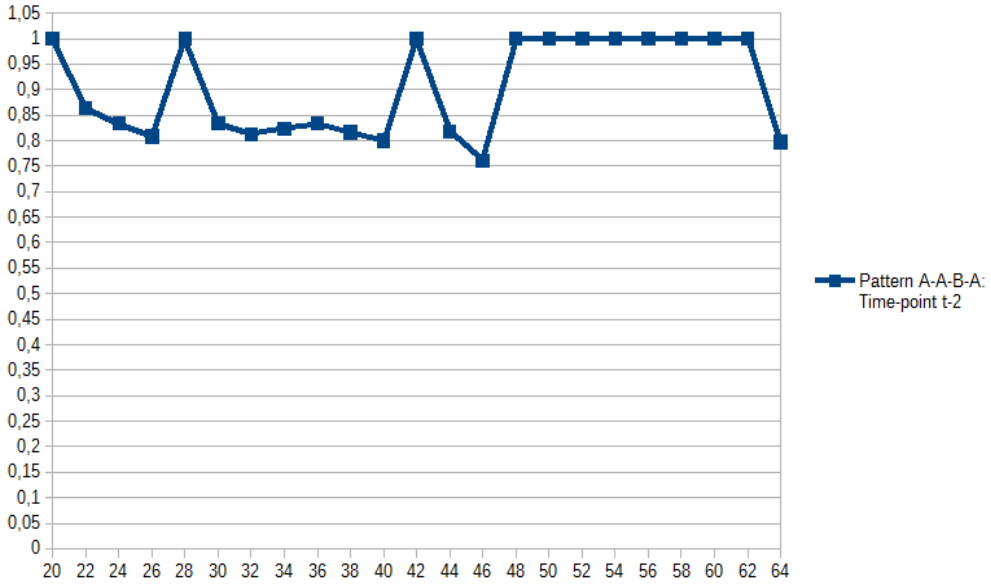


Figure 6.10: Rank-based predictability for pattern A-A-B-A at time-point t-2. X-axis: number of selected examples. Y-axis: predictability. Spikeness was estimated with the Jensen-Shannon divergence measure. The number of selected examples composing each subset contains an equal number of top-ranked selected examples from the positive class and bottom-ranked selected examples from the negative class. Therefore, in the subset with 20 selected examples, the top-ranked selected positive examples are 10 and the bottom-ranked selected negative examples are 10. The threshold value used is 0.7.

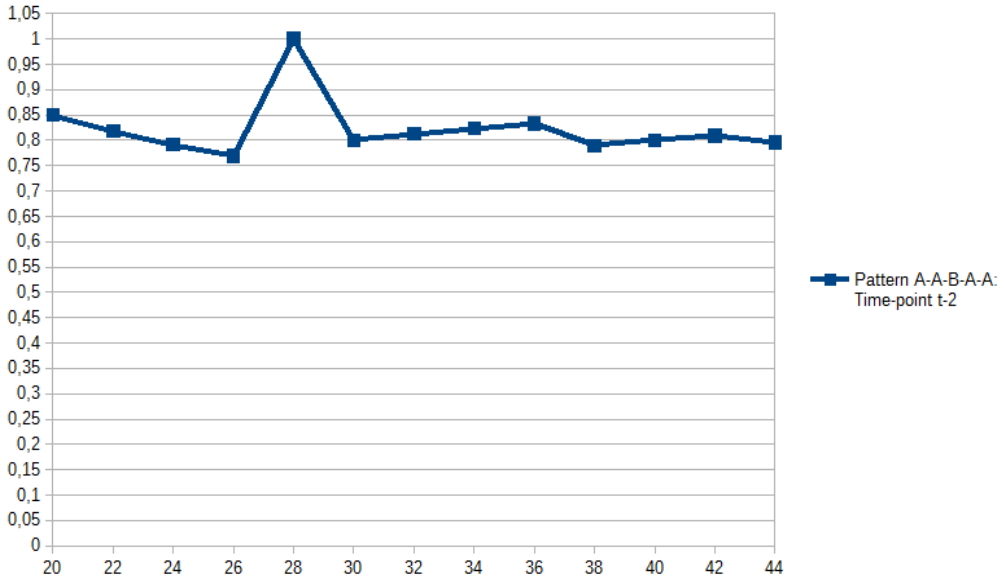


Figure 6.11: Rank-based predictability for pattern A-A-B-A-A at time-point t-2. X-axis: number of selected examples. Y-axis: predictability. Spikeness was estimated with the Jensen-Shannon divergence measure. The number of selected examples composing each subset contains an equal number of top-ranked selected examples from the positive class and bottom-ranked selected examples from the negative class. Therefore, in the subset with 20 selected examples, the top-ranked selected positive examples are 10 and the bottom-ranked selected negative examples are 10. The threshold value used is 0.7.

The experimentation continued with datasets $D_{t-1}$ and $D_{t-3}$, as well as repeated again with the Bray-Curtis measure for estimating spikeness. Since the spikeness with Bray-Curtis had different distribution than with Jensen-Shannon (figs. 6.6, 6.7), the ranking was different and thus predictability was different than before.

Figure 6.12 contained information regarding the rank-based predictability for pattern A-B-B at $D_{t-1}$ with the Jensen-Shannon measure. Its average predictability was 0.912 and was higher in comparison to the equivalent one at $D_{t-1}$ with the Jensen-Shannon measure, which was 0.781. This result strengthened the hypothesis that time-points closer to a temporal change than others, have been expected to be better predictors than the rest. Similar results were found for the rest of the patterns at $D_{t-1}$.

Figure 6.13 contained similar information for pattern A-B-A with the Bray-Curtis measure. Its average predictability was 0.82 and was higher in comparison to the

equivalent one at $D_{t-2}$ with the Jensen-Shannon measure, which was 0.79. While the result did not indicate an optimality of the measure for estimating spikeness (eq. 4.1) over Jensen-Shannon, it did indicate that this measure was as effective as Jensen-Shannon in rank-based example selection.

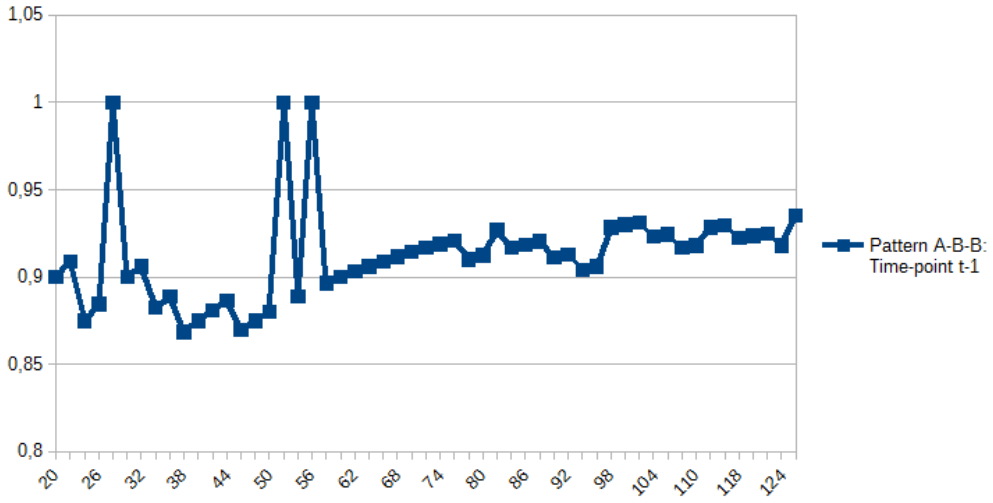


Figure 6.12: Rank-based predictability for pattern A-B-B at time-point t-1. X-axis: number of selected examples. Y-axis: predictability. Spikeness was estimated with the Jensen-Shannon divergence measure. The number of selected examples composing each subset contains an equal number of top-ranked selected examples from the positive class and bottom-ranked selected examples from the negative class. Therefore, in the subset with 20 selected examples, the top-ranked selected positive examples are 10 and the bottom-ranked selected negative examples are 10. The threshold value used is 0.7.

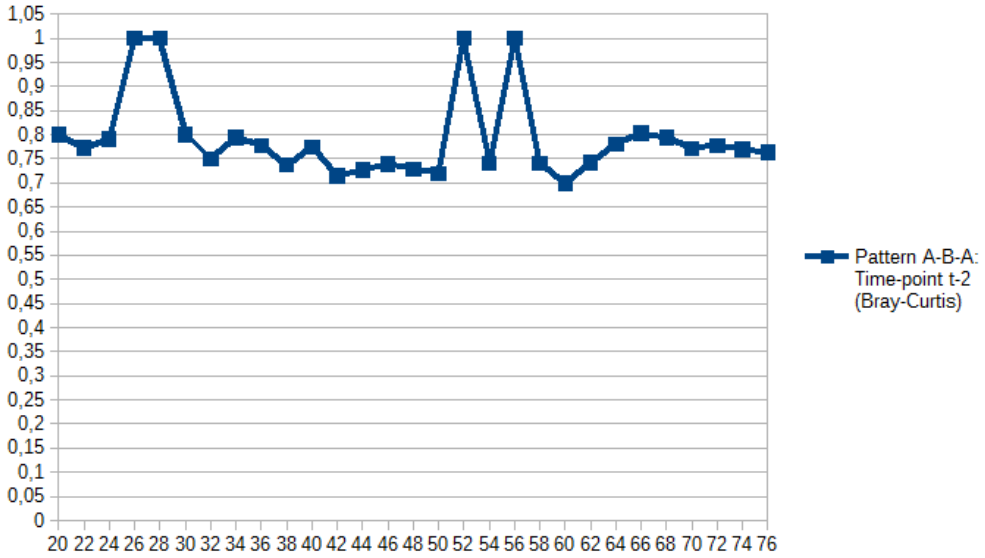


Figure 6.13: Rank-based predictability for pattern A-B-A at time-point t-2. X-axis: number of selected examples. Y-axis: predictability. Spikeness was estimated with the Bray-Curtis dissimilarity measure. The number of selected examples composing each subset contains an equal number of top-ranked selected examples from the positive class and bottom-ranked selected examples from the negative class. Therefore, in the subset with 20 selected examples, the top-ranked selected positive examples are 10 and the bottom-ranked selected negative examples are 10. The threshold value used is 0.7.

As far as coverage of the aforementioned datasets was concerned, the results were summarized in tables 6.1 and 6.2. Note that positive coverage is equal to 1 in almost all cases.

Table 6.1: Total Coverage for all datasets per pattern (Jensen-Shannon). X-axis: tested pattern. Y-axis: feature vectors per time-point.

| Dataset/Pattern | ABA | ABB | AABA | AABAA |
|---|---|---|---|---|
| $D_{t-1}$ | 0.17 | 0.16 | 0.18 | 0.19 |
| $D_{t-2}$ | 0.17 | 0.16 | 0.18 | 0.19 |
| $D_{t-3}$ | 0.09 | 0.16 | 0.18 | 0.19 |

Table 6.2: Positive Coverage for all datasets per pattern (Jensen-Shannon). X-axis: tested pattern. Y-axis: feature vectors per time-point.

| Dataset/Pattern | ABA | ABB | AABA | AABAA |
|---|---|---|---|---|
| $D_{t-1}$ | 1 | 1 | 1 | 1 |
| $D_{t-2}$ | 1 | 1 | 1 | 1 |
| $D_{t-3}$ | 0.52 | 1 | 1 | 1 |

### 6.4.2 Imbalanced Predictability

The imbalanced predictability was tested for all patterns at $D_{t-2}$. The imbalanced subsets were classified as clarified in the beginning of that section. For each subset, the results from balanced and imbalanced predictability were gathered together, and the subsets were presented in an ascending order. Furthermore, the f-measure was used to estimate the classification performance of the black-box classifier models, since the accuracy measure is not suitable for imbalanced datasets.

According to the predictability results in figures 6.14, 6.15, 6.16 and 6.17, the number of selected bottom negative examples has been extended for all patterns in greater numbers than the positive ones. The four tested patterns were predictable for a number of subsets constructed with the extended bottom negative examples and all positive examples. For all predictable patterns, the maximum number of negative examples was greater than the positives. While the total coverage was not large, it indicated the borders between the gray-zone and the top and bottom examples (sec. 4.3).

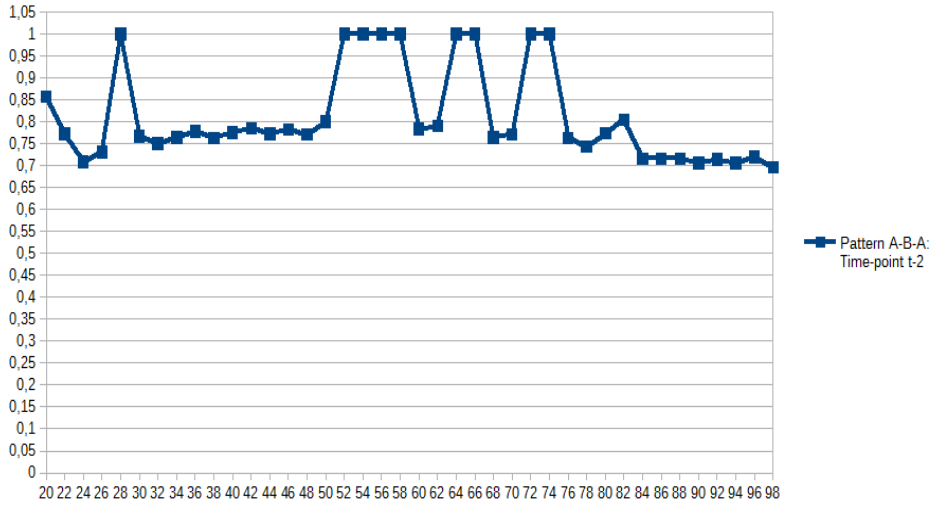


Figure 6.14: Rank-based imbalanced predictability for pattern A-B-A at time-point t-2. X-axis: number of selected examples. Y-axis: predictability. Spikeness was estimated with the Jensen-Shannon dissimilarity measure. The number of selected examples composing each subset contains an equal number of top-ranked selected examples from the positive class and bottom-ranked selected examples from the negative class until number 72. Afterwards, since the maximum number of positive examples (38) has been reached, the subsets selected contain 38 positive examples and n-38 negative examples, where n is the size of the subset. The threshold value used is 0.7.

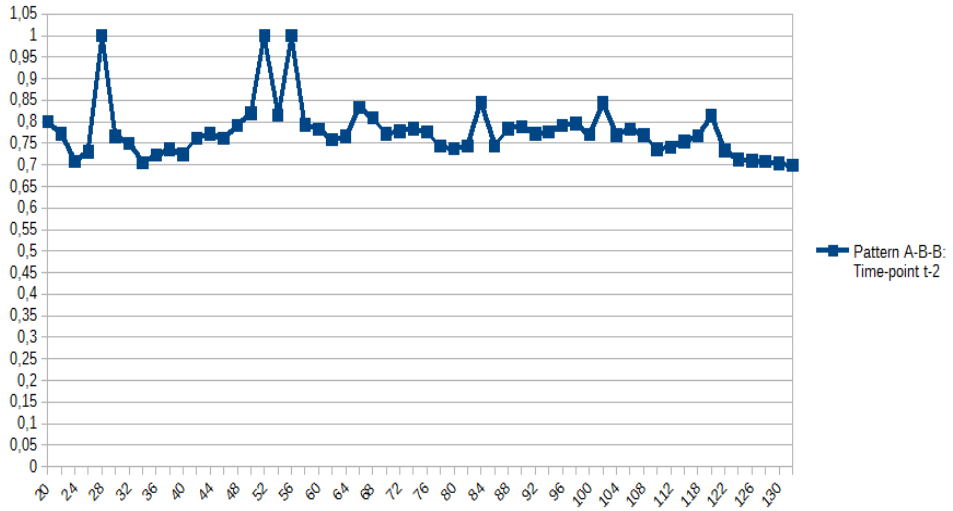


Figure 6.15: Rank-based imbalanced predictability for pattern A-B-B at time-point t-2. X-axis: number of selected examples. Y-axis: predictability. Spikeness was estimated with the Jensen-Shannon dissimilarity measure. The number of selected examples composing each subset contains an equal number of top-ranked selected examples from the positive class and bottom-ranked selected examples from the negative class until number 124. Afterwards, since the maximum number of positive examples (62) has been reached, the subsets selected contain 62 positive examples and n-62 negative examples, where n is the size of the subset. The threshold value used is 0.7.

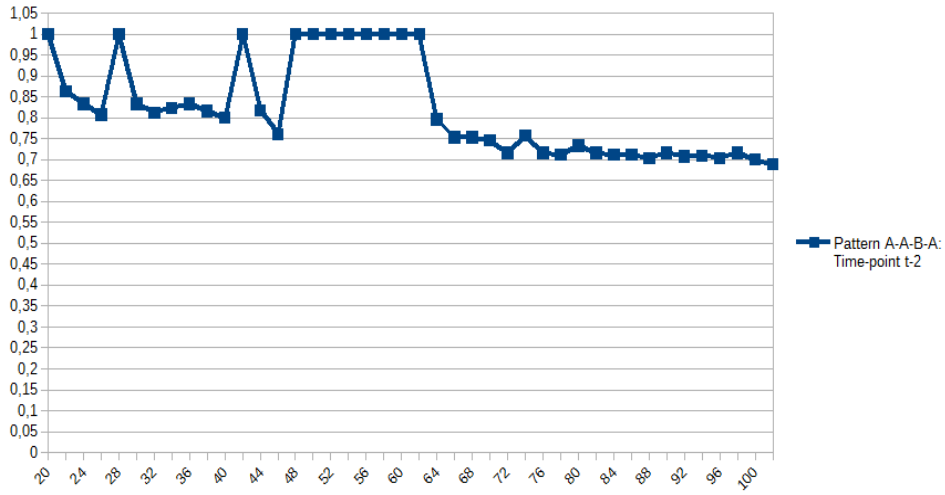


Figure 6.16: Rank-based imbalanced predictability for pattern A-A-B-A at time-point t-2. X-axis: number of selected examples. Y-axis: predictability. Spikeness was estimated with the Jensen-Shannon dissimilarity measure. The number of selected examples composing each subset contains an equal number of top-ranked selected examples from the positive class and bottom-ranked selected examples from the negative class until number 66. Afterwards, since the maximum number of positive examples (33) has been reached, the subsets selected contain 33 positive examples and n-33 negative examples, where n is the size of the subset. The threshold value used is 0.7.

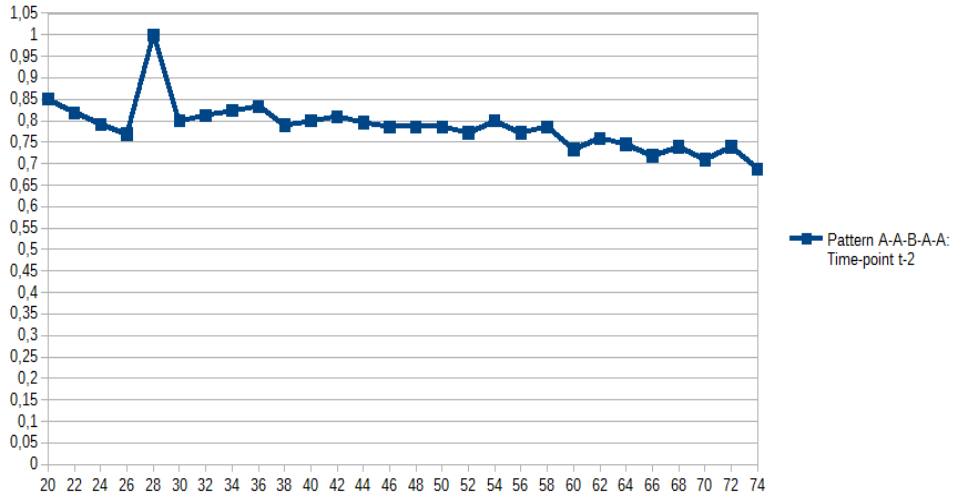


Figure 6.17: Rank-based imbalanced predictability for pattern A-A-B-A-A at time-point t-2. X-axis: number of selected examples. Y-axis: predictability. Spikeness was estimated with the Jensen-Shannon dissimilarity measure. The number of selected examples composing each subset contains an equal number of top-ranked selected examples from the positive class and bottom-ranked selected examples from the negative class, until number 44. Afterwards, since the maximum number of positive examples (22) has been reached, the subsets selected contain 22 positive examples and n-22 negative examples, where n is the size of the subset. The threshold value used is 0.7.

### 6.4.3 Rank-Based vs Random-Based Predictability

The random-based predictability procedure was repeated 10 times, and the mean, standard deviation, minimum and maximum values of all outputs were estimated.

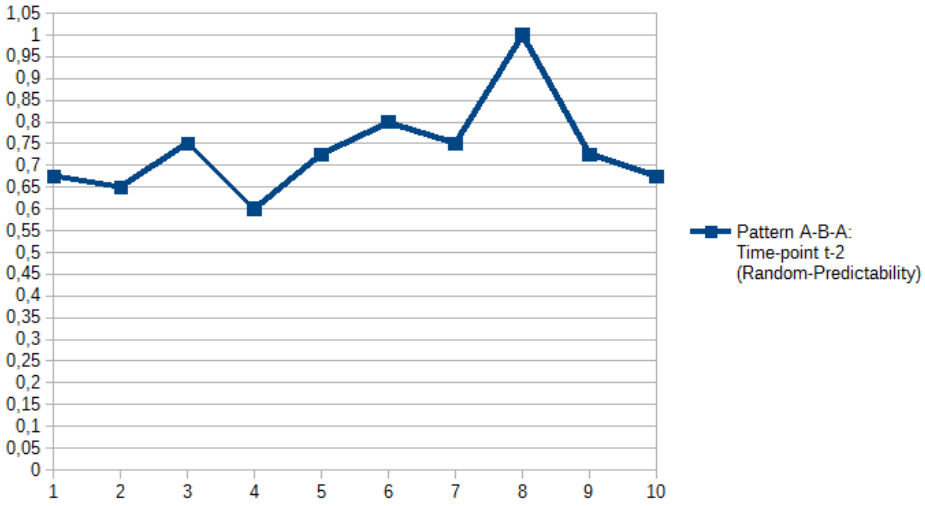


Figure 6.18: Random-based predictability for pattern A-B-A at time-point t-2 with 40 selected examples. X-axis: under-sampling repetitions. Y-axis: predictability. The number of selected examples composing the subset contains an equal number of the positive and negative class, hence 20 positive and 20 negative examples. The positive and negative examples were selected randomly from the dataset.

Figure 6.18 included the results of random-based predictability at time-point t-2, for pattern A-B-A with 20 randomly selected positive and 20 randomly selected negative examples, repeated 10 times. The results indicated predictability values with: mean = 0.735, standard deviation = 0.109, minimum value = 0.6, maximum value = 1. Despite the mean being bellow 0.79, which was the equivalent predictability of the rank-based approach, the maximum value exceeded it. This could be contributed to the fact that while 40 examples were selected, the rank-based predictability experiments for A-B-A (fig. 6.8) produced predictable subsets with maximum size = 76. Therefore, in case that the top 20 positive and bottom 20 negative examples were not selected, there were still 18 top positive and 18 bottom negative examples which, if randomly included in the subset, could produce high predictability values.

## 6.5 Discussion

The rank-based predictability of pattern ABA at time-point t-3 indicated a maximum predictable subset of both 20 top positive and 20 bottom negative examples for Jensen-Shannon and 22 top positive and 22 bottom negative examples for Bray-Curtis. Therefore, positive coverage in the first case was $52\%$ and in the second case $57\%$. For the rest of the patterns in time-point t-3 and for all patterns in time-points t-2 and t-1 with both dissimilarity measures, which accounted for 22 cases in total, the positive coverage was $100\%$ (table 6.2). Hence, the coverage for the positive examples was complete in almost all cases which indicated that the spikeness distribution of positive examples was differentiated than the bottom negative ones with size equal to the positives (indicative examples in figures 6.6 and 6.7).

Moreover, the results of rank-based predictability for pattern ABA with 40 selected examples (fig. 6.8) were reasonable when compared to the equivalent ones of random-based predictability (fig. 6.18). As shown in subsection 6.4.3, the existence of highly predictable random subsets could be accounted to the complete coverage of positive examples for most of the cases, as shown in the above paragraph. Moreover, the average random predictability did not exceed in any experiment the equivalent rank-based one (sub. 6.4.3), which indicated the superiority of selecting subsets based on spikeness, in terms of predictability.

The predictability per time-point for all patterns indicated that $D_{t-1}$ had higher values (indicative example in fig. 6.12), which could be interpreted as: predictors closer to a temporal change have been more accurate than the rest.

The comparison between the two dissimilarity measures used in terms of predictability indicated that the predictability of patterns for all time-points was quite similar between the Bray-Curtis and Jensen-Shannon approach for estimating spikeness (indicative example in fig. 6.13). Therefore, Bray-Curtis as a measure for estimating spikeness was proved to be as effective as Jensen-Shannon in terms of rank-based example selection.

Finally, the total coverage was not high (table 6.1). However, in all cases negative coverage was higher than positive coverage which indicated that there existed a decent percentage of negative examples which were differentiated from the positive ones in black-box classification.

# Chapter 7

# Conclusion and Future Work



## 7.1 Conclusion

This research investigated the predictability of temporal changes between various states in a longitudinal dataset composed of vaginal microbiome data. Initially, the analysis focused on the prediction of double changes in microbial composition, which were named as spikes. A hypothesis associating the spikes with fluctuations in the beta diversity of the time-points was made. According to that hypothesis, time-points present 1-3 time-points before a spike positive or negative one, were labeled as positive or negative. With regards to the temporal distance between a labeled time-point and a spike positive or negative one, datasets were constructed with the two categories and were classified with various classification models. The classification performance per dataset was associated with the predictability of the spike.

The classification results for all datasets were close to 0.7. To test for better results, subsets of the datasets with classification performance greater than the complete dataset were examined. A continuous measure describing the amount of change in composition between consecutive time-points, named spikeness was estimated for all time-points. The examples of each dataset were ranked based on spikeness and subsets based on the ranking were constructed. This procedure was named as rank-based

selection. Moreover, a new measure was used for quantifying the predictability of the various datasets and their subsets. Hence, predictability was defined as the generalization performance of an optimal classification system applied on a dataset and tested with a given measure. A dataset was considered predictable if its predictability exceeded a user-specified threshold. The optimal classification system was the black-box classifier, at which a given dataset was classified by a set of various classification models and external model parameters, wherein the classification results were derived from the best performing model. Finally, gradually increasing subsets of the dataset based on the top ranked positive and bottom ranked negative examples were tested on their predictability until the subsets contained K top ranked positive and K bottom ranked negative examples, were K was equal to the number of positive examples. This methodology was named as rank-based predictability and was compared with the predictability of randomly selected subsets. The results were satisfactory, since rank-based predictability indicated the existence of subsets with higher predictability than the complete dataset and higher or equal predictability to randomly selected ones.

Furthermore, the detection of spikes based on the beta diversity hypothesis was prone to errors. Moreover, according to this hypothesis, manually included thresholds were used to label the examples in the two categories. To automate the approach, a new way of detecting temporal changes was used. Instead of threshold values, the feature vectors at all time-points were discretized and represented by a symbolic value called state. The discretization was applied through clustering and thus the states were the clustering labels. Patterns of temporal changes in state like the spike, were detected with the use of a symbolic representation. The detection of the temporal patterns indicated a number of time-points as pattern positive or pattern negative. Feature vectors at time-points preceding the pattern positive/negative examples by 1-3 time-points were labeled as positive or negative ones and a number of datasets was constructed again based on the time-point distance from the pattern positive/negative one. The rank-based predictability approach was applied to the datasets, with various dissimilarity measures estimating spikeness which was used for ranking. The predictability results were compared with the random-based ones and exceeded their average predictability indicating that the rank-based predictability approach was superior to the random-based approach. Rank-based predictability was tested for a number of different temporal patterns, for a big number of subsets, different time-points preceding

the changes and different ways of measuring spikeness. The conclusion made from the analysis was that the tested patterns were predictable for subsets having a high coverage of the positive examples.

## 7.2 Future Work

As far as future work is concerned, the most important work could be the application of the algorithm for detecting predictable changes in other real time-series or longitudinal datasets. Even without pre-processing, the dataset used contained 937 time-points and thus its size was small in comparison to other available time-series datasets. Therefore, it has been of major interest to test the algorithm on larger datasets, as well as datasets of different origin like macro-economic time-series.
Another future work could be the inclusion of more classifiers to the black-box for a more optimal generalization performance. Since the black-box classifier utilized the most optimal classifier at a time, additional classifiers could increase the probability of improved classification accuracy. An example of alternative classification models for testing would be Bayesian Networks. As mentioned in [33], Multi-level Temporal Bayesian Networks have been useful in analyzing hierarchical health care data and thus could be applied to longitudinal or time-series data. However, statistical machine learning models have worked optimally with large datasets, and thus larger dataset than the one used in this analysis would be preferred.
The algorithm for detecting predictable temporal changes was based on discretizing the data on states. Discretization was prone to errors since it was based on a majority voting of clustering approaches, for whom the clustering performance optimality could not be certain. A different approach would be the detection of temporal changes based on continuous instead of discrete values. Instead of clustering the data, each time-point could be represented by its dissimilarity in composition with the previous one, as used in the analysis for estimating spikeness and for the beta diversity spike hypothesis. Since all time-points could be represented by a continuous value, then the ranking could be performed without class labels: the top k and bottom k examples would be chosen based on spikeness or another dissimilarity value, where k is user defined, and they would constitute the positive and negative category applied to rank-based predictability. Hence, instead of being pre-defined based on discretization,

class labeling would be performed for each rank-based selected subset separately. The advantage of this approach is that it is not affected by clustering errors and is based on direct dissimilarity between two consecutive time-points.

Another future work reference would be the implementation of feature selection to the rank-based predictability approach. After the predictability of each selected subset of the dataset has been estimated, a feature selection algorithm would be applied for selecting the most important features of the subset that were responsible for the result. The information of feature importance would be valuable in fields like Biology. By taking as an example the relative abundances of bacteria used in this thesis, the discovery of the most important bacteria in predictable subsets could lead the biologists into discovering important properties of those bacteria or searching for useful existing information.

Moreover, there is further work to be made with regards to prediction of temporal changes. The feature vectors used for estimating predictability of temporal changes belonged to time-points t-1, t-2 and t-3 before the pattern positive/negative time-point t. Earlier time-points like t-4, t-5 and t-6 could be included in the analysis in order to comprehend how backwards in time can feature vectors be selected for estimating predictability.

Furthermore, the datasets used in the analysis contained a positive and a negative category. The negative category was the category of examples for whom the pattern for search was not detected. However, despite the fact the the negative examples did not correspond to the searched pattern, they could correspond to another pattern of temporal change. Examples such as the aforementioned are indicative of temporal changes in state. There are two possible approaches for utilizing such examples in the analysis. The first approach, would be the estimation of predictability through a multi-class classification in which each pattern detected in the analysis would constitute a class/category. The examples which did not correspond to a pattern would be included in a negative class. A second approach, would be to include the examples corresponding to all detected patterns to a positive category. In this case, predictability would be estimated through a two-class classification in which the number of positive examples would be greater than in the current approach.

Finally, more clustering approaches could be applied to the discretization phase of detecting predictable temporal changes. Hierarchical clustering methods would still be preferred, since the number of clusters is not user defined. A divisive hierarchical

clustering method could be applied, with number of clusters decided based on an evaluation measure like Silhouette (section 2.3). Furthermore, the dip-means algorithm could be applied, since it utilizes the dip-dist criterion like the agglodip algorithm (section 2.3), and its performance on clustering was superior when compared with various clustering methods [23].

# Bibliography


[1] P. Gajer, R. Brotman, G. Bai, J. Sakamoto, U. Schütte, X. Zhong, S. Koenig, L. Fu, Z. Ma, X. Zhou, Z. Abdo, L. Forney, and J. Ravel, "Temporal dynamics of the human vaginal microbiota," *Science Translational Medicine*, vol. 4, no. 132, p. 132ra52, 2012.

[2] J. Ravel, P. Gajer, Z. Abdo, G. Schneider, S. Koenig, S. McCulle, S. Karlebach, R. Gorle, J. Russell, C. Tacket, R. Brotman, C. Davis, K. Ault, L. Peralta, and L. Forney, "Vaginal microbiome of reproductive-age women," *Proceedings of the National Academy of Sciences (PNAS)*, vol. 108, no. 1, p. 4680–4687, 2011.

[3] E. Costello, C. Lauber, M. Hamady, N. Fierer, J. Gordon, and R. Knight, "Bacterial community variation in human body habitats across space and time," *Science*, vol. 326, no. 5960, pp. 1694–1697, 2009.

[4] J. Willey, L. Sherwood, and C. Woolverton, *Prescott's Microbiology*. McGraw Hill, ninth ed., 2013.

[5] M. Nakai and W. Ke, "Statistical models for longitudinal data analysis," *Applied Mathematical Sciences*, vol. 3, no. 40, pp. 1979–1989, 2009.

[6] G. van Belle, D. Lloyd, P. Heagerty, and T. Lumley, *Biostatistics: a methodology for the Health sciences*. John Wiley and Sons Inc, second ed., 2004.

[7] C. Tamboli, C. Neut, P. Desreumaux, and J. Colombel, "Dysbiosis in inflammatory bowel disease," *Journal of Clinical Microbiology (JCM)*, vol. 53, no. 1, p. 1–4, 2004.

[8] Q. Wang, G. Garrity, J. Tiedje, and J. Cole, "Naïve bayesian classifier for rapid assignment of rrna sequences into the new bacterial taxonomy," *Applied and Environmental Microbiology*, vol. 73, no. 16, p. 5261–5267, 2007.

[9] R. Nugent, M. Krohn, and S. Hillier, "Reliability of diagnosing bacterial vaginosis is improved by a standardized method of gram stain interpretation," *Journal of Clinical Microbiology (JCM)*, vol. 29, no. 2, pp. 297–301, 1991.

[10] C. M. Bishop, *Pattern Recognition and Machine Learning*. Information Science and Statistics, Springer, first ed., 2006.

[11] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Proceedings of the 14th international joint conference on Artificial intelligence (IJCAI)*, vol. 2, pp. 1137–1143, Morgan Kaufmann Publishers Inc, 1995.

[12] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[13] P. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*. Pearson, first ed., 2005.

[14] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft, "When is "nearest neighbor" meaningful?," in *ICDT: International Conference on Database Theory*, pp. 217–235, 1999.

[15] N. Tomašev, K. Buza, K. Marussy, and P. Kis, "Hubness-aware classification, instance selection and feature construction: Survey and extensions to time-series," *Studies in Computational Intelligence*, vol. 584, pp. 231–262, 2014.

[16] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*. Academic Press, fourth ed., 2008.

[17] P. Jaskowiak and R. Campello, "Comparing correlation coefficients as dissimilarity measures for cancer classification in gene expression data," in *Brazilian Symposium on Bioinformatics*, pp. 231–262, 2011.

[18] R. Fisher, "The use of multiple measurements in taxonomic problems," *Annals of eugenics*, 1936.

[19] G. Hinton and R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[20] Y. Bengio, I. Goodfellow, and A. Courville, *Deep Learning*. MIT Press, first ed., 2015.

[21] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Advances in Neural Information Processing Systems 19*, pp. 153–160, MIT Press, 2006.

[22] P. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, vol. 20, pp. 53–65, 1987.

[23] A. Kalogeratos and A. Likas, "Dip-means: an incremental clustering method for estimating the number of clusters," in *Advances in Neural Information Processing Systems 25 (NIPS 2012)*, pp. 2402–2410, 2012.

[24] J. Hartigan and P. Hartigan, "The dip test of unimodality," *The Annals of Statistics*, vol. 13, no. 1, pp. 70–84, 1985.

[25] S. Kullback and R. Leibler, "On information and sufficiency," *The Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[26] J. Lin, "Divergence measures based on the shannon entropy," *IEEE Transactions on Information Theory*, vol. 37, no. 1, pp. 37:–145, 1991.

[27] J. Bray and J. Curtis, "An ordination of the upland forest communities of southern wisconsin," *Ecological monographs*, vol. 7, no. 4, pp. 325–349, 1957.

[28] K. Clarke, J. Paul, and M. Chapman, "On resemblance measures for ecological studies, including taxonomic dissimilarities and a zero-adjusted bray–curtis coefficient for denuded assemblages," *Journal of Experimental Marine Biology and Ecology*, vol. 330, no. 1, pp. 55–80, 2006.

[29] J. Jacobs, "Using $\beta$-diversity and similarity/dissimilarity indices to measure diversity across sites, communities, and landscapes," tech. rep., Semantics Scholar, 2008.

[30] Y. Leung and D. Cavalieri, "Fundamentals of cdna microarray data analysis," *TRENDS in Genetics*, vol. 19, no. 11, pp. 649–659, 2003.

[31] T. Dietterich, "Ensemble methods in machine learning," in *Proceedings of the First International Workshop on Multiple Classifier Systems*, pp. 1–15, Springer-Verlag, 2000.

[32] S. Needleman and C. Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *Journal of Molecular Biology 1970*, vol. 48, no. 3, pp. 443–453, 1970.

[33] M. Lappenschaar, A. Hommersom, P. Luca, J. Lagro, and S. Visscher, "Multi-level bayesian networks for the analysis of hierarchical healthcare data," *Artificial Intelligence in Medicine*, vol. 57, no. 3, pp. 171–183, 2013.

# Short Biography

Nestor Timonidis was born in 1991 in Thessaloniki, Greece. He received his Diploma degree from the Department of Computer Science and Engineering of the University of Ioannina, Greece, in 2015. His Diploma Thesis was based on the development of an agglomerative clustering algorithm and its application on face image clustering. Since then, he has been a Master student at the ”Informatics” Postgraduate Program of the same Department, with specialization in Technologies-Applications. In 2014 he was an intern at Qbase *R&D*, where he participated at the development of the Alexilio project, an intelligent system that provided smart indications to alert people from overexposure to solar radiation. In 2016, he was an intern at the Centre for Molecular and Biomolecular Informatics (CMBI) of Radboud University Medical Centre in Nijmegen, Netherlands. His research interests include Machine Learning methods (Clustering and Classification) and their application on Life Sciences and Systems Biology.