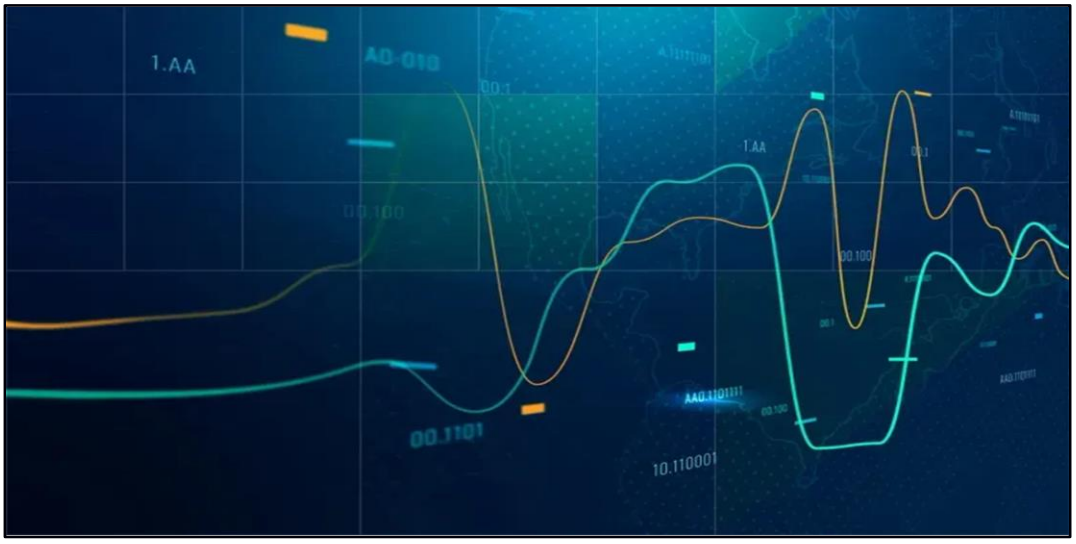
ΔΙΕΘΝΕΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΗΣ ΕΛΛΑΔΟΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

«Μελέτη αλγορίθμων Matrix Profile και πειραματική αξιολόγηση τους στη κατηγοριοποίηση χρονοσειρών με χρήση της γλώσσας προγραμματισμού Python»



Της φοιτήτριας
Αθηνά Στρίκου
Αρ. Μητρώου: 175134

Επιβλέπων
Λεωνίδας Καραμητόπουλος
Καθηγητής

Ημερομηνία 12/09/2025

Τίτλος Π.Ε. Μελέτη αλγορίθμων Matrix Profile και πειραματική αξιολόγηση τους στην κατηγοριοποίηση χρονοσειρών με τη χρήση της γλώσσας προγραμματισμού Python.

Κωδικός Π.Ε. 23325

Ονοματεπώνυμο φοιτήτριας Αθηνά Στρίκου

Ονοματεπώνυμο εισηγητή Λεωνίδας Καραμητόπουλος

Ημερομηνία ανάληψης Π.Ε. 05/11/2023

Ημερομηνία περάτωσης Π.Ε. 10/09/2025

Βεβαιώνω ότι είμαι ο συγγραφέας αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, έχω καταγράψει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών, εικόνων και κειμένου, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επιπλέον, βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά, ειδικά ως διπλωματική εργασία, στο Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του ΔΙ.ΠΑ.Ε.

Στον πατέρα μου και την μητέρα μου, η οποία μπορεί να μην είναι πλέον εδώ, αλλά πάντα τη νιώθω πάντα κοντά μου

Πρόλογος

Ώντας ένα άτομο με αγάπη για τον προγραμματισμό και την ανάλυση δεδομένων, ορισμένα μαθήματα της σχολής στο Τμήμα Πληροφορικής του Αλεξάνδρειου Τεχνολογικού Εκπαιδευτικού Ιδρύματος, βοήθησαν στην ανάπτυξη αυτών των ενδιαφερόντων. Αυτές οι δεξιότητες λοιπόν, αποτέλεσαν κύριο γνώμονα για την επιλογή του θέματος της πτυχιακής μου εργασίας. Με αυτόν τον τρόπο, κατάφερα να συνδυάσω τις ακαδημαϊκές γνώσεις αυτές με τις πρακτικές εφαρμογές σε μια πτυχιακή εργασία πάνω στην ανάλυση των χρονοσειρών. Επέλεξα την Python ως την γλώσσα προγραμματισμού λόγω της πρακτικότητας, των πλούσιων βιβλιοθηκών που παρέχει, αλλά και των βοηθητικών εργαλείων. Πιο συγκεκριμένα, χρησιμοποίησα τις βιβλιοθήκες STUMPY, Pandas, Numpy, Matplotlib, sklearn και SciPy. Μέσω αυτών κατάφερα να επιτελέσω τους απαραίτητους υπολογισμούς και τις οπτικοποιήσεις δεδομένων. Τα παραπάνω δε θα μπορούσαν να είναι δυνατά δίχως το Jupyter Notebook. Μιας πλατφόρμας που αποτέλεσε το κύριο περιβάλλον για την εκτέλεση του κώδικα. Κάθε ένα από αυτά έπαιξε το δικό του ρόλο στην δημιουργία της πτυχιακής εργασίας.

Το σύνολο δεδομένων που υλοποιήθηκε για την εύρεση των shapelet και τη κατηγοριοποίηση λέγεται BeetleFly. Βρίσκεται σε μια σελίδα όπου περιέχει σύνολα που χρησιμοποιούνται για κατηγοριοποίηση https://www.timeseriesclassification.com/description.php?Dataset=BeetleFly. Όλα αυτά προέρχονται από συλλογές όπως το UCR time series archive https://www.cs.ucr.edu/%7Eeamonn/time_series_data_2018/ και το UEA multivariate time series https://ueaeprints.uea.ac.uk/id/eprint/94905/. Για την συγκεκριμένη εργασία χρησιμοποιήθηκε από το UCR. Η κύρια σελίδα του timeseriesclassification φτιάχτηκε από τον Tony Bagnall. Βέβαια υπήρξαν πολλά άτομα που συνεισέφεραν σε αυτό με τον δικό τους τρόπο. Ερευνητές όπως οι Eamonn Keogh, ο Jason Lines, ο Aaron Bostrom, ο James Large και ο Matthew Middlehurst. Στόχος ήταν η δημιουργία ενός μέρους μέσα στον οποίο θα είναι συγκεντρωμένα σύνολα δεδομένων για εφαρμογές μηχανικής μάθησης.

Περίληψη

Στη σημερινή εποχή υπάρχει κατακλυσμός πληροφορίας. Δεδομένα αντλούνται με συνεχή ρυθμό από διάφορες πηγές. Όσο αυξάνεται η τεχνολογία, τόσο θα εντείνεται το θέμα. Οι διαστάσεις των χρονοσειρών θα πάρουν ακόμη μεγαλύτερο μέγεθος. Συνεπώς, είναι επιτακτική η χρήση τεχνικών κατάλληλων να διαχειριστούν το ζήτημα του όγκου. Με την υλοποίηση αυτών θα μπορούν να πραγματοποιούνται πιο εύκολα υπολογισμοί απόστασης. Βασικός στόχος είναι η εύρεση ομοιότητας ανάμεσα σε πρότυπα. Η έννοια αυτή εφαρμόζεται κατά κόρον από εργασίες εξόρυξης δεδομένων. Όπως στην ανακάλυψη μοτίβων, ανωμαλιών και στη κατηγοριοποίηση. Για την υλοποίηση των παραπάνω σημαντικό ρόλο παίζει το matrix profile. Μέσω αυτού γίνεται ο υπολογισμός της Ευκλείδειας απόστασης μεταξύ υποσειρών χρονοσειράς. Έπειτα οι μικρότερες τιμές κρατιούνται σε κάθε του θέση. Χάρη σε αυτό μπορούν να εξαχθούν μερικά αξιόλογα πορίσματα. Όπως για παράδειγμα στις χαμηλές τιμές υπάρχει μοτίβο, ενώ στις υψηλές ανωμαλία. Πολύ σημαντική είναι η χρήση του στην ανακάλυψη των shapelets. Ουσιαστικά είναι υποακολουθίες ικανές να διαφοροποιήσουν μεταξύ κλάσεων. Είναι κοινές για την δικιά τους κλάση, αλλά απέχουν πολύ από την άλλη. Μάλιστα βοηθούν άμεσα στην κατηγοριοποίηση δειγμάτων. Η ικανότητα τους κρίνεται από την τιμή ακρίβειας(accuracy) της διάκρισης που πετυχαίνουν. Για την ανάλυση τους χρησιμοποιήθηκε το σύνολο δεδομένων BeetleFly. Έχει μήκος 512 και αποτελείται από τις κλάσεις σκαθάρι και μύγα. Η πρώτη χαρακτηρίζεται από τιμή ετικέτας 1, ενώ η δεύτερη από 2. Το κομμάτι του κώδικα και της οπτικοποίησης πραγματοποιήθηκε από την Python και τις βιβλιοθήκες της. Πιο συγκεκριμένα της STUMPY, Pandas, Numpy, Matplotlib, sklearn και SciPy. Η κύρια εφαρμογή έγινε στο Jupyter Notebook. Με τη STUMPY έγινε υπολογισμός του matrix profile. Στο πρώτο κομμάτι βρέθηκαν τα shapelets της κλάσης σκαθαριού και στο δεύτερο για τη μύγα. Τέλος, μετρήθηκαν οι ακρίβειες για το καθένα και συγκρίθηκαν.

A STUDY OF MATRIX PROFILE ALGORITHMS AND EXPERIMENTAL EVALUATION IN TIME SERIES CATEGORIZATION USING THE PYTHON PROGRAMMING LANGUAGE

STRIKOU ATHINA

Abstract

In today's world, there is an information overload. Data is constantly being extracted from various sources. As technology advances, this issue will become even more pressing. Time series will become even larger in size. Therefore, it is imperative to use techniques that are suitable for managing the issue of data volume. With the implementation of these techniques, distance calculations can be performed more easily. The main objective is to find similarities between patterns. This concept is widely applied in data mining tasks. Such as in the discovery of patterns, anomalies and categorization. The matrix profile plays an important role in the implementation of the above. It is used to calculate the Euclidean distance between time series sub-sequences. Then the smallest values are kept in each position. Thanks to this, some noteworthy conclusions can be drawn. For example, there is a pattern in the low values, while there is an anomaly in the high values. Its use in the discovery of shapelets is very important. Essentially, these are subsequences capable of differentiating between classes. They are common to their own class, but very different from the others. In fact, they directly assist in the classification of samples. Their ability is judged by the accuracy of the distinction they achieve. The BeetleFly dataset was used for their analysis. It has a length of 512 and consists of the beetle and fly classes. The former is characterized by a label value of 1, while the latter by 2. The code and visualization were implemented using Python and its libraries. More specifically, STUMPY, Pandas, Numpy, Matplotlib, Sklearn and SciPy. The main application was done in Jupyter Notebook. STUMPY was used to calculate the matrix profile. In the first part, the shapelets of the beetle class were found, and in the second part, those of the fly class. Finally, the accuracies for each were measured and compared.

Ευχαριστίες

Αρχικά θα ήθελα να ευχαριστήσω ιδιαίτερα τον κύριο Λεωνίδα Καραμητόπουλο για την στήριξη καθόλη τη διάρκεια της πτυχιακής. Επίσης, τους φίλους μου που ήταν εκεί και με εμψύχωναν. Χωρίς αυτούς θα ήταν δύσκολα τα πράγματα. Τέλος θέλω να ευχαριστήσω τον πατέρα και την μητέρα μου, που δεν βρίσκεται πια κοντά μου. Αλλά εννοείται πως η στήριξη που πήρα θα με συντροφεύει για πάντα.

Περιεχόμενα

Πρόλογος 4
Περίληψη 5
Abstract 6
Ευχαριστίες 7
Περιεχόμενα 8
Κατάλογος Σχημάτων 10
Κατάλογος Πινάκων 11
Κεφάλαιο 1ο: Εισαγωγή 12
1.1 Διαδικασία εξόρυξης γνώσης-Data Mining 12
1.2 Εξόρυξη γνώσης με τη χρήση χρονοσειρών 13
1.2.1 Ορισμός χρονοσειράς 15
1.2.2 Εφαρμογές και χρησιμότητα 17
1.3 Παρουσίαση του Matrix Profile και της κατηγοριοποίησης 18
1.4 Επίλογος 21
Κεφάλαιο 2ο: Τεχνολογίες και Εργαλεία 22
2.1 Γλώσσα προγραμματισμού Python 22
2.1.1 Βιβλιοθήκες Python για ανάλυση χρονοσειρών 22
2.2 Εργαλείο Jupyter 24
2.2.1 Υπέρ και κατά του Jupyter 24
2.2.2 Αναπαράσταση χρονοσειρών μέσω Jupyter 26
2.3 Επίλογος 27
Κεφάλαιο 3ο: Θεωρητικό υπόβαθρο 28
3.1 Εργασίες - Στόχοι της διαδικασίας εξόρυξης δεδομένων μέσω της ανάλυσης χρονοσειρών 28
3.1.1 Query by Content 28
3.1.2 Prediction 29
3.1.3 Clustering 30
3.1.4 Motif Discovery 31
3.1.5 Classification 32
3.1.6 Ανακάλυψη κανόνων συσχέτισης 33
3.1.7 Anomaly Detection 34
3.1.8 Segmentation 35
3.2 Διαδικασία αναζήτησης όμοιων χρονοσειρών 36
3.2.1 Η έννοια της ομοιότητας χρονοσειρών 36
3.2.2 Μέτρα απόστασης χρονοσειρών 37
3.3 Μείωση Διαστάσεων ή Αναπαραστάσεις Χρονοσειρών 39
3.3.1 Discrete Fourier Transform (DFT) 39
3.3.2 Discrete Wavelet Transform (DWT) 40
3.3.3 Piecewise Aggregate Approximation (PAA) 41
3.3.4 Symbolic Aggregate Approximation (SAX) 41
3.4 Επίλογος 43
Κεφάλαιο 4ο: Αλγόριθμοι Matrix Profile 44

4.1 Εισαγωγή στο Matrix Profile 44
4.2 Ο αλγόριθμος STAMP 46
4.2.1 Περιγραφή και ανάλυση του αλγορίθμου 48
4.2.2 Λειτουργία του MASS και του Sliding Dot 49
4.3 Ο αλγόριθμος Brute Force 50
4.4 Εφαρμογές του Brute Force 52
4.5 Επίλογος 53
Κεφάλαιο 5ο: Πειραματική Μεθοδολογία 54
5.1 Σύνολο δεδομένων BeetleFly 54
5.2 Προεπεξεργασία δεδομένων 55
5.3 Εξαγωγή shapelets 56
5.4 Διαδικασία Κατηγοριοποίησης 57
5.5 Επίλογος 58
Κεφάλαιο 6ο: Αποτελέσματα 59
6.1 Πειραματική επιλογή της παραμέτρου m 59
6.2 Ανάλυση των shapelets του beetle dataframe 60
6.3 Εξαγωγή shapelets από την κλάση fly 62
6.4 Συνολική αξιολόγηση και παρατηρήσεις 64
6.5 Επίλογος 67
Κεφάλαιο 7ο: Συμπεράσματα και προτάσεις βελτίωσης 68
7.1 Συμπεράσματα μελέτης 68
7.2 Προτάσεις βελτίωσης 68
ΒΙΒΛΙΟΓΡΑΦΙΑ 70
ΠΑΡΑΡΤΗΜΑ: ΠΕΡΙΓΡΑΦΗ ΚΩΔΙΚΑ 73
Π1. Εισαγωγή βιβλιοθηκών Python στο Jupyter 73
Π2 Φόρτωση του συνόλου δεδομένων train 73
Π3 Διάσπαση του train 74
Π4 Εμφάνιση των χρονοσειρών beetle και fly 74
Π5 Υπολογισμός και Οπτικοποίηση Matrix Profile 76
Π6 Εύρεση και εμφάνιση κορυφαίων σημείων 77
Π7 Παρουσίαση των shapelets στο beetle 78
Π8 Αξιολόγηση των shapelets 79

Κατάλογος Σχημάτων

Σχήμα 1.2.1.Η αναπαράσταση του GSTA [36] 13
Σχήμα 1.2.1.1. Αναπαράσταση υποακολουθίας χρονοσειράς Q [8] 16
Σχήμα 1.2.1.2 Αναπαράσταση οικονομικής χρονοσειράς [3] 16
Σχήμα 1.2.2.1 Αναπαράσταση ταχύτητας αέρα [1] 18
Σχήμα 1.3.1 Αναπαράσταση Distance Profile μιας χρονοσειράς ECG-II [13] 19
Σχήμα 1.3.2 Αναπαράσταση εύρεσης shapelet [16] 20
Σχήμα 2.1.1.1 Ο υπολογισμός του Matrix Profile με χρήση της βιβλιοθήκης STUMPY [26] 24
Σχήμα 2.2.1.1 Χαρακτηριστικά του Jupyter [27] 25
Σχήμα 2.2.2.1 Ένα παράδειγμα εισαγωγής και εμφάνισης δεδομένων στο Jupyter 26
Σχήμα 2.2.2.2 Απεικόνιση δεδομένων στο Jupyter 27
Σχήμα 3.1.1.1 Αναπαράσταση Query by Content [9] 29
Σχήμα 3.1.2.1 Αναπαράσταση Prediction [9] 30
Σχήμα 3.1.3.1 Αναπαράσταση clustering αλγόριθμου [9] 31
Σχήμα 3.1.4.1 Αναπαράσταση Motif Discovery [9] 32
Σχήμα 3.1.5.1 Αναπαράσταση αλγορίθμου Classification [9] 33
Σχήμα 3.1.6.1 Πίνακας με δεδομένα καιρού [10] 34
Σχήμα 3.1.7.1 Αναπαράσταση αλγορίθμου εύρεσης ανωμαλιών [9] 35
Σχήμα 3.1.8.1 Αναπαράσταση Segmenation [9] 36
Σχήμα 3.3.3.1 Αναπαράσταση Piecewise Aggregate Approximation [21] 41
Σχήμα 3.3.4.1 Αναπαράσταση Symbolic Aggregate Approximation [21] 42
Σχήμα 4.1.1 Αναπαράσταση προφίλ απόστασης [31] 46
Σχήμα 4.1.2 Αναπαράσταση προφίλ πίνακα και θέσης [31] 46
Σχήμα 4.2.1 Αναπαράσταση γραφημάτων ενός συστήματος ψύξης [29] 47
Σχήμα 4.2.1.1 Αναπαράσταση αλγόριθμου STAMP[29] 48
Σχήμα 4.2.2.1 Αναπαράσταση SlidingDotProduct [41] 49
Σχήμα 4.2.2.2 Αναπαράσταση MASS [41] 50
Σχήμα 4.3.1 Αναπαράσταση Brute-force για την εύρεση ασυμφωνιών [33] 51
Σχήμα 4.3.2 Αναπαράσταση Brute-force για την εύρεση shapelets [16] 52
Σχήμα 5.1.1 Μετατροπή από σχήμα σε χρονική σειρά [43] 54
Σχήμα 5.1.2 Τα σύνολα δεδομένων Beetle/Fly και Bird/Chicken [43] 54
Σχήμα 5.2.1 Μετατροπή από .txt σε .csv 55
Σχήμα 5.2.2 Οπτικοποίηση ενός δείγματος σκαθαριού 55
Σχήμα 5.2.3 Οπτικοποίηση ενός δείγματος μύγας 55
Σχήμα 6.1.1 Αποτελέσματα ακρίβειας για m = 40 59
Σχήμα 6.1.2 Αποτελέσματα ακρίβειας για m = 50 59
Σχήμα 6.1.3 Αποτελέσματα ακρίβειας για m = 80 60
Σχήμα 6.1.4 Αποτελέσματα ακρίβειας για m = 59 60
Σχήμα 6.2.1 Εμφάνιση των Matrix Profile Pbeetle,beetle και Pbeetle,fly. 61
Σχήμα 6.2.2 Εμφάνιση των κορυφαίων 10 σημείων. 62
Σχήμα 6.2.3 Εμφάνιση των shapelets στο beetle_df 62
Σχήμα 6.3.1 Εμφάνιση matrix profile Pfly,fly και Pfly,beetle 63
Σχήμα 6.3.2 Εμφάνιση κορυφαίων shapelets για την κλάση fly 63
Σχήμα 6.3.3 Αναπαράσταση των shapelets στο fly dataframe 64
Σχήμα 6.3.4 Μέτρηση ακρίβειας των shapelets στην κλάση fly 64
Σχήμα 6.4.1 Αναπαράσταση shapelets για m = 110 65
Σχήμα 6.4.2 Αναπαράσταση shapelets για m = 250 65
Σχήμα 6.4.3 Αναπαράσταση shapelets για m = 23 66
Σχήμα 6.4.4 Αναπαράσταση shapelets για m = 80 66

Κατάλογος Πινάκων

Πίνακας 3.3.4.1 Τιμές σημείων διακοπής [21] 43

Κεφάλαιο 1ο: Εισαγωγή

1.1 Διαδικασία εξόρυξης γνώσης-Data Mining

Διάφορα ζητήματα προκύπτουν από το θέμα της συνεχούς αύξηση ποσότητας δεδομένων. Αν δεν γίνει η κατάλληλη διαχείριση, υπάρχει κίνδυνος να χαθεί αξιόλογο περιεχόμενο. Δηλαδή, γνώση ικανή να βοηθήσει πολλούς τομείς στη κοινωνία, δε θα μπορούσε να χρησιμοποιηθεί. Συνεπώς τα αρχικά δεδομένα θα καθιστούν άχρηστα. Αυτό το φλέγον ζήτημα καλείται να επιλύσει η εξόρυξη δεδομένων (data mining). Μέγιστη προτεραιότητα της είναι να γίνει εύρεση υποδειγμάτων που εμφανίζονται πολλές φορές. Αυτό μπορεί να πραγματοποιηθεί και μέσα σε βάσεις. Διότι από αυτά μπορεί να εντοπιστεί κάτι ωφέλιμο. Αρκετές φορές μάλιστα, η εύρεση των συγκεκριμένων δεν είναι τόσο απλή διαδικασία, καθώς μπορεί να μην είναι εύκολα επιτεύξιμη απόσπαση τους. Ωστόσο, το σημαντικότερο τμήμα δεν αποτελεί μόνο η εύρεση τέτοιων στοιχείων μέσα από τα δεδομένα, αλλά και το να γίνει η κατάλληλη εκμετάλλευση τους, ώστε να προκύψει κάτι καλό από αυτά. Ένα χαρακτηριστικό παράδειγμα αυτού, αποτελεί τη δυνατότητα εκτίμησης τιμών που ακολουθούν. Σε αυτό το κομμάτι βοηθάει πολύ η εξόρυξη δεδομένων. Ουσιαστικά στο να παράγεται ποιοτική γνώση από τα δεδομένα. Διότι η γνώση είναι δύναμη και είναι ικανή να δώσει μεγάλη ισχύ σε οποιοδήποτε μέλος της κοινωνίας, να του δώσει σημαντικό προτέρημα. Αυτό είναι κάτι που χρειάζεται σε αρκετούς τομείς. Όπως για παράδειγμα στις επιχειρήσεις. Βέβαια, λόγω της ραγδαίας εξέλιξης της τεχνολογίας, πλέον η λειτουργία αυτή μπορεί να επιτευχθεί εξ ’ολοκλήρου και χωρίς την επέμβαση ανθρώπινου παράγοντα. Εννοείται όμως ότι υπάρχει και ο τρόπος που είναι κάπου στη μέση, όπου συμβάλλει τόσο ο άνθρωπος όσο και το σύστημα.[10] Όσο για την υλοποίηση της εξόρυξης, στηρίζεται πάνω σε διάφορες μεθόδους. Αυτές χρησιμοποιούνται στους τομείς της μηχανικής μάθησης και μαθηματικών. [11]

Όταν γίνεται αναφορά στην εξαγωγή γνώσης, δεν πρέπει να παραλειφθεί οτι συγκεκριμένα, στα σύνολα δεδομένων, παρατηρούνται συνήθως κάποια στοιχεία που ονομάζονται περιπτώσεις (instances). Βέβαια, κάθε μια είναι το αποτέλεσμα συσχετίσεων μεταξύ των ποικίλων γνωρισμάτων που διατίθενται. Μάλιστα, αυτά δεν είναι αναγκαίο να περιέχουν αποκλειστικά αριθμούς, καθώς μπορεί να περιλαμβάνουν και γράμματα. Ακόμη, ανάλογα με το μέγεθος της εκάστοτε συλλογής μπορούν να γίνουν και αντίστοιχες συνθέσεις. Στο τέλος στόχος είναι να επιτευχθεί η εκτίμηση. Αυτή είναι δυνατή χάρη στους λεγόμενους κανόνες (rules), οι οποίοι προκύπτουν μέσα από τις περιπτώσεις. Ανάλογα με το τι είδους πρόγνωση κάνουν, διακρίνονται σε δύο κατηγορίες:

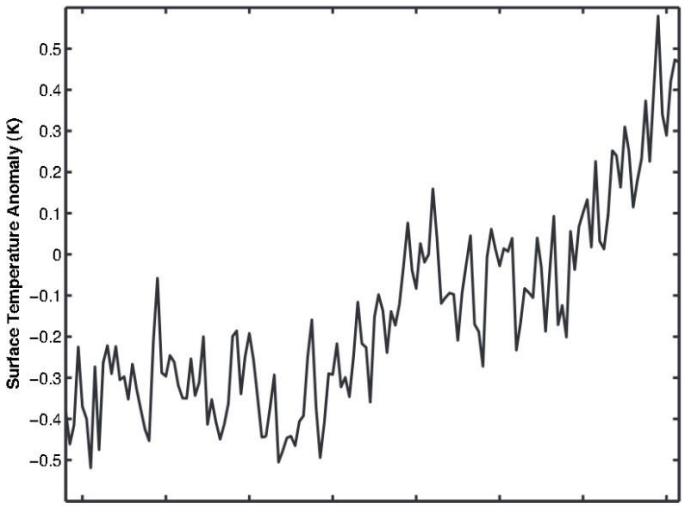
- κανόνες ταξινόμησης(classification rules). Η ικανότητα τους περιορίζεται στην εκτίμηση μιας μόνο κλάσης.
- κανόνες συσχέτισης(association rules).Το κάνουν για ένα μεγάλο αριθμό γνωρισμάτων. [10]

Πέρα από τους κανόνες συσχέτισης και ταξινόμησης, στην εξόρυξη δεδομένων υπάρχει και η πρόγνωση (Forecasting) η οποία γίνεται από ένα πόρισμα που βγαίνει με βάση τα υπάρχοντα δεδομένα. Επιπλέον, μέσα σε αυτά συγκαταλέγεται και η ανάλυση ακολουθιών (Sequence analysis), όπου στηρίζεται στην άμεση σχέση ενός συμβάντος με αυτό που έπεται ακριβώς μετά από αυτό χρονικά [11]

1.2 Εξόρυξη γνώσης με τη χρήση χρονοσειρών

Επειδή μια χρονοσειρά υλοποιείται μέσα σε ένα διάστημα, είναι λογικό το να έχει δημιουργηθεί μια πορεία. Παρακολουθώντας το πώς πάει λοιπόν αυτή, είναι δυνατόν να προκύψουν κάποια συμπεράσματα. Ανάμεσα σε αυτά είναι και η δυνατότητα ανίχνευσης υποδειγμάτων με συχνή εμφάνιση. Αυτό συγκαταλέγεται μέσα στις διάφορες ευθύνες που έχει αναλάβει η εξόρυξη. [9] Μάλιστα, από την πορεία αυτή που διαγράφει η χρονική σειρά, μπορούν να εξαχθούν και κάποιες ιδιότητες της. Για παράδειγμα, αν έχουν παρουσιαστεί μέσα σε διαστήματα που ορίστηκαν με ίδιο μέγεθος, επαναλήψεις ενός προτύπου, τότε υπάρχει περιοδικότητα (periodicity). Μερικές φορές, δεν μπορεί να ελεγχθεί τι έρχεται. Στην πραγματική ζωή, τα διαστήματα αυτά μπορεί να αλλάξουν. Αυτό γίνεται εξαιτίας μερικών εξωτερικών στοιχείων που παίζουν μεγάλο ρόλο στη μορφή του χρονικού πλαισίου. Με βάση αυτά προκύπτουν οι όροι κυκλικότητα (ciclicity) και εποχικότητα (seasonality). Χάρη στο seasonality είναι επιτεύξιμη η εκτίμηση της μελλοντικής κατάστασης των δεδομένων της χρονοσειράς. [37]

Ένα ακόμη γνώρισμα που παρατηρείται από την εξέλιξη μιας χρονοσειράς είναι η τάση (trend). Από την άλλη, η απαλοιφή της από τα δεδομένα λέγεται αφαίρεση τάσης ή αλλιώς detrending. Η τάση είναι μια λειτουργία, στην οποία η χρονοσειρά παρουσιάζει δύο διαφορετικές συμπεριφορές. Αυτή της αύξησης ή αυτής της μείωσης. Δηλαδή, θα γίνεται είτε το ένα είτε το άλλο. Ουσιαστικά δεν είναι τίποτα άλλο πέρα από ένα μοτίβο που φαίνεται ότι παρουσιάζει μια συγκεκριμένη πορεία μέσα στον χρόνο. Έχουν και τα δύο ποικίλες υλοποιήσεις και μπορούν να χρησιμοποιηθούν σε αρκετά πράγματα. Όπως για παράδειγμα, η εύρεση της έχει μεγάλη λειτουργικότητα σε σύνολα που αφορούν τον καιρό, όπου γίνεται η ανάδειξη της φυσικής μεταβολής των δεδομένων μέσα σε ένα χρονικό διάστημα. Από την άλλη, η αφαίρεση της, έχει μεγάλη ισχύ ιδιαίτερα στον κλάδο της ανάλυσης του φάσματος. Επειδή σε αυτή την περίπτωση, είναι καίριο ζήτημα να μείνει ανέγγιχτο το σύνολο δεδομένων από την τάση. Σε γενικά πλαίσια υπάρχουν διάφορα είδη τάσης. Δύο από αυτές που θα αναφερθούν στην πορεία, είναι εκείνης του μη σταθερού μέσου όρου και αυτής που χρησιμοποιείται αρκετά. Η πρώτη προκύπτει όταν σε μια χρονοσειρά εφαρμόζεται ο μη σταθερός μέσος όρος.. Όσο για τη δεύτερη, δεν είναι ιδανική σαν εκδοχή, ειδικά όταν τα δεδομένα μεταβάλλονται συνεχώς. Προκύπτει κάνοντας την υπόθεση ότι η γραμμή που φαίνεται να τέμνει τα δεδομένα δεν μεταβάλλεται ούτε και αλλάζει κλίση.[36]



Σχήμα 1.2.1.Η αναπαράσταση του GSTA [36]

Στο σχήμα 1.2.1 είναι ευδιάκριτη η φυσική ανοδική εξέλιξη που έχουν πάρει τα δεδομένα. Απ' ότι φαίνεται είναι αρκετά πιθανό στη συνέχεια αυτή να κινείται με τον ίδιο τρόπο. Σε αυτή την περίπτωση, είναι δυνατόν να παρατηρηθεί μια τάση. Μάλιστα, στον άξονα y'y φαίνεται η τιμή που έχει μετρηθεί μέσα σε μια συγκεκριμένη χρονική περίοδο. Αυτή είναι πολύ συγκεκριμένη, περιλαμβάνει χρονιές από το 1961 μέχρι και το 1990. Μέσα σε αυτό το διάστημα καταγράφεται το πόσο μακριά είναι από τη μέση τιμή της θερμοκρασίας της επιφάνειας της γης. Για να το πετύχει αυτό, δεν λαμβάνει όλες τις τιμές των θερμοκρασιών. Αντίθετα, χρησιμοποιεί μόνο όσες παρεκκλίνουν. Στον άξονα χ'χ είναι το χρονικό διάστημα μέσα στο οποίο πραγματοποιήθηκε η μελέτη αυτή. Σε αυτή την περίπτωση είναι από το 1856 μέχρι και το 2003. [36]

Σε γενικές γραμμές, τα πορίσματα που μπορούν να αντληθούν από την πορεία της σειράς όσον αφορά τις ιδιότητες της είναι τα εξής:

Μεταβλητότητα (Variability). Δείχνει το πως μεταβάλλονται τα στοιχεία μιας χρονοσειράς. Πιο συγκεκριμένα, από τι μπορεί να επηρεαστούν. Λίγο πιο αναλυτικά, διακρίνονται δύο κύριοι παράγοντες που φαίνεται ότι σχετίζονται άμεσα με την μεταβλητότητα. Οι πρώτοι είναι οι κύριοι. Αναφορικά με τους δεύτερους, μέσα στη διάρκεια του χρόνου δεν είναι σταθεροί. Σημαντικό είναι το να γίνει αξιολόγηση του πόσο καλά λειτουργεί ο καθένας. Αυτό θα επιτευχθεί από τη παρατήρηση των χρονοσειρών.[37]

Ομοιογένεια (Homogeneity). Από την μέτρηση τιμών ή στοιχείων μέσα σε ένα διάστημα, προκύπτει μια χρονοσειρά. Όταν μέσα σε αυτό οι τιμές είναι όλες της ίδιας κατηγορίας, τότε υπάρχει αυτό που είναι γνωστό ως ομοιογένεια. Αυτό συμβαίνει για παράδειγμα όταν όλες οι ξεχωριστές παρατηρήσεις μετριούνται σε βαθμούς κελσίου, μέτρα, εκατοστά κτλ. Πρέπει δηλαδή όλοι να ανήκουν στην ίδια οικογένεια. Αυτή η ιδιότητα δεν είναι κάτι σίγουρο ότι θα υπάρχει στη χρονοσειρά. Συνεπώς είναι υψίστης σημασίας να γίνεται πρώτα ένας τυπικός έλεγχος της προτού γίνει η οποιαδήποτε διερεύνηση. [37]

Περιοδικότητα (Periodicity). Αυτό το γνώρισμα υπάρχει όταν μέσα σε ίδια χρονικά πλαίσια, παρατηρούνται ξανά σε μεγάλη συχνότητα στοιχεία της χρονοσειράς. Το μέγεθος αυτού του πλαισίου εξαρτάται από το τι είναι αυτό που αποτυπώνεται. Εννοείται πως δεν είναι όλες οι περιστάσεις ίδιες. Όταν η ποσότητα αυτή είναι το ακαθάριστο εγχώριο προϊόν, είναι επιτακτική η εφαρμογή ενός σχετικά εκτενές πλαισίου. Από την άλλη, υπάρχουν άλλες όπως στην καταγραφη ισοτιμίας που δεν χρειάζεται καθόλου ένα τέτοιο αντίστοιχο μέγεθος, καθώς δεν προσφέρει κάτι. Οι δύο όροι που αναφέρθηκαν πριν, εκείνος της εποχικότητας (periodicity) και της κυκλικότητας (cyclicity) είναι αρκετά σημαντικοί. Καθένα σχετίζεται με ένα διαφορετικό μέγεθος και μορφή αντίκτυπου που έχουν σε μια χρονοσειρά. Εκ των οποίων, το πρώτο όπως εύλογα προκύπτει συμπερασματικά από την ονομασία, εξαρτάται από καθορισμένες χρονικές περιόδους. Χαρακτηριστικό παράδειγμα εποχικότητας που παρατηρούνται σε χρονοσειρές, αποτελεί ο τουρισμός όπου σε κάποια διαστήματα μεταβάλλεται. Από την άλλη, το δευτερο σχετίζεται άμεσα με αιτίες οικονομικού τύπου, που δεν έχουν κάποιο συγκεκριμένο διάστημα εμφάνισης. [37]

Αλληλεξάρτηση (Interdependence). Το συγκεκριμένο χαρακτηριστικό βασίζεται στη ιδιότητα των παρατηρήσεων να συνδέονται χρονικά μεταξύ τους. Αυτό είναι απόλυτα λογικό, καθώς το στοιχείο

για το οποίο μετρώνται δεδομένα μέσα σε μια χρονοσειρά παραμένει σταθερό.Στην ουσία δεν είναι ανεξάρτητα μεταξύ τους. Αντιθέτως, είναι σαν να είναι συνδεδεμένα με μια νοητή αλυσίδα.[37]

Εξαιτίας του ζητήματος ραγδαίας αύξησης των μεγέθους των δεδομένων και των ποικίλων συνόλων από όπου αυτά μπορούν να αντληθούν, είναι απαραίτητο να ανταπεξέλθουν ανάλογα στις απαιτήσεις οι μέθοδοι ανάλυσης. Προκύπτουν κάποιες μικρές ανησυχίες, όσον αφορά την αντιμετώπιση και προσέγγιση αυτού. Μάλιστα είναι επιτακτική η συζήτηση αυτών, εξαιτίας αυτού του καίριου προβλήματος.. Πιο συγκεκριμένα, είναι σημαντική η εύρεση τρόπου απεικόνισης των χρονοσειρών που να καταφέρει δύο πράγματα. Πρώτων, να διατηρήσει την αρχική πληροφορία με όση ακρίβεια γίνεται. Δεύτερον, να επιτύγχει στο να κάνει το μέγεθος μικρότερο. Εξίσου ιδιαίτερο σε αξία, είναι η ύπαρξη ενός μέτρου το οποίο θα καταγράφει το μέγεθος της συσχέτισης ανάμεσα σε οντότητες. Ουσιαστικά το πόσο κοντά οι μακριά είναι. Κάτι σημαντικό που πρέπει να τονιστεί είναι ότι αυτό δε θα στηρίζεται αποκλειστικά στην κλασική έννοια της απόστασης. Τέλος, με βεβαιότητα μπορεί να πει κανείς ότι είναι απαραίτητη μια μέθοδος που να βοηθάει στην εύκολη εύρεση οποιασδήποτε πληροφορίας θέλει κάποιος.[9]

1.2.1 Ορισμός χρονοσειράς

Αν γίνει η υπόθεση ότι έχουμε ένα καθορισμένο χρονικό διάστημα μέσα στο οποίο οι τιμές του χρόνου είναι συγκεκριμένες και πραγματοποιηθεί μέσα σε αυτό καταγραφή μιας τιμής, τότε αυτό είναι μια χρονοσειρά. Στην ουσία κάθε μια από τις τιμές επέρχεται της άλλης.[3] Μάλιστα, ο τρόπος με τον οποίο γίνεται αυτό παίζει πολύ μεγάλο ρόλο, καθώς η μια σχετίζεται την άλλη.[4] Ακόμη, όταν γίνεται εγγραφή των μεγεθών της, υπάρχει μια συγκεκριμένη συχνότητα.[9] Μια χρονοσειρά ορίζεται με τον παρακάτω τρόπο:

Ως μια διατεταγμένη σειρά T που περιέχει n αριθμό παρατηρήσεων. Ο τύπος ισχύει για n να είναι ίσο με πραγματικούς αριθμούς. [9]

$$T = (t_1, \dots, t_n), t_i \in \mathbb{R} \text{ [9]} \qquad (1.2.1.1)$$

Μια χρονοσειρά μπορεί να χαρακτηριστεί με αρκετούς τρόπους. Τρεις από αυτούς είναι οι εξής:

- Όταν δεν σταματάει καθόλου η διέλευση των δεδομένων, μια χρονική σειρά μπορεί να θεωρηθεί επίσης και ως ήμι-άπειρη (semi-infinite).

- Ιδιαίτερο ενδιαφέρον παρουσιάζει το τι συμβαίνει, όταν σε ένα χρονικό πλαίσιο γίνεται η μέτρηση διαφόρων μεταβλητών. Συνεπώς, μέσα σε αυτό προκύπτουν αντίστοιχα χρονοσειρές αρκετές σε αριθμό. Το παραπάνω παρέχει τον χαρακτηρισμό μιας χρονοσειράς ως πολυμεταβλητή (multivariate).

- Αντίστοιχα, όταν αφορά μόνο μια μεταβλητή, θεωρείται μονομεταβλητή (univariate).

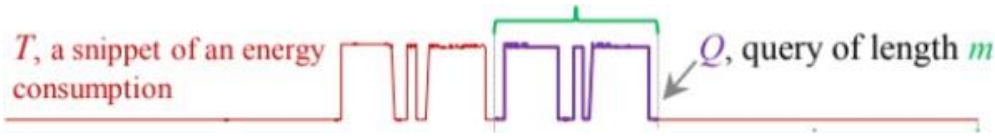
Γνωρίζοντας ότι σε αρκετές περιπτώσεις μπορούν να φτάσουν να έχουν μεγάλη έκταση, λόγω του ευρύ διαστήματος μέσα στο οποίο γίνεται η καταγραφή, είναι πιο εύκολη στη διαχείριση η διάσπαση της σε μικρότερα υποτμήματα. Όταν το πλήθος δεδομένων αυξάνεται, προκαλούνται διάφορα θέματα. Κάτι που συνηθίζεται σαν λύση σε αυτό είναι η αποφυγή αποθήκευσης της πραγματικής χρονοσειράς

σε βάσεις δεδομένων. Κάτι τέτοιο θα είχε παραπάνω πολυπλοκότητα. Αντί γι' αυτό, για την αποθήκευση χρησιμοποιείται η πιο βασική έκδοση της πραγματικής χρονοσειράς. [9]

Ένα μικρότερο τμήμα της T, αποτελεί η υπακολουθία S. Το μέγεθος της συμβολίζεται με το m και είναι πάντα μικρότερο από το αυτό της σειράς, δηλαδή το n. Όπως και για την αρχική χρονοσειρά, έτσι και στην S δεν σταματάει η καταγραφή τιμών.[9]

$$S = (t_k, t_{k+1}, \dots, t_{k+m-1})$$ [9] (1.2.1.2)

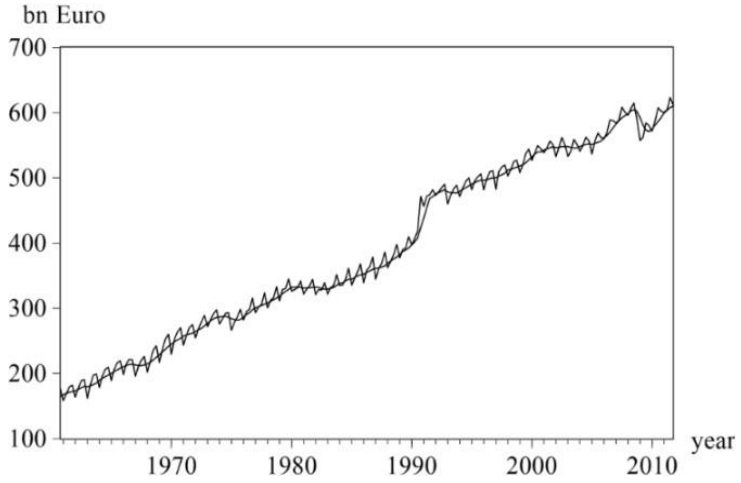
Ο τύπος ισχύει για $1 \leq k \leq n - m+1$ [9]



Σχήμα 1.2.1.1 Αναπαράσταση υποακολουθίας χρονοσειράς Q [8]

Στο σχήμα 1.2.1.1 η χρονοσειρά T μετράει το ποσό της ενεργειακής δαπάνης και η Q αποτελεί την υποσειρά. [8]

Πέρα από τα γνωρίσματα που αναφέρθηκαν προηγουμένως, όπως ήμι-άπειρη, μονομεταβλητή και πολυμεταβλητή, υπάρχουν και άλλοι τρόποι χαρακτηρισμού μιας σειράς. Πιο συγκεκριμένα, ανάλογα με τον ρυθμό χρόνου στον οποίο πραγματοποιείται η μέτρηση των δεδομένων μιας χρονοσειράς, προκύπτουν ενδιαφέροντα γνωρίσματα, όπως είναι η συνεχής και η διακριτή. Αυτές είναι που χαρακτηρίζουν μια χρονοσειρά. Μάλιστα, η πρώτη απορρέει όταν δεν υπάρχει παύση στη μέτρηση, ενώ η δεύτερη όταν ο ρυθμός αυτός είναι δεν αλλάζει. [5]



Σχήμα 1.2.1.2 Αναπαράσταση οικονομικής χρονοσειράς [3]

Στο σχήμα 1.2.1.2 φαίνεται με τη μορφή των δις. ευρώ από το χρονικό διάστημα 1961 με 2011 μια χρονοσειρά που δείχνει το Ακαθάριστο Εγχώριο Προϊόν(ΑΕΠ) της Γερμανίας. Στον άξονα x'x βρίσκεται το διάστημα μέσα στο οποίο έγινε η καταγραφή των δεδομένων, ενώ στον y'y μετριέται η ποσότητα του Ακαθάριστου Εγχώριου Προϊόντος σε δισεκατομμύρια ευρώ. [3]

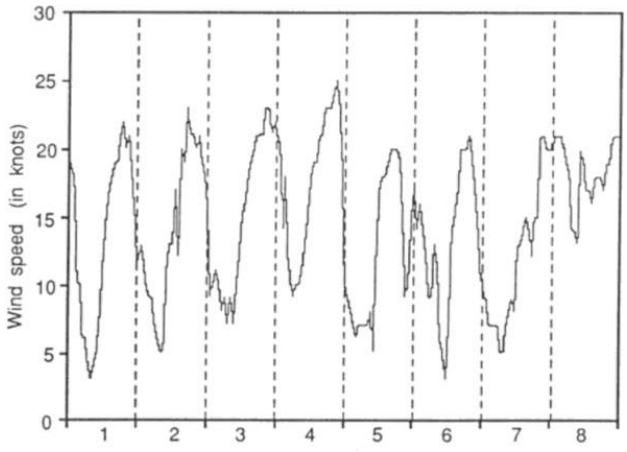
Μια πρόταση που προσπαθεί να αντιμετωπίσει το πρόβλημα της συνεχής αύξησης της πληροφορίας, στηρίζει ότι αντί να χρησιμοποιούνται τα δεδομένα αυτά έτσι όπως είναι, μπορεί να γίνει προσήλωση σε μικρότερα σύνολα. Το παραπάνω επιτελείται σαν εργασία μέσω ενός πλαίσιο(framework). Μάλιστα, σε αρκετές περιπτώσεις δεν χρειάζεται μόνο ένα, άλλα πολλά. Καθένα από αυτά είναι ικανό να διαχειριστεί όλο αυτό τον όγκο πληροφορίας.[7]

1.2.2 Εφαρμογές και χρησιμότητα

Οι χρονοσειρές έχουν την ευχέρεια να υλοποιηθούν σε πολλούς τομείς. Κατά κόρον υπάρχει μεγάλη εφαρμογή τους στον τομέα της οικονομετρικής χάρη στη δυνατότητα χρήσης τεχνικών στατιστικού χαρακτήρα για τη συσχέτιση των οικονομικών δεδομένων με τέτοιο τρόπο ώστε να παρθούν χρήσιμα πορίσματα.[3] Εντοπίζονται διάφοροι αξιόλογοι τρόποι με τους οποίους μπορούν να ασκήσουν βοηθητική επίδραση στον άνθρωπο και σε ό,τι έχει να κάνει με την κοινωνία. Όπως για παράδειγμα, για κάποιον μπορεί να είναι σημαντικό το να ενημερώνεται για τον καιρό και την εξέλιξη του, ενώ για άλλον το να δει αν είναι καλά ή όχι μέσω ενός ηλεκτροκαρδιογραφήματος. Ακόμη, μερικοί τις θεωρούν χρήσιμες καθώς τους παρέχει διευκόλυνση όταν πάνε να ελέγξουν το πόσο έχουν ξοδέψει. Σε άλλες περιπτώσεις φαίνεται χρήσιμο για εκείνους που θέλουν να ελέγχουν την ποιότητα ύπνου τους. Επιπλέον σε λίγο πιο μεγαλύτερα πλαίσια παρέχει τη δυνατότητα πληροφόρησης για το δημογραφικό. Επιπρόσθετα, οι εκάστοτε εταιρείες μπορούν να τις αξιοποιήσουν κατάλληλα ώστε να δουν που βρίσκονται στο σήμερα. Εν πάση περίπτωση, αν αξιοποιηθεί σωστά το ιστορικό των χρονικών σειρών, θα προκύψει εκτίμηση του πως θα εξελιχθούν μακροπρόθεσμα. Πέρα από αυτό, η ενδελεχής μελέτη τους μπορεί να αναδείξει κάτι ενδιαφέρον σαν γνώση για τον τρόπο που προέκυψαν.[4]

Σε γενικές γραμμές, ιδιαίτερο ενδιαφέρον αποκτά η χρήση τους όσον αφορά την διεξοδική διερεύνηση του πως δρα ένας άνθρωπος. Γι’ αυτό το λόγο, κατάλληλοι ειδικοί εξάγουν με επιτυχία τα δεδομένα που παίρνουν από συγκεκριμένα σημεία συλλογής. Μάλιστα, αυτά τα δεδομένα αφορούν κυρίως διάφορες ανθρώπινες καταστάσεις και παρουσιάζουν κοινωνικό χαρακτήρα. Όπως ο αριθμός των ανθρώπινων σκοτωμών, αυτοκτονιών ή και των ατόμων που παίρνουν διαζύγιο. Όλα τα προαναφερόμενα βοηθάνε τους ειδικούς στην ανάλυση τους. Μεγάλη εφαρμογή γίνεται και στη έρευνα που περιλαμβάνει έναν επόπτη. Αρκετές φορές στο περιβάλλον μιας τάξης ένας έχει τον ρόλο του επιβλέπων. Αυτός επιβλέπει την κατάσταση και σε περίπτωση που υπάρξει κάποια αντίδραση καλείται να την καταγράψει ανά κάποια πλαίσια χρόνου, όπως για παράδειγμα 5 λεπτά. Επίσης, πολύ μεγάλη βοήθεια παρέχουν και στον κλάδο της ψυχολογίας. Επιτρέπουν σε έναν επιστήμονα μέσα στο βάθος των χρόνων να παρακολουθεί τα άτομα και την κατάσταση τους ανά κάθε σταθερό διάστημα. Με αυτό τον τρόπο, θα μπορέσει μετέπειτα να έχει μια πολύ καλή εικόνα για την γενική εξέλιξη τους .[39]

Ιδιαίτερο ενδιαφέρον αποκτά η μεγάλη λειτουργία των χρονοσειρών στην εξαγωγή συμπερασμάτων, όπως στην μελέτη καιρικών φαινομένων. Πιο συγκεκριμένα, τόσο με την μελέτη της πορείας κατά την οποία φυσάει ο αέρας, αλλά και το πόσο γρήγορα, είναι εφικτή η εύρεση στοιχείων που επαναλαμβάνονται μέσα στη σειρά. [1]



Σχήμα 1.2.2.1 Αναπαράσταση ταχύτητας αέρα [1]

Στην εικόνα 1.2.2.1 γίνεται η μέτρηση της ποσότητας ταχύτητας του ανέμου σε κόμβοι μέσα στο χρονικό διάστημα 8 ημερών, δηλαδή από τις 1 Δεκεμβρίου 1971 μέχρι και τις 8 Δεκεμβρίου. Αυτή η έρευνα πραγματοποιήθηκε στην Δυτική Αυστραλία. Πιο συγκεκριμένα, στο λιμάνι Fremantle. [1]

Πολύ σημαντική σε αξία είναι επίσης και η χρήση των χρονοσειρών για την μέτρηση στοιχείων που αφορούν τον κλάδο της υγείας και ό,τι έχει να κάνει με την επίβλεψη αυτής. Δηλαδή, τα δεδομένα που εξάγονται χρησιμοποιούνται κυρίως τόσο για την αποφυγή μελλοντικών δυσάρεστων καταστάσεων όσο και για την πρόβλεψη τους. Μάλιστα, τα συστήματα που σχετίζονται άμεσα με αυτόν τον κλάδο, έχουν ρόλο να καταγράφουν το οτιδήποτε θεωρηθεί περίεργο στο πως κινείται η εξέλιξη μιας ασθένειας. [2]

1.3 Παρουσίαση του Matrix Profile και της κατηγοριοποίησης

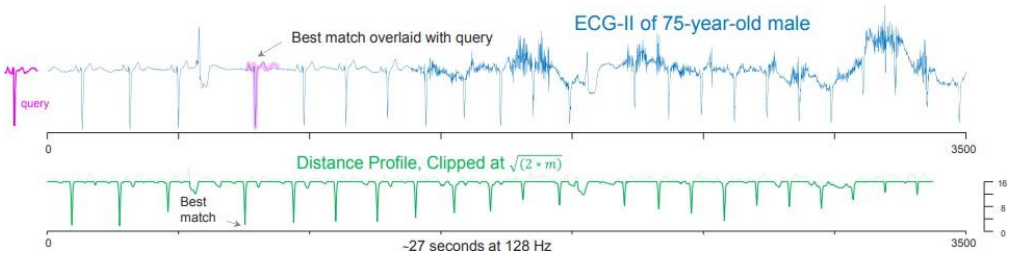
Το κλειδί στην εμβάθυνση μιας χρονοσειράς σε μεγαλύτερο βαθμό, προκύπτει από την παρατήρηση της απόστασης των υποακολουθιών που είναι μέσα σε χρονοσειρές. Αναλυτικότερα, ανάλογα με το πόσο μεγάλη η μικρή είναι η απόσταση μεταξύ τους, προκύπτουν κάποια σημαντικά στοιχεία, εκ των οποίων η μια ονομάζεται μοτίβο (Motif) και η άλλη ασυμφωνία (Discord). Στην περίπτωση που η απόσταση τους είναι μικρή, τότε υπάρχει μοτίβο. Από την άλλη, όταν αυτή είναι μεγάλη, ακόμα και από τις υποακολουθίες που βρίσκονται πιο κοντά, τότε προκύπτει η ασυμφωνία.

Ωστόσο, πρέπει να αναφερθεί το σημαντικό εργαλείο που βοηθάει στην εύρεση των μοτίβων και των ασυμφωνιών, γνωστό και ως Matrix Profile. Προκειμένου να επιτευχθεί βαθύτερη κατανόηση των υποσειρών, είναι καλή τακτική η εισαγωγή περαιτέρω πληροφοριών γι’ αυτές. Δηλαδή, στοιχεία που αφορούν κυρίως τη συμπεριφορά τους. Ουσιαστικά λειτουργούν σαν ετικέτες. Ένα παράδειγμα αυτού υλοποιείται από το Matrix Profile, καθώς μέσα σε αυτό περιλαμβάνονται στοιχεία που δείχνουν το πόσο όμοιες είναι οι υποσειρές μεταξύ τους. Πιο συγκεκριμένα, παίρνονται αυτές με τη μικρότερη απόσταση. Μάλιστα, αυτό έχει ωφέλιμη αξία στο κομμάτι της ανάλυσης. [13]

Ο χώρος μέσα στον οποίο περιλαμβάνονται όλες τις τιμές που δείχνουν το πόσο απέχουν όλες οι μικρότερες ακολουθίες μεταξύ δύο σειρών ονομάζεται προφίλ αποστάσεων (Distance profile). Μάλιστα αυτές της δεύτερης σειράς παίζουν το ρόλο του ερωτήματος (query). Μέσα σε αυτόν

αποθηκεύονται οι Ευκλείδειες αποστάσεις με το που βρεθούν. Χρησιμοποιώντας τον αλγόριθμο MASS η εκτέλεση του γίνεται γρηγορότερα.[13]

Στο σχήμα 1.3.1 χρησιμοποιούνται δεδομένα που πηγάζουν από την μέτρηση του ενός ηλεκτροκαρδιογραφήματος ECG-II, που αντλήθηκε από έναν άντρα 75 χρονών. Από ότι φαίνεται στο διάγραμμα με μπλε χρώμα έχει χρησιμοποιηθεί μια υποακολουθία που λειτουργεί σαν ερώτημα. Μάλιστα, η υποακολουθία αυτή δεν μένει σταθερή συνέχεια, αλλά μετακινείται. Στο πράσινο διάγραμμα από κάτω είναι ευδιάκριτο το τελικό αποτέλεσμα, δηλαδή το προφιλ αποστάσεων, στο οποίο μάλιστα διακρίνονται και κάποιες τιμές λίγο πιο χαμηλές σε ύψος, οι οποίες είναι αυτές που συνήθως υποδηλώνει την ύπαρξη μοτίβων. Η μικρότερη από αυτές τις αποστάσεις θεωρείται η καλύτερη επιλογή. Όντως, αν τοποθετήσει κανείς το καλύτερο ταίριασμα από το προφίλ αποστάσεων και το βάλει πάνω στο πρώτο διάγραμμα, γίνεται εμφανές ότι είναι πολύ όμοια με το αρχικό ερώτημα. [13]



Σχήμα 1.3.1 Αναπαράσταση Distance Profile μιας χρονοσειράς ECG-II [13]

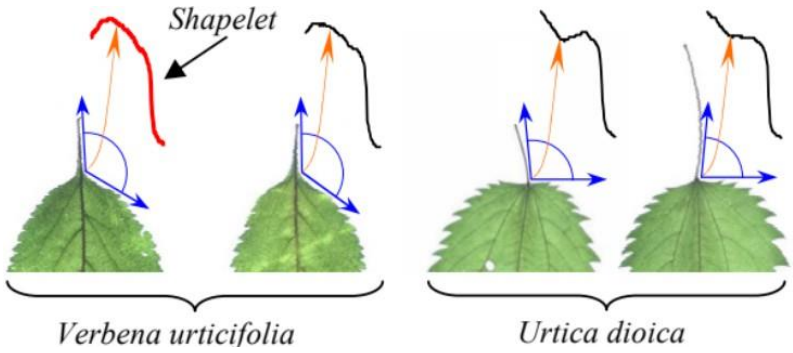
Στην πραγματικότητα δεν υπάρχει κάποιος περιορισμός όσον αφορά το κομμάτι του ποιες χρονοσειρές θα χρησιμοποιηθούν ώστε να βρεθούν οι αποστάσεις στο Matrix Profile. Με βάση αυτό, προκύπτουν δύο έννοιες, η πρώτη αφορά μια διαδικασία η οποία λέγεται αυτό ένωση (self-join) και η δεύτερη ονομάζεται ένωση AB (AB-join). Η αυτοένωση, όπως προδίδει ελάχιστα και από την ονομασία της, προκύπτει από το γεγονός ότι η χρονοσειρά και οι υποακολουθίες που χρησιμοποιούνται για τον υπολογισμό του Matrix Profile προέρχονται όλα από την ίδια χρονοσειρά Ta.

Αντίθετα, στη δεύτερη κατηγορία, δηλαδή την ένωση AB, οι υποακολουθίες που χρησιμοποιούνται για την εύρεση της απόστασης δεν πηγάζουν μόνο από την ίδια χρονοσειρά Ta, αλλά και από μια άλλη χρονοσειρά Tb. Δηλαδή, ο τελικός πίνακας όπου θα αποθηκεύει μέσα του τις αποστάσεις μετά από τον υπολογισμό του Matrix Profile, θα περιλαμβάνει τις μικρότερες αποστάσεις που έχουν οι υποακολουθίες της Ta από αυτές στη Tb. Επίσης θεωρείται σημαντικό να αναφερθεί ότι οι αποστάσεις που προκύπτουν στο τελικό πίνακα υπολογίζονται από την Ευκλείδεια απόσταση. Αυτό ισχύει και για τα δύο προφίλ απόστασης. [13]

Είναι αδιαμφισβήτητό το γεγονός πώς η αντιμετώπιση θεμάτων όπως η πρόωρη ταυτοποίηση ασθενειών με βάση κατάλληλα δεδομένα, απαιτεί και την υλοποίηση κατηγοριοποίησης χρονοσειρών, που προσφέρει σημαντική βοήθεια. Παρ' όλα αυτά, υπάρχει ένας μικρό ζήτημα στο κομμάτι της υλοποίησης τους. Πιο συγκεκριμένα, στην εφαρμογή τεχνικών τεχνητής νοημοσύνης για την κατηγοριοποίηση, υπάρχει μια δυσκολία όσον αφορά τις πληροφορίες που παρέχουν τα εκάστοτε μοντέλα όσον αφορά την υλοποίηση τους. Παράγουν πολύ σημαντικά αποτελέσματα οι κατηγοριοποιητές που στηρίζονται σε αυτή, όπως τα βαθιά νευρωνικά δίκτυα (Deep Neural Networks). Παρ' όλα αυτά, οι ειδικοί που μπορεί να θέλουν να χρησιμοποιήσουν τα μοντέλα αυτά για

οποιοδήποτε σκοπό, δε θα έχουν εμπιστοσύνη στον μηχανισμό λόγω της έλλειψης εξηγήσεων ως προς το πως δουλεύει. Προκειμένου να είναι εφικτή η εμπιστοσύνη, χρειάζεται μια προσέγγιση γνώριμη με τον χρήστη, η οποία θα καλύπτει αυτό το κενό της έλλειψης εξήγησης και θα είναι αρκετά παρόμοια με την αντίληψη του. Συνεπώς, προτείνεται και μια άλλη πρόταση πολύ καλύτερη στην αντιμετώπιση αυτού του θέματος η οποία επιτυγχάνεται με την κατηγοριοποίηση με χρήση shapelets. [15]

Τα shapelets υιοθετούν τα ελαττώματα από την κατηγοριοποίηση με τη χρήση του πιο όμοιου γείτονα, δηλαδή την ανάγκη μεγάλης μνήμης και διαστήματος που παίρνει για την εκτέλεση. Πιο αναλυτικά, αποτελούν την εικόνα μιας ομάδας, καθώς είναι μικρά τμήματα χρονοσειράς που έχουν χαρακτηριστικά στοιχεία πάνω τους που είναι όμοια με των υπολοίπων στην κλάση. Αν το ζήτημα είναι να γίνει κατηγοριοποίηση όταν οι κλάσεις είναι φύλλα, πρέπει να προσδιοριστούν κάποια πράγματα. Αρχικά, είναι δύσκολο το να βασιστεί όλο πάνω στο σχήμα, καθώς τα δύο φύλλα μπορεί να αποκτήσουν μερικά όμοια χαρακτηριστικά λόγω φυσικών περιστατικών, όπως φάγωμα επιφάνειας από έντομα κτλ. Μια πιθανή λύση με ωστόσο πολύ κακή επίδοση, που προκύπτει κυρίως λόγω κάποιων περίεργων τιμών, είναι το να πάρει τη μορφή χρονοσειράς το αρχικό φύλλο και να γίνει με βάση αυτή η διαδικασία της κατηγοριοποίησης. Κάτι πολύ πιο αποδοτικό και πρακτικό είναι η χρήση shapelet, δηλαδή μιας υποακολουθίας χαρακτηριστικής για την κλάση στην οποία ανήκει.[16]



Σχήμα 1.3.2 Αναπαράσταση εύρεσης shapelet [16]

Στο σχήμα 1.3.2 υπάρχουν δυο είδη φύλλων και για τις δυο κλάσεις Verbena urticifolia και Urtica dioica. Όπως αναφέρθηκε, το κατάλληλο shapelet είναι αυτό που θα είναι παρόμοιο με τα στοιχεία της υπόλοιπη κλάσης, όμως που παράλληλα θα είναι και εντελώς διαφορετικό από εκείνα της άλλης κλάσης,. Έτσι και σε αυτή την περίπτωση αυτό που είναι σημειωμένο με κόκκινο, το Verbena urticifolia είναι όμοιο με το άλλο της κλάσης του, δηλαδή σχηματίζουν ίδια γωνία. Ωστόσο, παρατηρείται χαρακτηριστική διαφορά με την κλάση Urtica dioica όσον αφορά την γωνία που σχηματίζει το φύλλο, που είναι 90ο. Το αμέσως επόμενο στάδιο είναι το θέσιμο σε λειτουργία ενός απλού κατηγοριοποιητή δέντρου αποφάσεων. Για να επιτευχθεί αυτό, πρέπει πρώτα να εντοπιστεί η υποσειρά που είναι πιο όμοια με το shapelet. Το ψάξιμο γίνεται για κάθε υποσειρά μέσα στη βάση δεδομένων. Μόλις γίνει αυτό, μετριέται η απόσταση μεταξύ αυτής και του shapelet.[16]

1.4 Επίλογος

Μια ακολουθία στην οποία καταγράφονται τιμές μέσα σε ένα χρονικό διάστημα αποτελεί χρονοσείρα. Αναλύωντας αυτές επωφελούνται αρκετοί τομείς, όπως η μετεωρολογία, οικονομετρία, υγεία κτλ. Η εξόρυξη δεδομένων χρονοσειρών είναι αρκετά μεγάλης σημασίας. Χάρη σε αυτή είναι εφικτό να γίνουν κάποιες εργασίες όπως η κατηγοριοποίηση, η εύρεση κανώνων συσχετίσεων, η συστάδοποίηση, η πρόβλεψη και η αναλυση ακολουθιών. Ακόμη, μέσω αυτής μπορούν να εξαχθούν ενδιαφέροντα πορίσματα. Για παράδειγμα τα μοτίβα και τις ακραίες τιμές, δηλαδή οι ασυμφωνίες. Το πρώτο δείχνει την συχνή εμφνάνισης μιας υποσειράς στη χρονοσειρά. Από την άλλη το δεύτερο, παρουσιάζει αυτά που δεν είναι καθόλου κοινά με τα υπόλοιπα της σειράς, που ξεφεύγουν Το Matrix Profile είναι το πιο κατάλληλο εργαλείο για την εύρεση μοτίβων και ασυφμωνιών. Είναι ένας πίνακας που περιέχει όλες τις ελάχιστες αποστάσεις που προέκυψαν ανάμεσα σε χρονοσειρές. Για την εύρεση αυτών των τιμών χρησιμοποιήθηκαν οι υποσειρές τους. Επιπλέον, συσχετίζεται άμεσα με την εξαγωγή shapelets. Δηλαδή, υποσειρών που είναι χαρακτηριστικές για την κλάση τους και διαφορετικών από την άλλη. Τα παραπάνω εφαρμόζονται στην κατηγοριοποίηση. Αξιόλογα στοίχεια βγαίνουν παρατηρώντας τις οπτικές ιδιότητες μιας χρονικής σειράς. Πιο συγκεκριμένα,η μεταβλητότητα, η ομοιογένεια, η περιοδικότητα και η αλληλεξάρτηση. Όλα τα παραπάνω παρέχουν παραπάνω πληροφορίες για την χρονοσειρά.

Κεφάλαιο 2ο: Τεχνολογίες και Εργαλεία

2.1 Γλώσσα προγραμματισμού Python

Είναι αληθές ότι η εφαρμογή της Python εκτείνεται σε πολλούς και διαφορετικούς τομείς. Όπως για παράδειγμα την επίλυση ζητημάτων στην μηχανική μάθηση. Το γεγονός αυτό εξηγεί το όλο και αυξανόμενο ενδιαφέρον που υπάρχει προς αυτή. Επιπλέον, υλοποιείται ώστε να έρθουν σε μια συγκεκριμένη δομή τα δεδομένα. Μέσω αυτού, καθιστάται δυνατή η διερεύνηση τους. Το ίδιο ισχύει και για τις οποιεσδήποτε μεθόδους εφαρμοστούν για να το πετύχουν. Αυτό εννοείται είναι απαραίτητο αν θέλει κάποιος να διευκολύνει τη διαδικασία της διαχείρισης ή εκμετάλλευσης τους. Επίσης, ανήκει στην ευρεία οικογένεια των ερμηνευόμενων γλωσσών προγραμματισμού (interpreted languages). Χωρίς περιορισμούς, έχει την ικανότητα να υλοποιηθεί από σενάρια (scripts), αλλά και να βοηθήσει στην ανάπτυξη διαδικτυακών τόπων. Το τελευταίο επιτυγχάνεται από διάφορα εργαλεία και υπηρεσίες (frameworks). Μάλιστα, ένα πολύ γνωστό που φτιάχτηκε με τη υλοποίηση της είναι το γνωστό Django. Σε γενικά πλαίσια, ο κώδικας της είναι αρκετά πρακτικός στην εφαρμογή του. Επιπρόσθετα, δεν είναι δύσκολα κατανοητός. Ακόμη, παρέχει τη δυνατότητα της αναπαράστασης γραφημάτων. Χάρη σε αυτό, γίνεται αντιληπτή η επίδραση της στον κλάδο της ανάλυσης. Κάτι σημαντικό που δεν πρέπει να παραλειφθεί, είναι ότι καμία εφαρμογή δε θα ήταν επιτεύξιμη, χωρίς τη συμβολή των βιβλιοθηκών που χρησιμοποιεί. Μια άλλη, λίγο διαφορετική προσέγγιση της, είναι παίρνοντας τον ρόλο του συνδετικού κρίκου κομματιών από κώδικα που γίνεται εκμετάλλευση κυρίως από οργανισμούς και που υλοποιούνται σε FORTRAN ή C. Ωστόσο, προτιμάται είναι η προσαρμογή του σε C. Επειδή τακτική που αναφέρθηκε δεν είναι κάτι που συνιστάται.[25]

2.1.1 Βιβλιοθήκες Python για ανάλυση χρονοσειρών

Κατά κοινή ομολογία, η Python έχει ευρύ φάσμα λειτουργιών, τις οποίες τις επιτυγχάνει μέσω της χρήσης ποικίλων βιβλιοθηκών. Βέβαια, αν και η κάθε μια χρησιμοποιείται για διαφορετικούς σκοπούς, ο συνδυασμός πολλών είναι που φέρνει ιδιαίτερα αποτελέσματα. Όχι μόνο στην ανάλυση των δεδομένων, αλλά και για τις ανάγκες της παρούσας πτυχιακής εργασίας, γίνεται η χρήση τους. Μάλιστα, οι πιο συνήθεις από αυτές είναι:

NumPy. Σημαντικές υπηρεσίες και εργαλεία διατίθενται από αυτή. Όσον αφορά τις μαθηματικές καταγραφές, δίνει τα απαραίτητα για την υλοποίηση Fourier μετασχηματισμού. Το επόμενο αφορά τα στοιχεία που βρίσκονται μέσα σε μια μονάδα αποθήκευσης. Διαθέτει τη δυνατότητα άσκησης ενεργειών στα παραπάνω. Πέρα από αυτό, παρέχει και εναν ιδιαίτερο τύπο δεδομένων. Είναι κάτι εναλλακτικό αυτού που προσφέρει η Python. Δηλαδή τον πίνακα με όνομα ndarray. Διακρίνεται για τον μικρό χρόνο που παίρνει στις μετρήσεις να τρέξουν. Αυτό ισχύει και για όσες εργασίες χρειάζονται δεδομένα που είναι αριθμοί. Επίσης, για οποιαδήποτε πράξη μπορεί να πραγματοποιηθεί σε κάποιους πίνακες βοηθάει αρκετά. Προσφέρει τα κατάλληλα εφόδια που χρειάζονται για την υλοποίηση τους. Όλα αυτά που αναφέρθηκαν, έχουν κάτι κοινό. Χρειάζονται πολύ σε κλάδους που ασχολούνται με αριθμούς. Την προσπάθεια αυτή θέλει να στηρίξει η NumPy με τις διάφορες μεθόδους που διαθέτει. [25]

Pandas. Αποτελεί έναν συνδυασμό δύο λέξεων. Η πρώτη είναι αυτή της ανάλυσης. Όσο για τον δεύτερη, είναι αυτή του πάνελ δεδομένων (panel data). Εφαρμόζεται κατά κόρον από τον κλάδο που παρατηρεί και αναλύει διεξοδικά τα οικονομικά δεδομένα. Ακόμη, υλοποιείται σε έναν μεγάλο βαθμό από τις εταιρείες. Σαν βιβλιοθήκη, όπως η NumPy παρέχει το NumPy array, μια δική του μορφή δεδομένων, έτσι και η Pandas περιλαμβάνει το Series και το DataFrame, το οποίο υιοθετεί από την R. Οι διαφορές μεταξύ των δύο αφορούν κυρίως τις διαστάσεις. Το Series περιλαμβάνει μόνο μια στήλη. Από την άλλη, το DataFrame έχει περισσότερες από δύο διαστάσεις. Ωστόσο, το χαρακτηριστικό που είναι όμοιο και για τα δύο είναι ότι έχουν τη μορφή πίνακα. Κάτι αρκετά ενδιαφέρον, είναι ότι καταφέρνει με επιτυχία να πραγματοποιεί κάποιες αξιόλογες ενέργειες. Όπως αυτή της συνάθροιση στοιχείων (aggregation). Μια άλλη, αποτελεί την αναδιαμόρφωση (reshape). Υπάρχει και μια που περιλαμβάνει το να πάρει κάποιος ένα μικρότερο σύνολο από το αρχικό. Όλα αυτά είναι δυνατόν να συμβούν χάρη σε μια ιδιότητα της Pandas στον τρόπο με τον οποίο δουλεύει. Πιο συγκεκριμένα, στο γεγονός ότι από το Excel, την SQL και τη NumPy υιοθετεί στοιχεία. Πέρα από αυτά που αναφέρθηκαν αποτελεί ένα σημαντικό εργαλείο για την διαχείριση χρονοσειρών με κατάλληλο τρόπο που οδηγεί στην παραπάνω εμβάθυνση τους. Ακόμη, δεν επιβαρύνεται η λειτουργία του, σε περίπτωση που δεν είναι παρόντα όλα τα δεδομένα. Επίσης δεν χρειάζεται η δημιουργία πολλών διαφορετικών μοντέλων. Διότι για κάθε προέλευση δεδομένων, υπάρχει ίση αντιμετώπιση.

Matplotlib. Εννοείται ότι από το κομμάτι της ανάλυσης δεδομένων, δε θα μπορούσε να λείψει η δυνατότητα οπτικής αναπαράστασης τους. Το παραπάνω επιτυγχάνεται από τη συγκεκριμένη, καθώς πέρα από την κλασσική μορφή οπτικοποίησης, συμπεριλαμβάνει και αυτές που χρησιμοποιούν τον άξονα χ και y.

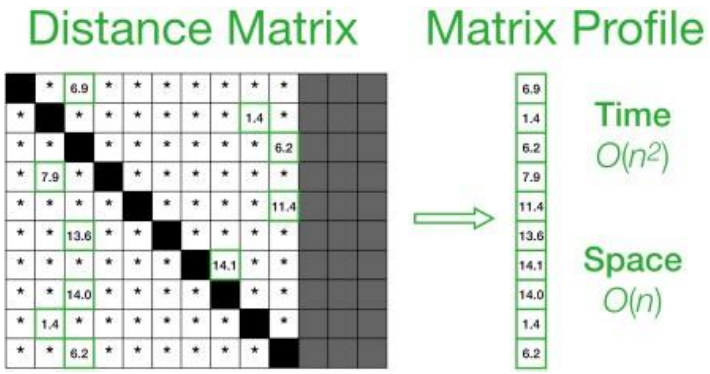
SciPy. Εντάσσεται κάτω από το ευρύ φάσμα της υλοποίησης μετρήσεων με σκοπό την αριθμητική ανάλυση. Γι' αυτό και αυτά που παρέχει βοηθούν αναλόγως στη διαχείριση τέτοιων ζητημάτων. Μάλιστα, κυμαίνονται από την διευκόλυνση ανάλυσης δεδομένων που προέρχονται από σήματα μέχρι και την παροχή βοήθειας στις εξισώσεις.

Scikit-learn. Η συγκεκριμένη παρέχει τις κατάλληλες τεχνικές ώστε να μπορέσουν να επιλυθούν οποιαδήποτε θέματα αφορούν τον κλάδο της μηχανικής μάθησης, αλλά μπορεί να επεκταθεί και σε ζητήματα της επιστήμης δεδομένων. Βέβαια, αυτό είναι επιτεύξιμο σε συνδυασμό με άλλες βιβλιοθήκες. Ο λόγος για τον οποίο εφαρμόζεται κυρίως σε τέτοιου είδους θέματα είναι εξαιτίας των δυνατοτήτων που παρέχει. Όπως για παράδειγμα, τεχνικές που κυμαίνονται από μεθόδους που ελλατώνουν το μέγεθος δεδομένων χρονοσειρών μέχρι και. την υλοποίηση κατηγοριοποίησης.

Statsmodels. Διαθέτει διάφορα εργαλεία που βοηθούν με τον υπολογισμό της διασποράς των δεδομένων μέχρι και την τελική απεικόνιση τους ως γραφήματα. Στην μελέτη οικονομικών δεδομένων, όλα αυτά βρίσκουν μεγάλη εφαρμογή. Τα παραπάνω είναι εφικτά χάρη στην statsmodels. Όπως υποδεικνύει και λίγο το πρώτο μέρος της λέξης, εντάσσεται στην κατηγορία βιβλιοθηκών που χρησιμοποιούνται κυρίως στην χρήση μεθόδων στατιστικής σε δεδομένα. [25]

STUMPY. Η βιβλιοθήκη αυτή χρησιμοποιείται και στην πειραματική ανάλυση για την κατηγοριοποίηση των χρονοσειρών σε παρακάτω ενότητα. Η συγκεκριμένη έρχεται να φέρει μια νεα αποδοτική πρόταση στο θέμα μελέτης χρονοσειρών και στον τρόπο που υλοποιείται το προφίλ πίνακα. Είναι υπερβολικά μεγάλος, ο χρόνος που χρειάζεται για την εύρεση συνάφειας μέσα στην ίδια χρονοσειρά από τον αλγόριθμο ωμής βίας. Συνεπώς, η διαδικασία επεξεργασίας είναι αρκετά

επιβαρυμένη. Η επιβάρυνση αυτή που φέρει ο συγκεκριμένος αλγόριθμος συμβολίζεται με $O(n^2m)$. Σε αυτή την περίπτωση, για να γίνει λίγο πιο κατανοητός ο τύπος, το n και το m δείχνουν το πλάτος της χρονοσειράς και της υποσειράς αντίστοιχα. Άμεση απόρροια αυτού είναι ότι όσο πιο πολύ αυξάνεται το n ή το m, τόσο πιο μεγάλη θα είναι και η συνολική πολυπλοκότητα. Στο σχήμα 2.1.1.1 παρουσιάζεται ένα παράδειγμα εύρεσης του Matrix Profile μέσω της STUMPY, χρησιμοποιώντας μια χρονοσειρά. [26]



Σχήμα 2.1.1.1 Ο υπολογισμός του Matrix Profile με χρήση της βιβλιοθήκης STUMPY [26]

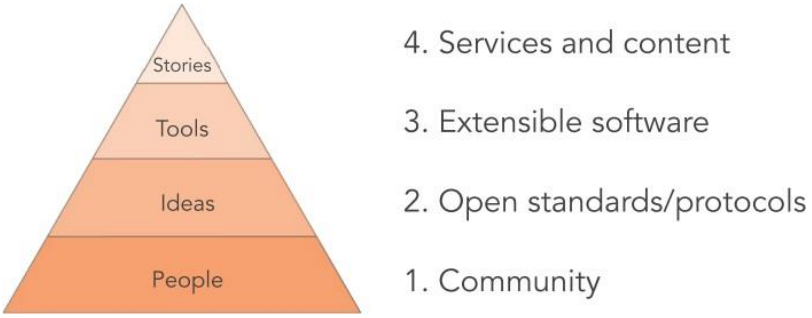
2.2 Εργαλείο Jupyter

Γνωστό και ως Jupyter, προήλθε αρχικά από το πρότζεκτ IPython (IPython project). Σαν γλώσσα προγραμματισμού, η Python διαθέτει έναν διερμηνέα. Μάλιστα, επιτελεί μια πολύ σημαντική λειτουργία. Βασισμένοι πάνω σε αυτό και βλέποντας πώς μπορεί να εξελιχθεί, προέκυψε το IPython. Δηλαδή, με αυτό τον τρόπο, ανέδειξαν παραπάνω τις δυνατότητες του κλασικού. Επειδή το Jupyter Notebook έπεται χρονικά από το IPython Notebook, αποτελεί εξέλιξη του. Μέσω του σημειωματαρίου (notebook), που είναι τύπου .ipynb, επιτελούνται. ποικίλες λειτουργίες. Όπως για παράδειγμα, την δυνατότητα που προσφέρει σε εφαρμογές με ανάλυση δεδομένων να τα απεικονίσουν. Μέσα σε όλα αυτά, καταφέρνει και συνδυάζει με επιτυχία το βασικό κομμάτι στην ανάπτυξη εφαρμογών, που είναι το γράψιμο κώδικα. Ακόμη, οι λειτουργίες που αναφέρθηκαν, κρατιούνται σαν πληροφορία στο τελικό .ipynb. Δηλαδή, όλα αυτά συνυπάρχουν μέσα στον ίδιο χώρο.[25] Κάτι αξιοσημείωτο και κύριο χαρακτηριστικό του Jupyter είναι ότι οι ενέργειες που γίνονται από κάποιον που γράφει κώδικα επηρεάζουν το notebook και αυτό αντίστοιχα ασκεί επίδραση στον τελικό χρήστη.Διέθετε περιορισμένες δυνατότητες, όταν βγήκε το Jupyter και ήταν ακόμη στο πρώιμο στάδιο. Συνεπώς, η συλλογή γλωσσών που υλοποιούνταν από αυτό δεν ήταν μεγάλη. Συγκεκριμένα, περιείχε μόνο την Python. Βέβαια, μέσα σε αυτή προστέθηκαν παραπάνω, όταν πέρασε λίγο ο καιρός,. [27]

2.2.1 Υπέρ και κατά του Jupyter

Ο ρόλος του Jupyter είναι ενός βοηθητικού εργαλείου, που κάνει τη δουλειά αυτού που το χρησιμοποιεί ευκολότερη. Αυτό ισχύει για τον οποιοδήποτε κώδικα εκτελέσει ή αναπαράσταση φτιάξει. Σε γενικές γραμμές, δείχνει ότι φαινομενικά δύσκολες διαδικασίες, μπορούν να γίνουν επιτυχώς. Γι' αυτό εξηγείται η υλοποίηση του σε διάφορους κλάδους. Στο Jupyter είναι άμεσα

συνυφασμένοι οι ρόλοι που σχετίζονται με την όλη διαδικασία υλοποίησης. Δεν είναι ξεχωριστοί και διακρίσιμοι. Αυτός που γράφει τον κώδικα και εκείνος που τον εκτελεί συγχωνεύονται σε έναν. Μια ιδιότητα του που ενισχύει την παραπάνω ιδέα είναι το γεγονός ότι αν κάποιος συντάξει ένα κομμάτι κώδικα, που περιλαμβάνει ένα στοιχείο από πριν, και το τρέξει, θα εμφανίσει το ανάλογο αποτέλεσμα εκείνη τη στιγμή. Είναι κάτι εφικτό, χάρη στο ιστορικό που διαθέτει με το τι συνέβη. Ουσιαστικά δεν χάνεται η πληροφορία που προκύπτει. Αντιθέτως, μπορεί να αξιοποιηθεί. Όλα αυτά που αναφέρθηκαν, μαζί με την δυνατότητα του να μπορεί να κοινοποιεί κάποιος το notebook του με άλλα άτομα, καθιστούν το Jupyter ένα παραπάνω από ικανό μέσο για την εφαρμογή εργασιών που αφορούν ποικίλους τομείς. Όπως για παράδειγμα στη μηχανική μάθηση. Σίγουρα είναι σημαντικό το γεγονός ότι παρέχει πολλές επιλογές όσον αφορά τη γλώσσα που θα χρησιμοποιηθεί για την εκάστοτε εργασία. Ένα ακόμα πιο ενδιαφέρον χαρακτηριστικό είναι ότι παρέχει την ελευθερία στον άνθρωπο να κινηθεί όπως αυτός θέλει, χωρίς δηλαδή να υπάρχει κάποιο μονοπάτι. Αφουγκράζεται το περιβάλλον και τις πληροφορίες που του παρέχονται. Μετά μπορεί να ενεργεί ανάλογα με το πως και πότε κρίνει σωστό. [27]



Σχήμα 2.2.1.1 Χαρακτηριστικά του Jupyter [27]

Όπως φαίνεται στο σχήμα 2.2.1.1, το Jupyter στηρίζεται πάνω σε τέσσερις κύριες αξίες, οι οποίες καθιστούν την πλατφόρμα αυτή αρκετά σημαντική. Μάλιστα, το σχήμα έχει τη μορφή πυραμίδας. Ξεκινάει από τους ανθρώπους και φτάνει μέχρι τον τελικό χρήστη. Το πιο κάτω κομμάτι είναι αφιερωμένο σε αυτούς που κατέβαλαν χρόνο και κόπο για να βοηθήσουν και να το εξελίξουν παραπάνω. Τίποτα από όλα αυτά δε θα υπήρχε χωρίς αυτούς. Το πάνω στρώμα δείχνει ότι ένας από τους λόγους που κατάφερε να εξελιχθεί τόσο, είναι χάρη στο ότι τα πρωτόκολλα που χρησιμοποιήθηκαν για την δημιουργία του δεν είναι περιορισμένης πρόσβασης. Αντίθετα, είναι δημόσια και μπορούν να τα δουν οι άνθρωποι, αλλά και να τα αξιοποιήσουν κατάλληλα. Με αυτό τον τρόπο προκύπτει το αμέσως επόμενο στάδιο, οπού ενθαρρύνονται εφαρμογές που να μπορούν να προσαρμοστούν εύκολα σε διάφορες καταστάσεις. Στο πιο ψηλό τμήμα της πυραμίδας παρουσιάζεται το βήμα, όπου είναι ο χρήστης και το πως αυτός το χρησιμοποιεί. [27]

Όπως και με οποιοδήποτε άλλο εργαλείο, έτσι και το Jupyter δεν αποτελείται μόνο από πλεονεκτήματα. Αντιθέτως, στο κομμάτι του κώδικα φαίνεται πως υστερεί αρκετά. Είναι αλήθεια ότι εύκολα μπορεί να μπερδευτεί κάποιος ως προς το πως διαβάζεται η δομή του κώδικα όσον αφορά το Jupyter Notebook. Για την εξέταση του παραπάνω γεγονότος, έγινε μια πειραματική έρευνα. Για να επιτευχθεί ο στόχος, χρησιμοποιείται η Python και εφαρμόζεται σε δύο διαφορετικά περιβάλλοντα. Με αυτό τον τρόπο υπάρχει μια ξεκάθαρη διάκριση ανάμεσα στα δύο. Επίσης, μπορεί να προκύψει κάποιο ενδιαφέρον συμπέρασμα. Το πρώτο περιβάλλον στο οποίο υλοποιείται είναι τα σενάρια (scripts). Από την άλλη, το δεύτερο στο οποίο χρησιμοποιείται είναι το Jupyter. Άλλα βασικά

ζητήματα, αφορούν την αδυναμία του να μπορεί να τρέξει κάποιος ένα κομμάτι κώδικα και να έχει τα ίδια αποτελέσματα την επόμενη φορά. Πέρα από αυτό αρκετές φορές μπορεί να προκύψει κώδικας που να έχει επαναληφθεί, να έχει εμφανιστεί και αλλού.[28]

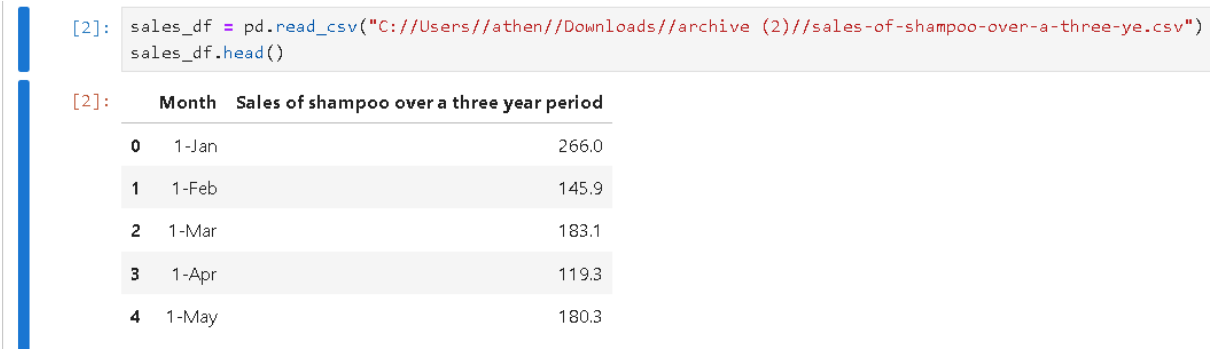
2.2.2 Αναπαράσταση χρονοσειρών μέσω Jupyter

Το Jupyter έχει πολλές εφαρμογές και χρήσεις, ειδικά σε τομείς στους οποίους η ανάλυση δεδομένων είναι απαραίτητη. Δίνει τη δυνατότητα να εμπλουτίζει κάποιος την εργασία που κάνει με πράγματα όπως κώδικα ή και κείμενο σε οποιαδήποτε μορφή. Σαν εργαλείο έχει ως στόχο να είναι πρακτικό και να μπορεί κάποιος να επιτελέσει οτιδήποτε θέλει, δίνοντας και τη δυνατότητα να απεικονίσει τα αποτελέσματα με τέτοιο τρόπο ώστε να είναι κατανοητά.

Στην πράξη λοιπόν, το Jupyter Notebook λειτουργεί ανοίγοντας το αρχείο με κατάληξη .ipynb, μέσα στο οποίο αποθηκεύονται όλες οι λειτουργίες που αναφέρθηκαν.

Η διαδικασία ανάλυσης δεδομένων της χρονοσειράς, ξεκινάει κάνοντας εισαγωγή τις βιβλιοθήκες της Python που χρειάζονται. Σε αυτό το παράδειγμα γίνεται χρήση της Pandas, stumpy, numpy, matplotlib και datetime.

Επίσης, απ' ότι φαίνεται και στην εικόνα 2.2.2.1, όταν τρέξει ένα κομμάτι κώδικα, κατευθείαν εμφανίζονται από κάτω τα αποτελέσματα. Κάτι επίσης καλό σαν δυνατότητα που παρέχει το Jupyter είναι ότι μπορεί κάποιος να αλλάξει ένα κομμάτι κώδικα, να το τρέξει πάλι και μετά αυτό να έχει και τα ανάλογα αποτελέσματα. Όπως αναφέρθηκε και πριν, κρατάει ιστορικό του τι υπολογισμούς έγιναν πριν. Σε αυτή την περίπτωση, φορτώνεται ένα αρχείο από μια διαδρομή που οδηγεί σε ένα αρχείο με επέκταση .csv τοπικά στον υπολογιστή. Αυτό θα πάρει τον ρόλο του συνόλου δεδομένων. Το συγκεκριμένο περιλαμβάνει δεδομένα από μετρήσεις μηνιαίες για τις πωλήσεις σαμπουάν, μέσα την συνολική περίοδο τριών χρόνων. Με την χρήση της βιβλιοθήκης Pandas, η οποία στην εικόνα για πρακτικούς λόγους έχει πάρει το εναλλακτικό όνομα pd, καλείται η συνάρτηση read_csv που διαβάζει αρχεία csv και μέσα σαν παράμετρο παίρνει τη διαδρομή που αναφέρθηκε πριν. Έπειτα, αυτό μετατρέπεται σε dataframe και μετά αποθηκεύεται σε μια μεταβλητή με όνομα sales_df. Αμέσως από κάτω καλείται η συνάρτηση head(), η οποία δείχνει τις 5 πρώτες εγγραφές του sales_df.



```
[2]: sales_df = pd.read_csv("C://Users//athen//Downloads//archive (2)//sales-of-shampoo-over-a-three-ye.csv")
     sales_df.head()
```

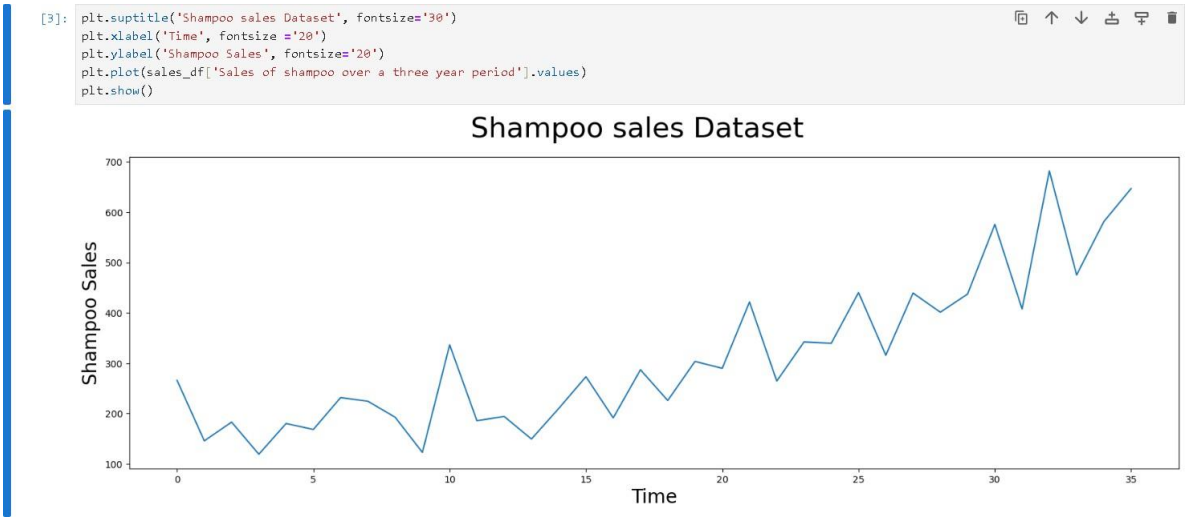
[2]:

	Month	Sales of shampoo over a three year period
0	1-Jan	266.0
1	1-Feb	145.9
2	1-Mar	183.1
3	1-Apr	119.3
4	1-May	180.3

Σχήμα 2.2.2.1 Ένα παράδειγμα εισαγωγής και εμφάνισης δεδομένων στο Jupyter

Όπως είναι εμφανές από την εικόνα 2.2.2.1, οι μετρήσεις για τις πωλήσεις σαμπουάν πραγματοποιούνται ανά ένα σταθερό χρονικό διάστημα, το οποίο σε αυτή την περίπτωση είναι ένας μήνας, πράγμα που αναδεικνύει το γεγονός ότι αποτελεί μια χρονοσειρά.

Στη συνέχεια, το αμέσως επόμενο στάδιο γίνεται η απεικόνιση των δεδομένων, με χρήση της Matplotlib. Για λόγους πρακτικούς και εδώ αντί να χρησιμοποιηθεί η κανονική ονομασία της βιβλιοθήκης Matplotlib, δίνεται η εναλλακτική ονομασία plt.



Σχήμα 2.2.2.2 Απεικόνιση δεδομένων στο Jupyter

Στην εικόνα 2.2.2.2, πέρα από το τελική απεικόνιση των δεδομένων χρονοσειράς, φαίνεται ο κώδικας που χρησιμοποιείται για την δημιουργία της. Πιο συγκεκριμένα, μέσω της plt καλούνται διάφορες συναρτήσεις οι οποίες συνεργάζονται για τη δημιουργία του διαγράμματος από κάτω. Η κάθε μια συνάρτηση που καλείται ασχολείται και με κάτι διαφορετικό. Όπως για παράδειγμα η suptitle() δίνει το τίτλο του διαγράμματος, όπως και τη μορφή που θα έχει το κείμενο. Στη συνέχεια, η xlabel() ασχολείται αποκλειστικά με τον τίτλο που θα φαίνεται στον άξονα x όπως και τη μορφή. Η ylabel() κάνει την ίδια δουλειά με το αμέσως προηγούμενο, μόνο που αυτή τη φορά επικεντρώνεται στον άξονα y. Τέλος η show() είναι αυτή που εμφανίζει την τελική απεικόνιση της χρονοσειράς.

2.3 Επίλογος

Η Python είναι μια γλώσσα προγραμματισμού που παρέχει πολλές δυνατότητες. Χρησιμοποιείται από την ανάπτυξη εφαρμογών μηχανικής μάθησης μέχρι και στην επιστήμη των δεδομένων. Αυτή η δύναμη προέρχεται από τις ποικίλες βιβλιοθήκες της. Για παράδειγμα, την NumPy που διαθέτει τα κατάλληλα εργαλεία για μαθηματικούς υπολογισμούς. Επίσης την Pandas, που μπορεί να διαχειριστεί series και dataframes. Η Matplotlib είναι υπεύθυνη για το κομμάτι της οπτικοποίησης. Η SciPy περιλαμβάνει επιστημονικού τύπου τεχνικές. Ακόμη, η scikit_learn χρησιμοποιείται αρκετά στη μηχανική μάθηση. Διότι παρέχει αυτά που χρειάζονται για την συσταδοποίηση, μείωση διαστάσεων κτλ. Επίσης, η statsmodels έχει όλα τα δεδομένα στατιστικής ανάλυσης. Όσο για την STUMPY, υλοποιείται κυρίως για την εύρεση του matrix profile. Για την τέλεση της εργασίας έγινε η χρήση της Python μέσα στο περιβάλλον Jupyter. Χάρη σε αυτό συνδυάζονται επιτυχώς το γράψιμο κώδικά με την οπτικοποίηση δεδομένων. Μάλιστα, όταν εκτελείται ένα κομμάτι κώδικα τα αποτελέσματα εμφανίζονται κατευθείαν από κάτω. Όλα αυτά γίνονται μέσα σε ένα Notebook, που είναι μέρος του Jupyter. Διαθέτει αρκετά καλά στοιχεία, όπως αυτή η άμεση αλληλεπίδραση με τον χρήστη. Ακόμη, είναι εφικτή η δυνατότητα διαμοιρασμού του Notebook με άλλα άτομα. Ωστόσο, ένα αρνητικό είναι ότι σε σύγκριση με ένα απλό σενάριο Python, είναι πιο δυσνόητο.

Κεφάλαιο 3ο: Θεωρητικό υπόβαθρο

3.1 Εργασίες - Στόχοι της διαδικασίας εξόρυξης δεδομένων μέσω της ανάλυσης χρονοσειρών

Μέσω κατάλληλων εργαλείων, ο τομέας της εξόρυξης δεδομένων προσπαθεί να εφαρμόσει την αυτόματη αναγνώριση τμημάτων που επαναλαμβάνονται μέσα σε ένα γράφημα που δημιουργείται από τα δεδομένα. Συνεπώς, οποιαδήποτε εργασία αφορά την εύρεση τέτοιων προτύπων, συνδέεται άμεσα με τον τομέα της εξόρυξης δεδομένων σε χρονοσειρές. Μέσω κατάλληλων εργαλείων, ο τομέας της εξόρυξης δεδομένων προσπαθεί να εφαρμόσει την αυτόματη αναγνώριση τμημάτων που επαναλαμβάνονται μέσα σε ένα γράφημα που δημιουργείται από τα δεδομένα. Συνεπώς, οποιαδήποτε εργασία αφορά την εύρεση τέτοιων προτύπων, συνδέεται άμεσα με τον τομέα της εξόρυξης δεδομένων σε χρονοσειρές.

Στην εξόρυξη δεδομένων σε χρονοσειρές, τα κυριότερα καθήκοντα που τις αφορούν άμεσα, τουλάχιστον όσον αφορά την προσέγγιση αυτής της ιδέας, είναι η αναζήτηση με βάση το περιεχόμενο (Query by Content), η πρόβλεψη (Prediction), η συσταδοποίηση (Clustering), η ανακάλυψη μοτίβου (Motif discovery), η κατηγοριοποίηση (Classification), η εύρεση ανωμαλιών (Anomaly detection) και η τμηματοποίηση (Segmentation)[9]

3.1.1 Query by Content

Σε γενικές γραμμές, η αναζήτηση με βάση το περιεχόμενο (Query By Content), είναι όταν σαν εισάγεται ένα υπόδειγμα ερώτησης (query), για το οποίο μετά υπολογίζεται η απόσταση του από το υπόλοιπο σύνολο δεδομένων, εξάγωντας ως αποτέλεσμα τα στοιχεία που ήταν πιο κοντά σε απόσταση από αυτό.

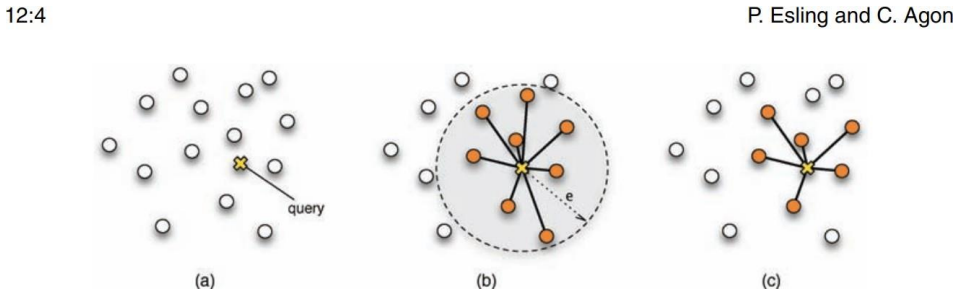
Στα ερωτήματα αυτά, αλλάζοντας το κομμάτι στο οποίο γίνεται η αντιστοιχία, προκύπτουν δύο διακριτές κατηγορίες. Η πρώτη από αυτές είναι όταν η συσχέτιση και η εύρεση των πιο κοντινών γίνεται με βάση όλη τη σειρά και λέγεται ταίριασμα ολόκληρης σειράς (whole series matching). Η δεύτερη είναι όταν η συσχέτιση γίνεται για την εύρεση όλων των μικρότερων τμημάτων μιας σειράς και είναι γνωστό ως ταίριασμα υποακολουθίας (subsequence matching)

Ανάλογα με το πώς γίνεται η εύρεση απόστασης, η αναζήτηση με βάση το περιεχόμενο, διακρίνεται στα εξής τμήματα:

Στην αναζήτηση με βάση τους k πιο όμοιους γείτονες, η μεταβλητή k παίζει κύριο ρόλο. Στην ουσία η χαρακτηριστική διαφορά μεταξύ των δύο τύπων αναζήτησης είναι ότι αντί να χρησιμοποιηθεί το εύρος e, γίνεται με την τιμή k. Στόχος του είναι η εμφάνιση των στοιχείων που έχουν την μικρότερη απόσταση από το ερώτημα. Αυτό επιτυγχάνεται παρέχοντας τη μεταβλητή k. Ανάλογα με το τι αριθμό της δώσει κάποιος, η ποσότητα των στοιχείων που θα φέρει αλλάζει. Η διαδικασία αυτή αφορά τις χρονοσειρές. Στο ερώτημα μπαίνει μια χρονική σειρά και μετά το τελικό σύνολο με τις πιο κοντινές περιλαμβάνει σειρές.

Μέσω της αναζήτησης εύρους (Range query), ένας κανόνα εισάγεται από τον χρήστη. Αφορά το πόσο κοντά ή μακριά μπορούν να είναι τα στοιχεία ώστε να είναι στο τελικό σύνολο. Πιο συγκεκριμένα, η μεταβλητή e που υπάρχει μέσα στον τύπο πιο κάτω το ορίζει αυτό και πρέπει να δίνεται με προσοχή.

Είναι μερικές περιπτώσεις στις οποίες είναι προτιμότερο να γίνεται η αναζήτηση με βάση τους k πιο όμοιους γείτονες, αντί για τη αναζήτηση εύρους. Αυτό επειδή, αν κάποιος δεν έχει γνώση του συνόλου δεδομένων, η αναζήτηση αυτή μπορεί να μην φέρει τα αποτελέσματα που περιμένει.[9]



Σχήμα 3.1.1.1 Αναπαράσταση Query by Content [9]

Στο σχήμα 3.1.1.1 φαίνεται από την αρχή μέχρι το τέλος όλη η διαδικασία που περιγράφηκε με βάση χρονοσειρές. Είναι χωρισμένη σε τρία στάδια. Το (b) και το (c) αφορούν τους διαφορετικούς τύπους αναζήτησης. Όσο για τη σειρά που λειτουργεί ως ερώτημα, αυτή συμβολίζεται από το κίτρινο x. Η διαδικασία της αναζήτησης με βάση το περιεχόμενο ξεκινάει από την εικόνα (a), στην οποία γίνεται κατάλληλη προετοιμασία. Στόχος είναι να εξισορροπηθούν όλα τα στοιχεία που βρίσκονται μέσα στον ίδιο χώρο. Ανάλογα με τις ανάγκες και τα ζητούμενα, υπάρχει η αναζήτηση εύρους ή η χρήση των k πιο όμοιων στοιχείων. Στο σχήμα, αυτό που φαίνεται στο (b) δείχνει το αποτέλεσμα που προκύπτει θέτοντας μια τιμή e για το εύρος. Παρατηρείται ότι ο αριθμός στοιχείων που αντιστοιχήθηκαν, πέρα από το ερώτημα, είναι 8. Κάποια από αυτά βρίσκονται σχετικά μακριά από τη σειρά του ερωτήματος, αλλά παρόλα αυτά εμφανίστηκαν. Από την άλλη, η εικόνα (c) δείχνει το τι γίνεται με την εφαρμογή μιας τιμής k. Απ' ότι φαίνεται, προέκυψε ότι μόνο 6 σειρές έχουν την μεγαλύτερη ομοιότητα με τη σειρά του ερωτήματος. Σε γενικές γραμμές, όλα τα σημεία με που βρέθηκαν είναι σχετικά κοντά με το ερω΄τημα. Τουλάχιστον σε σύγκριση με το (b), όπου δύο ήταν πιο μακριά. Επειδή όλες οι κουκίδες αποτελούν χρονοσειρές, στην πραγματικότητα τα αποτελέσματα που προκύπτουν και από την (a) και την (b) είναι και αυτά χρονικές σειρές. [9]

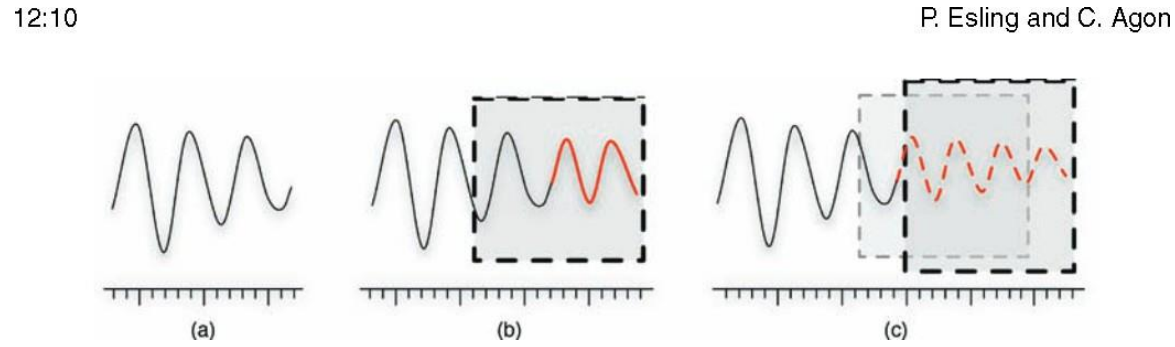
3.1.2 Prediction

Η πρόβλεψη (Prediction) είναι μια εργασία της εξόρυξης δεδομένων σε χρονοσειρές που είναι αρκετά δημοφιλής και αναζητά το πως θα εξελιχθούν χρονικά. Το παραπάνω το καταφέρνει, υλοποιώντας ένα πόρισμα που εξάγεται παρατηρώντας τις τιμές μιας σειράς και έπειτα βλέποντας το αν και πώς συσχετίζονται μεταξύ τους. Αν ο τρόπος με τον οποίο κινείται μια χρονοσειρά δεν έχει κάποια βάση, τότε δύσκολα θα μπορέσει να βρεθεί κάποιο πρότυπο ώστε να γίνει η όλη διαδικασία.

Είναι λογικό ότι μέσα στην πάροδο των χρόνων, υπήρξαν αρκετές τεχνικές οι οποίες αφορούσαν άμεσα τον τομέα της μηχανικής μάθησης. Γι' αυτό είναι αναγκαίο να γνωρίζει κάποιος ποιες από τις

μεθόδους αυτές βρίσκονται πιο υψηλά σε επίδοση σε σύγκριση με τις άλλες. Μετά από μια έρευνα που πραγματοποιήθηκε από τον Ahmed το 2009, τα αποτελέσματα έβγαλαν ξεκάθαρους νικητές. Η πρώτη είναι η παλινδρόμηση της γκαουσιανής διαδικασία. Η δεύτερη που βγήκε είναι το perceptron με πολλά επίπεδα.

Άλλες τεχνικές που σχετίζονται άμεσα με την πρόβλεψη θα αναφερθούν στη συνέχεια. Όπως αυτές που προήλθαν από τον Box το 1976, με τα AR μοντέλα, τον Koskela το 2003 με τα νευρωνικά δίκτυα και τέλος τον Barreto το 2007 με τον SOM. Όλα τα παραπάνω προσέγγισαν το θέμα της αναγνώρισης και εξαγωγής τιμών που δεν χρειάζονταν από δεδομένα που προέρχονταν καθαρά από σήματα. Δηλαδή στην ουσία καθαρισμού του σήματος. Όσο για τις σειρές που δεν είναι συνεχής, μια πρόταση δόθηκε από τον Pesaran το 2006. Πιο συγκεκριμένα, αναφέρθηκε ότι για την εκτίμηση των επόμενων τιμών, μπορεί να χρησιμοποιηθεί η γνωστή Μπαιζιαν πρόβλεψη(Bayesian prediction). [9]



Σχήμα 3.1.2.1 Αναπαράσταση Prediction [9]

Στο σχήμα 3.1.2.1 παρουσιάζονται όλα τα στάδια υλοποίησης αυτού. Πιο συγκεκριμένα, στο πρώτο φαίνεται η αρχική χρονοσειρά από την οποία θέλουμε να βρούμε ποιες τιμές ακολουθούν στη συνέχεια. Αν η χρονοσειρά εξελισσόταν χωρίς κάποια συγκεκριμένη λογική, τότε δε θα υπήρχε κάποιο πρότυπο με βάση το οποίο θα βρεθούν οι επόμενες τιμές της. Γι' αυτό πρέπει κάποια τμήματα της να επαναλαμβάνονται ανά κάποια χρονικά διαστήματα. Με αυτό τον τρόπο θα μπορέσει να γίνει εκτίμηση των μελλοντικών τιμών. Όπως φαίνεται και από το (a), ικανοποιούνται τα παραπάνω κριτήρια. Στο επόμενο βήμα, το πλαίσιο με τις διακεκομμένες γραμμές δείχνει το χώρο στον οποίο θα γίνει η εύρεση των στοιχείων που έπονται. Η ποσότητα αυτών όσο πιο μεγάλη είναι τόσο το καλύτερο. Στην τελευταία φάση φαίνεται τι συμβαίνει όταν η πρόβλεψη δεν είναι για μικρή χρονική διάρκεια. Για να μην σταματήσει η διαδικασία , ο αλγόριθμος συνεχίζει κανονικά. Μόνο που αυτή τη φορά, σαν βάση θα υλοποιήσει παλαιότερα κομμάτια. [9]

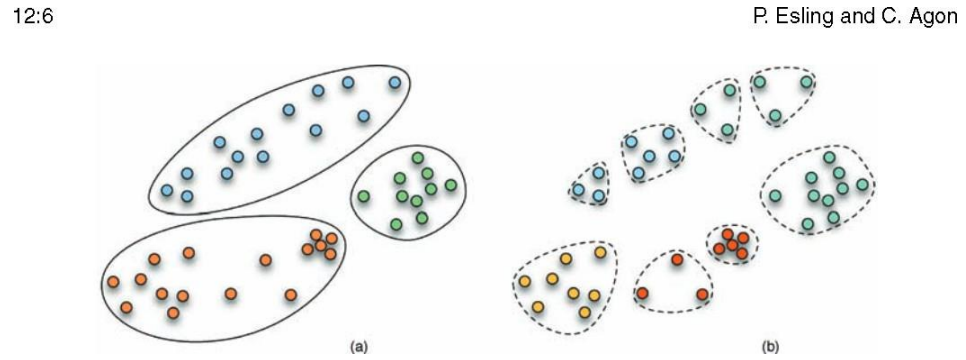
3.1.3 Clustering

Η εύρεση του ιδανικού αριθμού συστάδων είναι ένα αρκετά λεπτό ζήτημα, καθώς είναι κάπως ευαίσθητη η τιμή. Αλλάζοντας την ελάχιστα, μπορεί να βγουν εντελώς διαφορετικά αποτελέσματα. Αυτό αποτελεί ένα ουσιώδες κομμάτι της συσταδοποίησης (Clustering). Ο διαμοιρασμός των δεδομένων σε μικρότερα τμήματα γίνεται από αυτή την εξαιρετικά σημαντική εργασία Καθένα και από αυτά αποτελεί μια συστάδα (cluster). Όλα τα στοιχεία μέσα σε αυτή μοιράζονται κοινά χαρακτηριστικά που τους ενώνουν. Παράλληλα, αυτές οι ιδιότητες είναι που τους κάνουν να έχουν εξαιρετική απόσταση με εκείνα που είναι εκτός αυτής.

Κάτι που δεν πρέπει να παραλειφθεί, είναι ότι η συσταδοποίηση μπορεί να εφαρμοστεί και πάνω σε χρονοσειρές. Δηλαδή είναι δυνατό να μπουν κανονικά σε συστάδες, όπου πάλι ισχύουν τα ίδια που αναφέρθηκαν πριν.

Μάλιστα η διαδικασία τη συσταδοποίησης σε χρονοσειρές διακρίνεται σε δύο κατηγορίες με βάση τι υλοποιείται. Αν τη θέση των στοιχείων που χρησιμοποιήθηκαν για τη εύρεση συστάδων είχαν πάρει υποακολουθίες, λέγεται συσταδοποίηση υποακολουθιών (sub series clustering). Σε αντίθεση με αυτό, όταν χρησιμοποιούνται οι σειρές για την όλη διαδικασία, τότε αυτό είναι γνωστό ως συσταδοποίηση ολόκληρων σειρών (whole series clustering).

Κατά τη διάρκεια των ετών, προέκυψαν διάφοροι αλγόριθμοι γι' αυτό το θέμα. Από τους πρώτους, ο Chappelier και ο Grumbach το 1996, έκαναν αναφορά σε μια αρκετά καλή μέθοδο. Βασίζεται πάνω στους αυτο-οργανωμένους χάρτες (SOM). Ένα χρόνο αργότερα, τα κρυφά μοντέλα του Μάρκοβ προτάθηκαν από τον Smyth. Γενικά, δεν υπάρχει κάποιος περιορισμός όσον αφορά το ποιοι αλγόριθμοι συσταδοποίησης μπορούν να χρησιμοποιούνται στις χρονοσειρές Συνεπώς, μπορούν να εφαρμόζονται οι ίδιοι μέθοδοι που υλοποιούν οποιαδήποτε άλλα δεδομένα.[9]



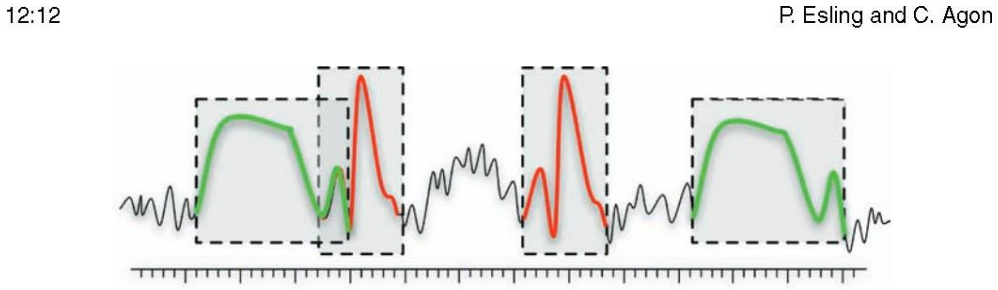
Σχήμα 3.1.3.1 Αναπαράσταση clustering αλγόριθμου [9]

Στο σχήμα 3.1.3.1, τόσο το (a) όσο και το (b) διαθέτουν κάτι κοινό που τους ενώνει. Προέρχονται από την εφαρμογή της συσταδοποίησης. Η μόνη διαφορά είναι στον αριθμό συστάδων που ορίστηκε. Στο (a) η τιμή αυτή είναι τρία. Από την άλλη, στο (b) η τιμή είναι οκτώ. Άμεσα μπορεί να παρατηρηθεί ότι υπάρχει ένα χάσμα όσον αφορά το πόσο διεξοδικά έγινε η διαδικασία της συσταδοποίησης και στα δυο. Μάλιστα στο (b) φαίνεται πώς είναι πολύ πιο καλά συγκροτημένες οι συστάδες. Τα στοιχεία μέσα σε κάθε μια από αυτές είναι πιο κοντά σε απόσταση μεταξύ τους. Από την άλλη, στο (α) φαίνεται πως υπάρχουν αρκετα κενά διαστήματα. Κοιτώντας αυτά που προέκυψαν, μπορεί να γίνει αντιληπτό το πόσο σημαντικό είναι το ζήτημα ορισμού τιμής για συστάδες.[9]

3.1.4 Motif Discovery

Υπάρχουν κάποια σοβαρά θέματα που προκύπτουν από την υλοποίηση της διαδικασίας ανακάλυψης μοτίβων (Motif discovery). Προέρχονται από διάφορα πράγματα. Όπως για παράδειγμα, όταν δεν είναι εφικτή η διάκριση του κάθε μοτίβου. Ακόμη, όσο αυξάνεται ο αριθμός των μοτίβων, τόσο χειροτερεύει το θέμα. Όλα αυτά οδηγούν στην επιβάρυνση υπολογιστικής δύναμης. Σαν εργασία, έχει ρίζες στον τομέα που χρησιμοποιείται για την ανάλυση γονιδίων. Όταν παρατηρώντας μια χρονοσειρά διακρίνεται ότι ένα κομμάτι πάνω στο γράφημα που δημιουργείται, είναι ίδιο με ένα άλλο, τότε αυτό λέγεται μοτίβο. Η μέθοδος που ακολουθείται για την εύρεση πολλών, λέγεται ανακάλυψη μοτίβων.

Γενικά υπήρξαν διάφορες μέθοδοι που προσπάθησαν να προσεγγίσουν αυτό το θέμα στις χρονοσειρές. Όπως για παράδειγμα, ο αλγόριθμος τυχαίας προβολής από τον Chiu το 2003 και αλγόριθμος όπου έχει χαλαρή τιμή ακρίβειας από τον Ferreira το 2006. Παρατηρώντας τις δυνατότητες και των δύο, τόσο το πρώτο όσο και το δεύτερο δεν στηρίζονται στην επίτευξη του τέλειου αποτελέσματος. Δηλαδή σε μια σειρά δεν χρειάζεται να βρίσκονται αυστηρά όλες οι ίδιες μικρότερες ακολουθίες. Ακόμη, και τα δύο είχαν επιτυχείς εφαρμογές. Σε ακολουθίες DNA, η ανίχνευση αυτών έγινε από τον αλγόριθμο τυχαίας προβολής. Όσο για το δεύτερο, κυρίως σε χρονικές σειρές που σχετίζονται με τη μελέτη πρωτεϊνών.[9]



Σχήμα 3.1.4.1 Αναπαράσταση Motif Discovery [9]

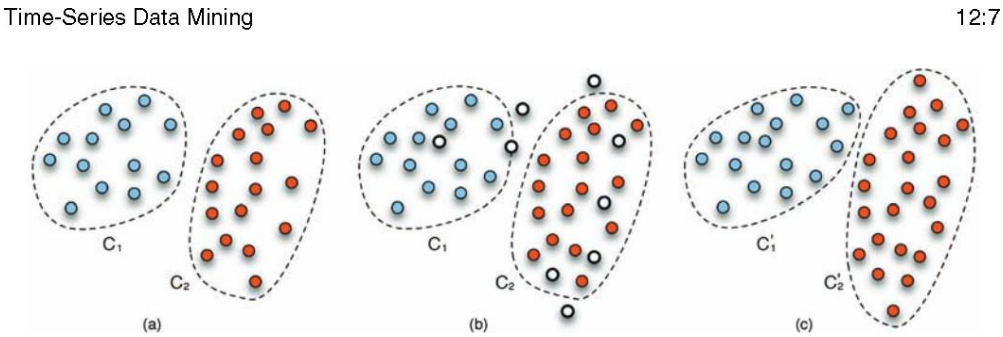
Στο σχήμα 3.1.4.1 απ’ ότι φαίνεται έχουν βρεθεί δύο διαφορετικά μοτίβα. Το πρώτο ταίριασμα είναι μεταξύ αυτών των υποακολουθιών με την πράσινη γραμμή, ενώ το δεύτερο είναι μεταξύ αυτών με την κόκκινη. Μέσω του σχήματος διακρίνονται κάποιες από τις δυσκολίες στην διαδικασία ανακάλυψης που αναφέρθηκαν πιο πριν. Για παράδειγμα, όπως ακριβώς φαίνεται και στην εικόνα 3.1.4.1 μεταξύ των δύο πρώτων υποακολουθιών, τα μοτίβα δεν διακρίνονται το ένα από το άλλο. Επίσης συγκρίνοντας τα μήκοι και των τεσσάρων υποακολουθιών, παρατηρείται ότι δεν υπάρχει κάποιο ενιαίο μέγεθος. Όλα τα παραπάνω δημιουργούν ιδιαίτερο θέμα, παρουσιάζοντας μια παραπάνω δυσκολία ως προς την αντιμετώπιση.[9]

3.1.5 Classification

Η ένταξη ενός στοιχείου σε μια κλάση, με βάση τα χαρακτηριστικά που παρουσιάζει είναι ευθύνη της κατηγοριοποίησης (Classification). Κάθε ένα από αυτά βοηθάει στην ενίσχυση ιδιαιτερότητας της εκάστοτε κλάσης. Γι’ αυτό είναι σημαντική η ταυτοποίηση αυτών. Ο αλγόριθμος κατηγιοποιοποίησης δεν μπορεί να εκτελεστεί χωρίς να έχει γίνει κάποια προετοιμασία. Ένα μεγάλο μέρος αυτής, καθιστά αναγκαίο να αναπτυχθεί η δεξιότητα του. Στην ουσία, ξεκινάει χωρίς καμία εμπειρία. Μετά, μέσω ενός μικρού συνόλου, την εξελίσει όλο και περισσότερο. Βέβαια, ήδη ξέρει πόσες και ποιες κλάσεις υπάρχουν. Δεν είναι δηλαδή κρυφή αυτή η πληροφορία. Όλο αυτό το κομμάτι αποτελεί στην ουσία την εκπαίδευση του. Οσον αφορα τις χρονοσειρές, η διαδικασία παραμένει ίδια. Μόνο που αυτό που κατηγοριοποιείται, είναι ένα δείγμα της τη φορά. Αυτό το καταφέρνει, βλέποντας τις ιδιότητες που παρουσιάζουν.

Ανάμεσα στις ποικίλες προτάσεις για την κατηγοριοποίηση χρονοσειρών, η πιο αποδοτική προκύπτει από τον συνδυασμό μεταξύ της δυναμικής χρονικής παραμόρφωσης και του ενός πιο όμοιου γείτονα. Μερικές από τις υπόλοιπες αναφορές πάνω στο ζήτημα, περιλαμβάνουν την αποσύνθεση μοναδικής

τιμής, από τον Jeng και τον Huang το 2008, την χρήση τάσεων από Bakshi και Stephanopoulos το 1994, ο Keogh και Pazzani το 1998 μίλησαν για χρήση τμηματικής αναπαράστασης κτλ.[9]



Σχήμα 3.1.5.1 Αναπαράσταση αλγορίθμου Classification [9]

Στο σχήμα 3.1.5.1 όπως φαίνεται στο (a) έχει την κλάση C1 που περιέχει κουκίδες με μπλε χρώμα. Αντίστοιχα στη C2 οι κουκίδες είναι με κόκκινο. Είναι ήδη είναι γνωστό σε ποια από τις δύο ανήκουν. Όσο για τις διακεκομμένες γραμμές γύρω την C1 και C2 έχουν σημαντική χρήση. Από αυτά κρίνεται η μοναδικότητα της κάθε κλάσης, αυτό που δίνει ιδιαίτερο ύφος. Έπειτα στο (b) οι άσπρες κουκίδες δείχνουν τα καινούργια στοιχεία που περιμένουν να μπουν σε μια κλάση. Τέλος στο (c) ανάλογα με το ποια από τις δύο κλάσεις βρίσκονται πιο κοντά σε απόσταση τα στοιχεία, εντάσσονται αναλογώς σε μια από τις δυο. [9]

3.1.6 Ανακάλυψη κανόνων συσχέτισης

Παρά την ραγδαία αύξηση των δεδομένων τη σημερινή εποχή, χωρίς την εξαγωγή και την κατάλληλη διαχείριση της, η οποιαδήποτε πληροφορία δε θα είχε χρήση. Συνεπώς, είναι επιτακτική η δυνατότητα συσχέτισης δεδομένων με τέτοιο τρόπο ώστε να προκύπτουν ενδιαφέροντα πορίσματα για το σύνολο τους. Η παραπάνω σχέση με στόχο την εύρεση συμπερασμάτων υπάρχει και μάλιστα είναι γνωστοί ως κανόνες συσχέτισης (Association Rules). Ανάλογα με το κάθε σύνολο, μπορούν να βγουν πολλοί διαφορετικοί κανόνες. Εξαιτίας αυτού προκύπτει ένα ζήτημα που αφορά τον κίνδυνο χάσιμου των σημαντικών πληροφοριών από τα δεδομένα.

Σύμφωνα με την παραπάνω λογική, μπορούν να προκύψουν πολλοί κανόνες συσχέτισης από τον πίνακα με τα δεδομένα καιρού στο σχήμα 3.1.6.1. Για παράδειγμα, για Outlook να είναι sunny, Temperature είναι cool, Humidity είναι normal και Windy είναι false, τότε το Play θα είναι yes. Δηλαδή αν ο καιρός είναι ηλιόλουστος, δεν κάνει ζέστη, έχει κανονική τιμή υγρασίας και καθόλου άνεμο, τότε ο καιρός είναι κατάλληλος για παιχνίδι. Με αυτό τον τρόπο, κάνοντας ποικίλους συνδυασμούς, προκύπτουν και πολλά διαφορετικά πορίσματα. Ακόμη ένα θετικό είναι ότι δεν υπάρχει κανένας περιορισμός όσον αφορά για ποιο χαρακτηριστικό θα εξαχθεί ένα συμπέρασμα από τους κανόνες συσχέτισης, δεν είναι δηλαδή κάτι απόλυτο.

Στο σχήμα 3.1.6.1 φαίνεται ενας πίνακας που περιέχει τα απαραίτητα στοιχεία που χρειάζονται για την ανακάλυψη κανόνων συσχέτισης. Η τελευταία στήλη Play είναι αρκετά σημαντική καθώς δείχνει ποια θα είναι η τελική απόφαση. Μπορεί να πάρει δύο διαφορετικές τιμές, που είναι ναι ή οχι. Όσο για τις υπόλοιπες στήλες του πίνακα, περιέχουν τα διαφορετικά χαρακτηριστικά που αφορούν τον καιρό. Αυτά είναι και οι κύριες κατηγορίες. Σε γενικές γραμμές, τα χαρακτηριστικά μπορούν να πάρουν διάφορες τιμές, δεν περιορίζονται δηλαδή σε μια. Συνδυάζοντας τις παραπάνω τιμές προκύπτει μια περίπτωση(instance). Σε αυτό συγκαταλέγεται και η τελευταία στήλη. Ουσιαστικά είναι μια σειρά

μέσα στον πίνακα του σχήματος 3.1.6.1. Ακόμη ένα θετικό που παρέχουν οι κανόνες συσχέτισης είναι ότι δεν υπάρχει κανένας περιορισμός όσον αφορά για ποια κατηγορία θα εξάγουν ένα συμπέρασμα, δεν είναι δηλαδή κάτι απόλυτο. Γι‘ αυτό και σε γενικες γραμμές υπάρχει εμφάνιση πολλών κανόνων συσχέτισης. [10]

Table 1.2 The weather data.

Outlook	Temperature	Humidity	Windy	Play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no

Σχήμα 3.1.6.1 Πίνακας με δεδομένα καιρού [10]

Μέσω ανάθεσης ορίων στις τιμές στοιχείων όπως η υποστήριξη και εμπιστοσύνη λύνεται το ζήτημα ύπαρξης μεγάλου αριθμού κανόνων συσχέτισης. Με αυτό τον τρόπο, απομονώνεται μόνο η σημαντική πληροφορία. Τα στοιχεία αυτά, που λειτουργούν ως κριτήρια με βάση τα οποία θα γίνει αυτή η επιλογή είναι δύο:
Το πρώτο προκύπτει από τη τιμή της υποστήριξης δια του πόσες φορές μέσα στο σύνολο δεδομένων θα μπορούσε να ισχύει. Αυτό ονομάζεται εμπιστοσύνη (confidence). Το δεύτερο βγαίνει αν μετρηθούν πόσες είναι οι προβλέψεις που όντως ήταν αληθείς και είναι γνωστό ως υποστήριξη (support). Ακόμη μέσα σε όλο αυτή τη διαδικασία επιλογής ισχυρότερων κανόνων, πρέπει να συμπεριληφθούν και όσοι είναι αρκετά παρόμοιοι μεταξύ τους. Πέρα από αυτό, υπάρχει και άλλος τρόπος επίλυσης του θέματος. Προκύπτει μετά από σύγκριση μεταξύ των κανόνων Σε αυτές τις περιπτώσεις και έχοντας ως στόχο το να μείνουν οι πιο σημαντικοί, αυτό που συνιστάται είναι γίνει σύγκριση μεταξύ τους και στο τέλος να μείνει αυτός με τη μεγαλύτερη ισχύ.[10]

3.1.7 Anomaly Detection

Η συγκεκριμένη εργασία έχει ποικίλες εφαρμογές. Πιο συγκεκριμένα, αφορούν άμεσα οτιδήποτε περιέχει ανίχνευση αποκλινόντων τιμών. Όπως για παράδειγμα στην εύρεση περίεργων τιμών που ίσως μπορεί να δείχνουν μια πιθανή επίθεση. Με αντίστοιχο τρόπο μπορεί να βοηθήσει και στον κλάδο που σχετίζεται με την παρακολούθηση βιολογικών δεδομένων, έτοιμο να ανιχνεύσει οτιδήποτε βγει εκτός. Τέτοιου είδους συμπεριφορά θεωρείται ως ασυμφωνία. Η διαδικασία ταυτοποίησης αυτών (Anomaly Detection) είναι αρκετά σημαντική και όπως φαίνεται είναι πολύ βοηθητικές σε αρκετά πράγματα.

Σαν διαδικασία υπάρχουν ποικίλοι τρόποι υλοποίησης. Μέσα σε αυτούς το 2003 οι Ma και Perkins αναφέρθηκαν στη χρήση υποστηρικτικής παλινδρόμησης διανυσμάτων,οι Ypma και Duin το 1997, που πρότειναν τον SOM κτλ. Μάλιστα, για να πετύχει τον σκοπό εύρεσης, το δεύτερο από αυτά έκανε την χρήση της πρόβλεψης. Αν και διαφορετικές, όλες οι μέθοδοι που αναφέρθηκαν στηρίχτηκαν σε κάτι κοινό. Στο πόσο σημαντική είναι η εύρεση μιας βάσης για την σύγκριση. Διότι δεν είναι δυνατόν να επιτευχθεί η ταυτοποίηση της οποιασδήποτε τιμής που θεωρείται περίεργη χωρίς αυτή.[9]

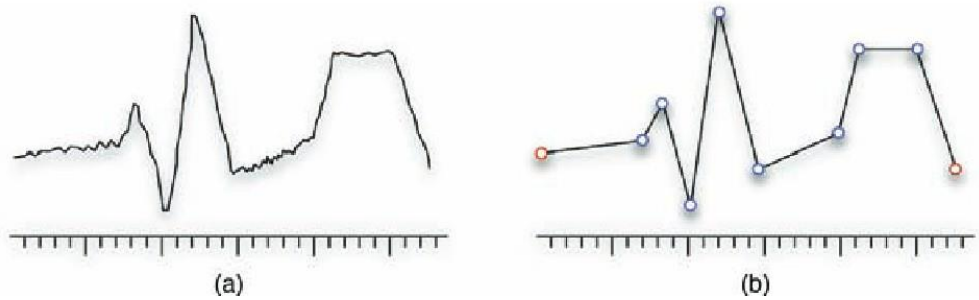
Σχήμα 3.1.7.1 Αναπαράσταση αλγορίθμου εύρεσης ανωμαλιών [9]

Στο σχήμα 3.1.7.1 απ' ότι διακρίνεται στα τμήματα που είναι εκτός του διακεκομμένου τετραγώνου, παρατηρείται εύκολα ότι μια υποσειρά εμφανίζεται ξανά ανά κάποια συγκεκριμένα χρονικά διαστήματα. Με βάση αυτό λοιπόν μπορεί να εξαχθεί πρότυπο της χρονοσειράς. Άρα είναι λογικό οτιδήποτε βγει εκτός, πάντα σε σύγκριση με αυτό το πρότυπο, θεωρείται ως περίεργο. Σε αυτή την περίπτωση, θεωρείται ως περίεργη η υποσειρά που βρίσκεται μέσα στις διακεκομμένες γραμμές. Δηλαδή η υποσειρά με το κόκκινο χρώμα [9]

3.1.8 Segmentation

Ένας από τους κινδύνους που σχετίζεται με αυτή την εργασία είναι η ιδιαίτερη μορφή μιας χρονοσειράς να αλλάξει και να γίνει διαφορετική. Αυτό είναι κάτι που προσπαθεί να αποφύγει η συγκεκριμένη μέθοδος. Από αυτή τη διαδικασία προκύπτει ένα σφάλμα. Αυτό δείχνει ότι η προσέγγιση της χρονοσειράς πλησιάζει την αρχική μορφή. Συνεπώς είναι καλό να είναι όσο πιο μικρό γίνεται σε αριθμό. Η διαδικασία αυτή είναι γνωστή ως τμηματοποίηση. Θέλει να μην χάσει την αρχική πληροφορία. Ταυτόχρονα όμως, προσπαθεί να μικρύνει τη σειρά σε μέγεθος.

Όσον αφορά το κομμάτι του πως επιτυγχάνεται η τμηματοποίηση υπάρχουν διάφοροι μέθοδοι που χρησιμοποιήθηκαν. Όπως εκείνη της χρήσης του κάτω προς τα πάνω από τον Keogh και Pazzani το 1998, των κρυφών μοντέλων Μάρκοβ από τον Kehagia το 2004, ο Li το 1998 έκανε αναφορά για το πάνω προς τα κάτω, το κάτω προς τα πάνω από τον Keogh και Pazzani, το 1998 οι Shatkay και Zdonik το 1996 πρότειναν τα συρόμενα παράθυρα κτλ. τρεις κύριες μέθοδοι κάθε μια από τις οποίες παρέχει και κάτι διαφορετικό. Το καλύτερο κατά ένα μεγάλο βαθμό είναι το κάτω προς τα πάνω, μεταξύ των των συρόμενα παραθύρων και του πάνω προς τα κάτω. Ακόμη εκείνη η μέθοδος όπου στο τέλος προκύπτουν πολυωνυμικές συναρτήσεις που αντιστοιχούν σε κάποια κομμάτια της αρχικής σειράς, η αλλιώς τμηματική γραμμική προσέγγιση (Piecewise Linear Approximation) είναι από τις πιο δημοφιλείς. Προήλθε από τους Shatkay και Zdonik το 1996.[9]



Σχήμα 3.1.8.1 Αναπαράσταση Segmenation [9]

Στο σχήμα 3.1.8.1 παρουσιάζεται μια χρονοσειρά πριν την τμηματοποίηση στο (a). Στο (b) δείχνει το μετά στάδιο. Μάλιστα στο (a) παρατηρείται ότι υπάρχουν δεδομένα τα οποία δεν χρειάζονται, ενώ στο (b) φαίνεται ότι αυτά έχουν εξαφανιστεί λόγω της όλης διαδικασίας. Πράγμα λογικό καθώς στο (b) στην ουσία η μικρότερη εκδοχή της πρώτης διακρίνεται, τουλάχιστον όσον αφορά τα σημεία. Επίσης ο στόχος ικανοποιήθηκε διότι φαίνεται ότι το (b) κατάφερε να μην χάσει πολύ πληροφορία από το αρχικό σχήμα που δείχνει στο (a).[9]

3.2 Διαδικασία αναζήτησης όμοιων χρονοσειρών

Προτού εφαρμοστεί η οποιαδήποτε μέθοδος πάνω στα δεδομένα χρονοσειρών, πρέπει να υπάρξει ένα στάδιο προεπεξεργασίας τους. Μάλιστα, ανάλογα με την λειτουργία διακρίνονται σε δύο κατηγορίες. Στην πρώτη ανήκουν αυτές που εξαιτίας του όγκου δεδομένων που έχουν οι χρονοσειρές έχουν σαν στόχο τη μείωση αυτών (dimensionality reduction). Στη δεύτερη περιλαμβάνονται εκείνες που είναι για χωρική πρόσβαση (spatial access methods). Χάρη στην υλοποίηση αυτών, μπορούν να εφαρμοστούν οι εργασίες εξόρυξης. Από την κατηγοριοποίηση (classification) μέχρι και την ανακάλυψη μοτίβων (motif discovery). Μπορεί να είναι διαφορετικές μεταξύ τους, ενώνονται όμως από κάτι κοινό. Αυτό είναι πως όλα στηρίζονται πάνω στην έννοια των συναφών οντοτήτων. Η έννοια της απόστασης στην περίπτωση των βάσεων είναι το αποτέλεσμα της σύγκρισης ενός προτύπου (query) με τα υπόλοιπα στοιχεία που υπάρχουν μέσα στο σύνολο. Ένα εύλογο συμπέρασμα είναι ότι αυτό με τη μικρότερη απόσταση είναι και το ποιο όμοιο. Συνεπώς, η εύρεση και η ταυτοποίηση αυτής της σχετικότητας βοηθάει στην εξαγωγή ενδιαφερόντων πορισμάτων, που χρησιμοποιούνται άμεσα σε πολλούς τομείς.[17] Βέβαια, η αναγνώριση ολόιδιων στοιχείων δεν είναι ο στόχος. Όπως στη μελέτη δεδομένων που σχετίζονται με τον καιρό. Πιο συγκεκριμένα, στον εντοπισμό παραπλήσιων συνθηκών ανάμεσα σε περιοχές και για ποια χρονική περίοδο, το ταίριασμα δεν είναι απόλυτα λεπτομερές. Σε άλλη περίπτωση, όταν τα δεδομένα της χρονοσειράς αφορούν μια μετοχή, στην εύρεση προτύπων που επαναλαμβάνονται, δεν είναι αναγκαία η τέλεια αντιστοίχιση. Με ανάλογο τρόπο, το ίδιο ισχύει και για οποιεσδήποτε άλλες εφαρμογές που χρησιμοποιούνται σε προβλήματα της καθημερινής ζωής.[18]

3.2.1 Η έννοια της ομοιότητας χρονοσειρών

Όπως αναφέρθηκε και στη προηγούμενη ενότητα, η εύρεση άμεσης συσχέτισης ανάμεσα σε παρεμφερείς οντότητες είναι κάτι αρκετά σημαντικό. Πρέπει να γίνει αντιληπτό ότι δεν χρειάζεται να

υπάρχει η εύρεση ή αλλιώς το κυνήγι του τέλειου ταιριάσματος. Αντί αυτού, θα πρέπει να μπορεί να βρεθεί στο περίπου. Το ποια προσέγγιση πρέπει να ακολουθηθεί γι' αυτό, είναι ένα αξιόλογο ζήτημα που θα μελετηθεί σε αυτές τις ενότητες. Ένα άλλο θέμα αφορά τις εργασίες εξόρυξης δεδομένων. Είναι επιτακτική η ανάγκη προσαρμογής αυτών. Όλο αυτό γίνεται ως απάντηση στην αύξηση των δεδομένων χρονοσειρών.

Μέχρι τώρα έχει καλυφθεί το λίγο πιο θεωρητικό κομματι του τι σημαίνει ο υπολογισμός παρόμοιων στοιχείων. Τώρα στην πράξη, αυτό εφαρμόζεται με έναν συγκεκριμένο τρόπο. Μάλιστα, συμβολίζεται με το γράμμα D. Ο τύπος που ακολουθείται για την εύρεση αυτού είναι ο D(T, U). Στις υπποσειρές λειτουργεί με ανάλογο τρόπο. Η μόνη διαφορά είναι στις παραμέτρους που θα υλοποιηθούν. Χαρακτηρίζεται ως μετρική με βάση το αν καλύπτονται τα εξής τρία κριτήρια:

- Τριγωνική ανισότητα. Όταν το άθροισμα των αποστάσεων αυτής που αναφέρθηκε πριν και αυτής που προκύπτει από την U και V, είναι μεγαλύτερο από αυτό της απόστασης T και V.

- Συμμετρική. Όταν δύο μέθοδοι που μετρούν την απόσταση έχουν ίσο αποτέλεσμα με κοινές τις δύο χρονοσειρές όπου η μόνη διαφορά είναι ότι αλλάζει η θέση των χρονοσειρών από το ένα στο άλλο.

- Εννοείται ότι σε ό,τι αφορά τον υπολογισμό απόστασης το αποτέλεσμα θα είναι πάντα μεγαλύτερο του μηδέν. [9]

3.2.2 Μέτρα απόστασης χρονοσειρών

Για την χρήση των μετρικών απόστασης πρέπει να πρώτα να υλοποιηθούν κάποια κριτήρια όσον αφορά την λειτουργία τους. Αρχικά πρέπει να υπάρχει ελευθερία για τον τύπο δεδομένων της χρονοσειράς στα οποια θα γίνει ο υπολογισμός της απόστασης. Δηλαδή, είναι σημαντικό το να είναι ευέλικτο το μέτρο σε διαφόρους τύπους δεδομένων. Επίσης, οφείλει να βρίσκει τις ιδιαιτερότητες που εμφανίζουν τα δεδομένα και είναι πολύ πιο ευδιάκριτες σε σύγκριση με άλλες. Ακόμη, ανεξάρτητα από το πώς μπορεί να αλλάξει μια χρονοσειρά, αυτό πρέπει να συνεχίσει να κάνει την δουλειά του. [9]

Δεν υπάρχει μόνο ένας τρόπος υπολογισμού απόστασης. Αυτό είναι καλό, καθώς το καθένα χρησιμοποιείται και σε διαφορετικές εφαρμογές. Αναλυτικότερα, περιλαμβάνονται εκείνοι που λειτουργούν χρησιμοποιώντας τα χαρακτηριστικά (feature-based), το σχήμα (shape-based), τη συμπίεση (compression-based), το μοντέλο (model-based) και την επεξεργασία (edit-based). Στην πρώτη οικογένεια, με τα χαρακτηριστικά περιέχεται ο διακριτός Φουριερ μετασχηματισμός και ο διακριτός μετασχηματισμός κυματιδίων. Στην δεύτερη, με το σχήμα διαθέτει την δυναμική χρονική παραμόρφωση και την την Ευκλείδεια απόσταση. Ακόμη, σε αυτή είναι που χρονοσειρές υποβάλλονται σε παρατήρηση όσων χαρακτηριστικών πάνω στο γράφημα έχουν την ίδια μορφή Στην τρίτη, με τη συμπίεση υπάρχει μια που είναι γνωστή ως μέτρο ανισότητας. Όσο για την τέταρτη, με τη δομή, διαθέτει τα ARMA.Τέλος, αυτοί με την επεξεργασία έχουν την απόσταση Levenshtein [9]

Τα κυριότερα μέτρα απόστασης στις χρονοσειρές είναι:

Μινκόφσκι απόσταση (Minkowski distance). Ο τύπος που αναπαριστά αυτή την απόσταση στο 3.2.2.1, χρησιμοποιείται σαν βάση για τον υπολογισμό των υπολοίπων που θα αναφερθούν. Πιο συγκεκριμένα, της Τσέμπιεσεβ, της Ευκλείδειας.και της Μανχάταν. Μάλιστα, η μεταβλητή που είναι

υπεύθυνη για αυτή την αλλαγή είναι η p. Ένας εντελώς άλλος τύπος μετρικής προκύπτει από την μεταποίηση της τιμής αυτής. Για παράδειγμα, αν προσεγγίζει το άπειρο βγαίνει η Τσέμπιεσεβ, αν γίνει ένα, τότε γίνεται η Μανχάταν, αντίστοιχα αν γίνει δύο τότε υπάρχει η Ευκλείδεια. [20]

Ο τύπος της Μινκόφσκι:

$$d_{Mink}(S,Z) = \sqrt[p]{\sum_{i=1}^{m} |S_i - Z_i|^p} \qquad (3.2.2.1)$$

Ευκλείδεια απόσταση (Euclidean distance).Η συγκεκριμένη προκύπτει από το να πάρει κανείς οντότητες μέσα σε ένα χώρο και να χαράξει την πορεία που τους χωρίζει. Εννοείται βέβαια πως η διαδρομή αυτή είναι ίσια. Η ζητούμενη απόσταση προκύπτει από την εύρεση του πόσο απέχει η μια από το άλλη.[20] Ακόμη, ανήκει στην οικογένεια μετρικών που λειτουργούν χρησιμοποιώντας το σχήμα. Επίσης, λειτουργεί ακόμα και όταν υπάρχει όγκος στοιχείων. Δηλαδή, δεν πτοείται. Αν και αποτελεί μια μετρική όπου χρησιμοποιείται σε διάφορα πράγματα, έχει κάποια σημαντικά μειονεκτήματα. Όπως ότι έχει ευαισθησία σε τιμές όπου είναι πολύ ξεχωριστές, από την άποψη ότι σε σύγκριση με τις υπόλοιπες, εμφανίζουν σημαντική διαφορά. [9]

Παρακάτω δίνεται ο τύπος της Ευκλείδειας απόστασης:

$$d_{Euclidean}(S,Z) = \sqrt[2]{\sum_{i=1}^{m} |S_i - Z_i|^2} \qquad (3.2.2.2)$$

Μανχάταν απόσταση (Manhattan distance). Έχει μια λίγο διαφορετική αντίληψη όσον αφορά το πόσο απέχουν δύο σημεία μεταξύ τους. Στόχος είναι η άφιξη σε έναν προορισμό, δεδομένου ότι το ξεκίνημα έγινε από μια άλλη περιοχή. Στο τέλος, για την ακριβή καταγραφή όλης αυτής της διαδρομής, χρησιμοποιούνται τα μπλοκ. Δηλαδή, συγκροτημάτων κτιρίων που χωρίζονται μεταξύ τους από δρόμο. Γι' αυτό κιόλας η μετρική αυτή είναι γνωστή και ως απόσταση αστικού τετραγώνου(city-block instance).[20]

Ο τύπος της απόστασης Manhattan:

$$d_{Manhattan}(S,Z) = \sqrt{\sum_{i=1}^{m} |S_i - Z_i|} \qquad (3.2.2.3)$$

Δυναμική χρονική παραμόρφωση (Dynamic Time Warping). Αν και πολύ καλή σε αυτό που κάνει, έχει ένα βάρος όσον αφορά την εκτέλεση, που βελτιώνεται χάρη στην μεθόδο γνωστή ως άνω και κάτω περίβλημα. Προήλθε το έτος 2005 από τους Keogh και Ratanamahatana. Έχει σαν στόχο το να βρει ποια είναι η καλύτερη συσχέτιση μεταξύ χρονοσειρών. Προκειμένου να επιτευχθεί αυτό, το πιο σημαντικό είναι ο υπολογισμός ενός χώρου που κρατάει τις αποστάσεις των σειρών (distance matrix). Για να φτάσει κανείς από την αρχή μέχρι και το τέλος του υπάρχουν διάφοροι τρόποι. Καθένας από

αυτούς, αποτελεί το γνωστό μονοπάτι αλλοίωσης (warping path) Ωστόσο, αυτή που επιλέγεται είναι η πιο μικρή. Εφόσον η πορεία αυτή στην ουσία είναι μια ακολουθία τιμών, είναι η προτιμότερη γιατί αποτελείται από χαμηλές τιμές. [9]

3.3 Μείωση Διαστάσεων ή Αναπαραστάσεις Χρονοσειρών

Στόχος είναι το να μην χαθεί το πρότυπο σχήμα της χρονοσειράς. Δηλαδή να διατηρηθεί όσο γίνεται η αρχική πληροφορία. Το να επιτευχθεί αυτό, ενώ ταυτόχρονα το μέγεθος της μειώνεται είναι μια πρόκληση. Αποτελεί απάντηση στο ζήτημα που είναι γνωστό ως κατάρα της διαστατικότητας (dimensionality curse). Αναφέρθηκε από το 2001 από τον Bohm. Κάτι που συνδέεται στενά με την διαρκή αύξηση της εισροής δεδομένων. Συνεπώς ο κατάλληλος χειρισμός της μεγάλης διάστασης των δεδομένων είναι αρκετά σημαντικός.

Οι τεχνικές που θα εξηγηθούν παρακάτω στην συνέχεια αυτής της ενότητας, αφορούν προσεγγίσεις για την επίλυση αυτού του καίριου θέματος σε χρονοσειρές.

Πιο συγκεκριμένα αυτές είναι ο διακριτός μετασχηματισμός Φούριερ (Discrete fourier transform), η Διακριτή μετατροπή κυματισμών (Discrete Wavelet Transform), η αποσύνθεση μεμονωμένων τιμών (Singular value decomposition) και η συμβολική προσέγγιση αθροίσματος (Symbolic aggregate approximation). [9]

3.3.1 Discrete Fourier Transform (DFT)

Όταν παρατηρηθεί ότι μπορεί να αναπαρασταθεί με ημιτονοειδής όρους μια σειρά, αφού πρώτα εξεταστεί, και μετά διασπαστεί σε ένα συνδυασμό αυτών, αυτό σημαίνει ότι υπάρχει ανάλυση Φούριερ. Απομονώνοντας τις παραμέτρους για κάθε έναν από αυτούς, μένει ο διακριτός μετασχηματισμός Φούριερ. Στην ουσία στόχος της είναι να καταλήξει στο τέλος με μικρότερα κομμάτια. Αυτά προκύπτουν, αφού η αρχική χρονοσειρά έχει υποστεί ένα σπάσιμο. Μάλιστα, αν κάποιος δει σχηματικά αυτά τα κομμάτια, θα καταλάβει ότι παρουσιάζουν μια πολύ συγκεκριμένη μορφή που προσεγγίζει στενά την ημιτονοειδή. Ωστόσο, δεν τίθεται κάποιος περιορισμός όσον αφορά το αν αυτά θα είναι ημιτονοειδή ή κάτι άλλο, είναι κάτι που εξαρτάται από το εκάστοτε πρόβλημα. Υπάρχουν αρκετοί λόγοι που γίνεται συγκεκριμένα η επιλογή των ημιτονοειδών. Όπως ότι οποιεσδήποτε μεταποιήσεις γίνουν, έχουν μεγάλη ανθεκτικότητα. Πιο συγκεκριμένα αυτές αφορούν τον χρόνο.[22]

Σε γενικές γραμμές, ο μετασχηματισμός Φούριερ είναι υπεύθυνος για τη μετατροπή των συναρτήσεων χρόνου στις αντίστοιχες συχνότητας. Το ανάποδο επίσης ισχύει και πραγματοποιείται από τον αντίστροφο μετασχηματισμό Φούριερ. Αυτό συμβαίνει επειδή αυτά τα δύο στοιχεία είναι στενά συνδεδεμένα μεταξύ τους. [19]

Η ανάγκη χρήσης του διακριτού μετασχηματισμού Φούριερ προέκυψε από μια σημαντική μετάβαση. Εξαιτίας της εξελιξης που έχει η τεχνολογία, οι καταγραφές σημάτων γίνονται ανά κάποιο συγκεκριμένο διάστημα χρόνου, σε αντίθεση με παλαιότερα που δεν υπήρχε αυτό. Δηλαδή γινόταν χωρίς διακοπές. Βέβαια, χάρη στον γρήγορο μετασχηματισμό Φούριερ (Fast Fourier Transform) αυξάνεται κατά πολύ η απόδοση του. Στα ποικίλα θέματα που προκύπτουν μέσω της υλοποίησης του διακριτού μετασχηματισμού, στοιχεία όπως της διαρροής, της δειγματοληψίας και του σφάλματος

περιτύλιξης βοηθούν αρκετά στον περιορισμό αυτών. Αυτό είναι επιτευκτό λόγω της συνέλιξης. Ωστόσο, χρονικά προηγείται η χρήση μιας μεθόδου παραθυροποίησης. Μέσω αυτής ενα μικρότερο κομμάτι του αρχικού χρησιμοποιείται, επειδή δεν εφαρμόζεται ο Φούριερ όταν το σήμα είναι συνεχές. Έπειτα, πολλαπλασιάζεται με την SHAH συνάρτηση που έχει σχέση με παλμούς. Στη συνέχεια πολλαπλασιάζοντας με τη SHAH συνάρτηση, επιτυγχάνεται η οριοθέτηση της συχνότητας. Εννοείται πως οι δύο αυτές συναρτήσεις είναι διαφορετικές μεταξύ τους. Αφού ολοκληρωθεί αυτό, σημαίνει ότι έχει φτάσει το κατάλληλο μέγεθος ώστε να μπορέσει να κρατηθεί από την μνήμη. Είναι δυνατή .Πλέον, μπορεί να υλοποιηθεί ο διακριτός μετασχηματισμός Φούριερ. Έτσι, το παράθυρο ευθύνεται για τη μετατροπή του διακριτού μετασχηματισμού από τον κλασικό Φούριερ. [19]

3.3.2 Discrete Wavelet Transform (DWT)

Επιδιώκει να μείνει στο τέλος με τις μικρότερες συνιστώσες μιας σειράς. Αυτό το επιτυγχάνει σπάζοντας την. Με αυτόν τον τρόπο, αντι να εφαρμοστεί ολόκληρη η χρονική σειρά για τις εφαρμογές εξόρυξης, υλοποιούνται μόνο αυτά τα τμήματα. Ακόμη, μετά από αυτό θα περίμενε κανείς ότι κάποια από τα κομμάτια θα έχαναν την αρχική τους πληροφορία, κάτι που δεν ισχύει. Η συγκεκριμένη μέθοδος που υλοποιείται γι' αυτή τη διαδικασία ονομάζεται διακριτός μετασχηματισμός κυματιδίων (Discrete Wavelet Transform). Οποιοσδήποτε κλάδος χρειάζεται αυτά που προσφέρει, είναι βέβαιο ότι εφαρμόζει αυτή τη μέθοδο. Όπως αυτός που διαχειρίζεται δεδομένα που έρχονται από σήματα. Το ίδιο ισχύει και για εκείνα που προέρχονται από τον κλάδο υγείας. Απ' ότι φαίνεται γίνεται με μεγάλη επιτυχία στα δεδομένα που αντλήθηκαν από το σύνολο ηλεκτρικών σημάτων της εγκεφαλικής δραστηριότητα ατόμων. Στόχος του είναι το να βγει το πόρισμα για το αν ένας κάποιος έχει επιληψία. Μάλιστα αξίζει να αναφερθεί ότι στο τέλος επιβεβαιώθηκε πως δεν είναι καθόλου άσχετα τα πορίσματα, δηλαδή όντως έχουν μια λογική. Εξήχθησαν χάρη στον κατηγοριοποιητή. Ωστόσο, δε θα μπορούσε να το κάνει αυτό, αν δεν του δίνονταν τα υποδεέστερα τμήματα. Αυτά προήλθαν μέσω της εφαρμογής της διακριτής μετατροπής κυματισμών. Παρέχει εφαρμογές και λύσεις σε ποικίλα θέματα. Ένα από τα κυριότερα είναι ότι σε σύγκριση με το αρχικό της μέγεθος μένει στο τέλος με λιγότερα Ακόμη, αφήνει την χρονοσειρά καθαρή από δεδομένα ανούσια.

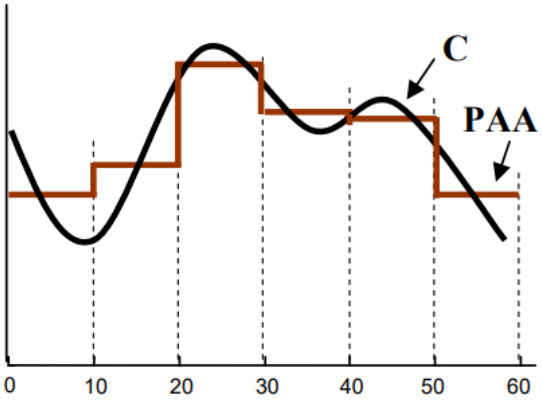
Κάποια ενδιαφέροντα χαρακτηριστικά που διαθέτουν τα τμήματα, είναι ότι παρά το γεγονός ότι προέρχονται από την ίδια χρονοσειρά και κάποια από αυτά δεν έχουν χάσει την αρχική πληροφορία τους, το καθένα δείχνει κάτι πιο αλλιώτικο από τα υπόλοιπα. Πιο συγκεκριμένα, αυτό που τα κάνει ιδιαίτερα είναι το ότι δείχνουν διαφορετικά μεταξύ τους μεγέθοι του κάθε πότε συμβαίνει κάτι. Στην ουσία είναι σαν να μπορεί να αναλυθεί μια χρονοσειρά σε διάφορα επίπεδα, κάθε ένα από τα οποία δείχνουν και κάτι εντελώς ξεχωριστό. Όσον αφορά το λίγο πιο πρακτικό κομμάτι, εννοείται πως τα μικρότερα τμήματα δεν προέκυψαν μόνα τους. Αντιθέτως, ευθύνονται εργαλεία που στηρίζονται στον τομέα των μαθηματικών. Είναι γνωστά ως κυματίδια (wavelets). Αυτό που προσφέρει η τεχνική αυτή των κυματιδίων και την κάνει να ξεχωρίζει από τους άλλους, είναι στο πόσο ευπροσάρμοστη είναι. Δηλαδή, ό,τι σήμα έρθει, θα λειτουργήσει εξίσου καλά.

Αφού δημιουργηθούν τα μικρότερα τμήματα, διατηρούνται μέσα σε αυτά τα στοιχεία της σειράς η οποία προήλθε. Με αυτόν τον τρόπο, δεν χάνεται αυτή η πληροφορία. Το θετικό είναι ότι στην πορεία, η σειρά αυτή μπορεί να φτιαχτεί πάλι. Αυτό είναι επιτευκτό από τον κατάλληλο συνδυασμό των επιμέρους τμημάτων. Επίσης χάρη σε αυτή την ιδιότητα που έχουν τα κομμάτια, βοηθάει στο ζήτημα του μεγέθους δεδομένων, καθώς οι διαδικασίες διερεύνησης μπορούν να υλοποιηθούν σε αυτά. Με τρόπο επιτυγχάνεται ελάττωση του αρχικού συνόλου. Ακόμη, σε μια σειρά επικρατεί μια χαοτική κατάσταση, καθώς περιλαμβάνει πολλά στοιχεία που μπορούν να εντοπιστούν. Η κατάσταση

γίνεται χειρότερη όταν δεν είναι και τόσο ευδιάκριτα. Στη σειρά ο χρόνος ταυτοποίησης μικρών ή μεγάλων μεταποιήσεων, μειώνεται σημαντικά χάρη σε αυτή τη μέθοδο. Στην ουσία, το καταφέρνει αυτό υποστηρίζοντας ότι αντί να προσπαθήσει να βγάλει κάποιος πορίσματα από την πρώτυπη, την διακρίνει από τις ιδιότητες στις οποίες φαίνονται μόνο οι τροποποιήσεις. Οι ιδιότητες αυτές, είναι ένα από τα δύο στοιχεία που προκύπτουν με την χρήση της διακριτής μετατροπής κυματισμών. Το άλλο δεν μπαίνει σε πολύ βάθος, αφορά κυρίως την συνολική εικόνα των δεδομένων. Είναι γνωστό και ως συντελεστής προσέγγισης. [23]

3.3.3 Piecewise Aggregate Approximation (PAA)

Η συγκεκριμένη μέθοδος, γνωστή και ως τμηματική προσέγγιση αθροίσματος, έχει κάποια ελαττώματα και προτερήματα. Τα κομμάτια της αρχικής χρονοσειράς που είχαν αξία μπορεί να πάψουν να υπάρχουν. Όπως φαίνεται και στο σχήμα 3.3.3.1, στην τελική χρονοσειρά που προκύπτει μετά την εφαρμογή του, αυτή με το κόκκινο χρώμα το επίπεδο λεπτομέρειας που θυσιάζεται είναι σημαντικό. Όμως αυτό εξισορροπείται κάπως, από την ικανοποιητική του απόδοση. Στόχος αυτής της τεχνικής είναι στο τέλος να φτάσει σε μια παράσταση που έχει προκύψει από ένα σύνολο μέσων τιμών. Κάθε μια από αυτές υπολογίζεται και από ένα ξεχωριστό τμήμα. Τα τμήματα αυτά είναι όλα ισομεγέθη. Μάλιστα, είναι το αποτέλεσμα σπασίματος της αρχικής χρονοσειράς. Ανάλογα με μέγεθος της μείωσης που θέλει να πετύχει κανείς, ορίζει και διαφορετική τιμή για το σε πόσα κομμάτια θα σπάσει η σειρά.[21]



Σχήμα 3.3.3.1 Αναπαράσταση Piecewise Aggregate Approximation [21]

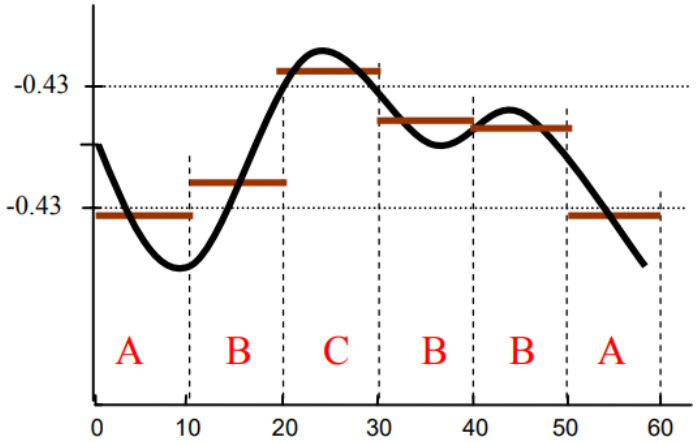
3.3.4 Symbolic Aggregate Approximation (SAX)

Ανήκει στην οικογένεια των μεθόδων που χρησιμοποιούνται για τη μείωση του μεγέθους μιας χρονοσειράς.[21] Μάλιστα η αποτελεσματικότητα της δεν συγκρίνεται.[9] Είναι γνωστή ως Συμβολική προσέγγιση αθροίσματος (Symbolic Aggregate Approximation). Όσο για τα προτερήματα και ελαττώματα της, τα υιοθετεί από την τμηματική συγκέντρωση αθροίσματος, της οποίας τη λογική χρησιμοποιεί για την υλοποίηση της. Πιο συγκεκριμένα ότι τόσο στη μια όσο και στην άλλη, παρά τις αυξημένες επιδόσεις που προκύπτουν, χάνεται αρκετή από την αρχική πληροφορία που υπήρχε στην χρονοσειρά. Πράγμα που δεν είναι καθόλου εύκολο, ειδικά όταν μπλέκονται στη μέση τομείς που απαιτούν να υπάρχει διεξοδική ανάλυση των δεδομένων. Γι’ αυτό λοιπόν έχει γίνει μια πρόταση με

σκοπό την βελτίωση αυτού του ζητήματος. Το μόνο που μένει ως έχει μεταξύ των δύο είναι η μέση τιμή. Η διαφορά με την κανονική εκδοχή είναι η ύπαρξη κάποιων στοιχείων. Το πρώτο είναι το μέγιστο. Όσο για το δεύτερο, είναι το ελάχιστο. [21]

Ο λόγος για τον οποίο μέσα στο όνομα της περιέχει τον όρο “συμβολική” είναι διότι στα τελικά τμήματα που προκύπτουν, το κάθε ένα μετατρέπεται σε ένα γράμμα.

Ένα παράδειγμα που δείχνει την λειτουργία της αναπαριστάται στο σχήμνα 3.3.4.1 όπου μέχρι ένα σημείο η λειτουργία είναι ολόιδια με αυτή της τμηματικής συγκέντρωσης αθροίσματος. Ξεκινάει από 60 το κανονικό μέγεθος της χρονοσειράς, την κόβει σε 6 κομμάτια, βρίσκει την μέση τιμή και έπειτα για το κάθε ένα γίνεται μετατροπή σε ένα γράμμα. Συνολικά υπάρχουν 3 γράμματα στα οποία μπορεί να γίνει η μεταποίηση.[21]



Σχήμα 3.3.4.1 Αναπαράσταση Symbolic Aggregate Approximation [21]

Ωστόσο εννοείται πως η μετατροπή του κάθε τμήματος και σε ένα γράμμα δεν είναι κάτι που γίνεται τυχαία. Γίνεται με βάση κάποιων τιμών που λέγονται οριακά σημεία. Στην ουσία αυτά προκύπτουν χάρη στην καμπύλη του γκάους. Πιο συγκεκριμένα, η επιφάνεια που δημιουργείται από αυτή, σπάει σε τόσα τμήματα όσα είναι και τα γράμματα που θέλουν να αναπαρασταθούν. Επειδή στην περίπτωση του παραδείγματος στο σχήμα 3.3.4.1 υπάρχουν 3 διαφορετικά σύμβολα, τότε αναλόγως θα υπάρχουν 3 μικρότερα μέρη. Από αυτά τα μέρη που προκύπτουν, τα στοιχεία που είναι σαν όρια μεταξύ αυτών είναι τα γνωστά οριακά σημεία. Επίσης, αυτά τα σημεία είναι πάντα ένας αριθμός κάτω από το σύνολο των γραμμάτων που είναι για απόδοση. Για παράδειγμα στο σχήμα 3.3.4.1 επειδή ο συνολικός αριθμός των γραμμάτων είναι 3, τότε αντίστοιχα τα σημεία διακοπής θα είναι 2. Το παραπάνω μπορεί να επιβεβαιωθεί κοιτώντας τον πίνακα 3.3.4.1, καθώς φαίνεται ότι για 3 στοιχεία υπάρχουν 2 οριακά σημεία. Εννοείται πως πέρα από το λίγο θεωρητικό κομμάτι, τα οριακά σημεία αντιστοιχούν και σε ορισμένους αριθμούς. Όλο αυτό έγινε δυνατό χάρη στη βοήθεια του πίνακα 3.3.4.1, όπου φαίνονται όλες οι πιθανές τιμές που θα μπορούσε να πάρει το κάθε οριακό σημείο, ανάλογα με την τιμή του α, δηλαδή του συνολικού αριθμού γραμμάτων. Ο τρόπος με τον οποίο μεταφράζεται το συγκεκριμένο πλαίσιο ώστε να γίνει σωστά η αντιστοιχία, βασίζεται στην τιμή του κάθε μεγέθους που θα προκύψει αφού έχουν χωριστεί τα τμήματα σε ίδια κομμάτια στην αρχική χρονοσειρά και έχει βρεθεί ο μέσος όρος για το καθένα. Η μετατροπή σε κάθε γράμμα γίνεται ανάλογα και με αυτή την τιμή. Σε Β αν το σημείο διακοπής β2 πέφτει πιο χαμηλά από αυτή την τιμή. Η μετατροπή θα γίνει σε Α, αν το πρώτο σημείο διακοπής β1 είναι πιο υψηλά από αυτή, όπως ακριβώς δείχνει και στο σχήμα 3.3.4.1. Με αντίστοιχο τρόπο γίνονται και τα υπόλοιπα.[21]

α	3	4	5
β1	-0.43	-0.67	-0.84
β2	0.43	0	-0.25
β3		0.67	0.25
β4			0.84

Πίνακας 3.3.4.1 Τιμές σημείων διακοπής [21]

3.4 Επίλογος

Η εξόρυξη δεδομένων στις χρονοσειρές παρέχει διάφορα σημαντικά καθήκοντα. Αυτά είναι η αναζήτηση περιεχομένου, η συσταδοποίηση, η πρόβλεψη, η ανακάλυψη μοτίβων, η κατηγοριοποίηση, η εύρεση κανόνων συσχέτισης, η εύρεση ανωμαλιών και η τμηματοποίηση. Στην ουσία, η ομοιότητα ανάμεσα σε χρονοσειρές είναι η απόσταση που έχουν. Αυτό είναι πάντα μια θετική τιμή. Ακόμη, η ταυτοποίηση της δεν χρειάζεται να είναι ακριβής. Γενικά υπάρχουν διάφορες μετρικές με τις οποίες καταγράφεται η απόσταση. Η πιο βασική είναι η οικογένεια Μινκόφσκι. Μέσα της υπάρχουν η Ευκλείδεια, η Μανχάταν και η Τσεμπιεσέβ. Ανάλογα με την τιμή του p στον τύπο της Μινκόφσκι, παίρνεται και μια διαφορετική μετρική απόστασης. Για p = 1 γίνεται η Μανχάταν, για p = 2 η Ευκλείδεια, ενώ για p = άπειρο η Τσεμπιεσέβ. Εξαιτίας της ραγδαίας αύξησης των δεδομένων, τα μεγέθη χρονοσειρών ολοένα μεγαλώνουν. Γι’ αυτό είναι επιτακτική η ανάγκη χρήσης τεχνικών ικανών να μειώσουν τις διαστάσεις. Όπως την εφαρμογή του διακριτού μετασχηματισμού Φούριερ, του διακριτού μετασχηματισμού κυματιδίων, της τμηματικής συγκέντρωτικής προσέγγισης και τέλος της συμβολικής συγκεντρωτικής προσέγγισης.

Κεφάλαιο 4ο: Αλγόριθμοι Matrix Profile

4.1 Εισαγωγή στο Matrix Profile

Παρά το γεγονός ότι προσδίδουν έναν φόρτο στο κομμάτι εφαρμογής τους, αυτά τα στοιχεία δεν χάνουν καθόλου την αξία τους. Όσον αφορά τις χρονοσειρές, βοηθούν στην υλοποίηση ποικίλων πραγμάτων. Το πρώτο είναι γνωστό ως προφίλ θέσης (Profile index). Από την άλλη, το δεύτερο ως προφίλ πίνακα (Matrix profile).

Ο λόγος που η ανάλυση του προφίλ πίνακα είναι τόσο σημαντική, είναι εξαιτίας των πορισμάτων που μπορούν να αντληθούν από αυτόν Ανάλογα με το που βρίσκονται στο διάγραμμα του, διακρίνονται σε δύο κατηγορίες:

- Η ασυμφωνία (discord). Είναι αρκετά πιθανό να υπάρχει εκεί που παρατηρείται μεγάλη αύξηση.
- Το μοτίβο (motif). Με αντίστοιχο τρόπο, βρίσκεται στις περιοχές όπου πέφτει πολύ το γράφημα.

Η ανακάλυψη των παραπάνω είναι δυνατή εξαιτίας μιας ιδιότητας που παρουσιάζει το προφίλ πίνακα. Ότι προσεγγίζει τη μορφή μιας ξεχωριστής ακολουθίας που κινείται μέσα στο χρόνο.[29] Ένας από τους πολλούς τομείς που επωφελούνται άμεσα από την εύρεση τους είναι αυτός που ασχολείται με την ανάλυση δεδομένων που αφορούν κλιματικές καταστάσεις. Μέσα στους ποικίλους τρόπου που βοηθάει, περιλαμβάνεται και η αναγνώριση συνθηκών που συνδέονται με σοβαρά περιστατικά. Σε συνδυασμό με τη γνώση που παρέχουν οι ασυμφωνίες, προκύπτει ένα πολύ ισχυρό εργαλείο για διάφορες εργασίες χρονοσειρών. [31]

Τόσο το προφίλ πίνακα (matrix profile) όσο και το προφίλ θέσης (profile index) είναι χώροι μέσα στους οποίους κρατιέται διαφορετική πληροφορία. Ωστόσο, συνδέονται στενά, καθώς το ένα συμπληρώνει το άλλο. Χωρίς το δεύτερο, δε θα μπορούσε να εξαχθεί μια σημαντική γνώση που παρέχει το πρώτο. Αυτή αφορά την ακριβή τοποθεσία του πιο κοντινού στοιχείου. Το προφίλ πίνακα περιέχει μέσα του, μεταξύ κάθε υποσειράς και της αντίστοιχης πιο όμοιας από την δεύτερη χρονική σειρά, τις τιμές των αποστάσεων. [29]

Σε γενικά πλαίσια δεν υπάρχει μόνο ένας τρόπος αντίληψης του προφίλ πίνακα. Στο παρακάτω κομμάτι, θα αναφερθεί μια λίγο πιο θεωρητική προσέγγιση της έννοιας του. Με βάση αυτήν, όσον αφορά υπολογισμό του, υπάρχει μια διαδικασία που πρέπει να γίνει. Πολύ σημαντικό ρόλο, παίζει ο εντοπισμός του προφίλ απόστασης (distance profile). Αυτό αποτελεί ένα μέρος που κρατάει τιμές που προήλθαν από την Ευκλείδεια μετρική.

Στόχος είναι η εύρεση για κάθε υποσειρά της πρώτης σειράς τις πιο κοντινές της στη δεύτερη. Εφαρμόζεται σε μια ομάδα που περιέχει ένα πλήθος με τις μικρότερες ακολουθίες μεταξύ δύο σειρών. Βασική προϋπόθεση, πρέπει όλες να έχουν σταθερή έκταση. Για να επιτευχθεί αυτό, γίνεται η χρήση ενός πλαισίου. Αρχίζει από την πρώτη υποσειρά της χρονοσειράς και αυξάνεται κατά μια θέση τη φορά. Ωστόσο το ερώτημα που προκύπτει, είναι πώς γίνεται αντιληπτό ποια είναι η εγγύτερη της στην

άλλη χρονική σειρά. Προκειμένου να απαντηθεί αυτό, πρέπει πρώτα να γίνουν ανάμεσα σε αυτά τα υποτμήματα, μετρήσεις που δείχνουν πόσο απέχουν. Μετά, καταλήγουν στο προφίλ απόστασης.

Για την εύρεση του προφίλ πίνακα, η γνώση των πλησιέστερων, που αναφέρθηκε πριν, είναι βασική. Αυτή η πληροφορία, σε συνδυασμό με το σε ποια υποσειρά αντιστοιχεί, εξάγεται από μια οντότητα γνωστή και ως σύνολο συνένωσης ομοιότητας (Similarity join set). Μέσα του περιέχει όλες αυτές τις συσχετίσεις μεταξύ των υποακολουθιών. Το προφίλ πίνακα (distance profile) προκύπτει, από την καταγραφή του μέτρου διαφοράς ανάμεσα σε αυτές.

Με βάση το τι αξιοποιείται για την δημιουργία του προφίλ πίνακα, προκύπτουν δύο είδη ομοιότητας:

- Η περίπτωση στην οποία χρησιμοποιείται μόνο μια χρονοσειρά. Άρα αυτό σημαίνει ότι οι υπολογισμοί της απομάκρυνσης θα γίνουν ανάμεσα στις υποσειρές της ίδιας. Το παραπάνω είναι γνωστό και ως σύνολο συνένωσης αυτό-ομοιότητας (self-similarity join set).

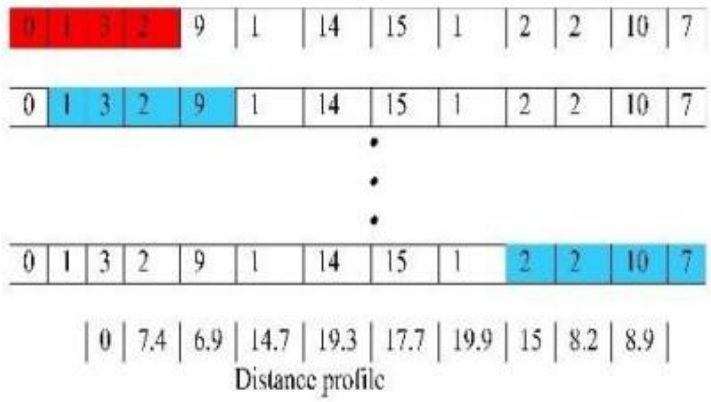
- Στην άλλη εκδοχή, όλο αυτό πραγματοποιείται ανάμεσα σε δύο χρονικές σειρές. Αυτό σημαίνει ότι το πόσο απέχουν, θα μετρηθεί μεταξύ των υποσειρών και των δύο.

Όπως και να έχει, δεν έχουν όλες οι τιμές την ίδια αξία. Μάλιστα, όταν είναι να βρεθεί το πόσο απέχουν τα μικρότερα τμήματα μιας χρονοσειράς, αυτές που αφήνονται εντελώς εκτός είναι εκείνες που προκύπτουν από την ίδια υποσειρά. Δηλαδή η απόσταση μιας υποσειράς από τον εαυτό της. [29]

Όλο αυτό που αναλύθηκε διεξοδικά παραπάνω, αποτελεί το ένα μέσο ερμηνείας του προφίλ πίνακα. Προκειμένου να γίνει λίγο πιο κατανοητή η υλοποίηση του άλλου, πρέπει πρώτα να αναφερθεί ένα στοιχείο. Αυτό ονομάζεται πίνακας απόστασης. Είναι ένας χώρος μέσα στον οποίο ως επί των πλείστον κρατούνται αποστάσεις με χρήση της Ευκλείδειας μετρικής. Κάθε τιμή μέσα σε αυτόν προσδιορίζει και το πόσο απέχουν μεταξύ τους υποακολουθίες δύο σειρών.

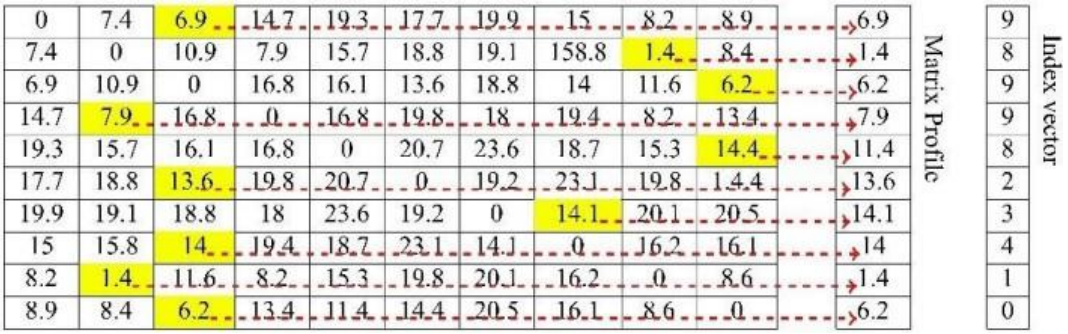
Αφού εξηγήθηκε η βασική έννοια, έφτασε η στιγμή της ανάλυσης του επόμενου σταδίου. Η κύρια διαφορά μεταξύ των δύο προσεγγίσεων, εντοπίζεται στο ποιες υποσειρές χρησιμοποιούνται για την εξαγωγή της απόστασης. Σε αντίθεση με την πρώτη, εδώ μετρούνται τιμές που δείχνουν το πόσο απέχουν μεταξύ τους όλες οι υποσειρές. Μετά στο σύνολο που προκύπτει, δεν παραμένουν όλα τα στοιχεία. Φεύγουν όλες, πέρα από εκείνες με τις πιο χαμηλές αποστάσεις. Αυτή προσέγγιση έχει ένα μεγάλο μειονέκτημα, ότι δεν είναι τόσο αποδοτική. Γι’ αυτό και δεν προτιμάται. [29]

Στο σχήμα 4.1.1 διακρίνεται το πώς προκύπτει το προφίλ απόστασης. Πιο συγκεκριμένα, από τις τέσσερις σειρές, η τελευταία αντικατοπτρίζει το τελικό αποτέλεσμα. Στόχος είναι σε αυτό να κρατηθεί η τιμή που βρίσκεται πιο κάτω σε σύγκριση με άλλες. Όπως έχει ήδη αναφερθεί, σε κάθε θέση του υπάρχει και ένα μέγεθος που συνδέεται με τη διαφορά που έχουν υποσειρές. Ακόμη, στη πρώτη σειρά, εκείνο το κόκκινο κομμάτι κάθε φορά που γίνεται η μέτρηση, ανεβαίνει κατά μια θέση. Αυτό φαίνεται και από το δεξί τμήμα του σχήματος. Μάλιστα, θα το κάνει αυτό από την αρχή μέχρι και το τέλος της σειράς. Επίσης, όσο γίνεται αυτό, αυτή που είναι στη δεύτερη σειρά μένει σταθερή.[31]



Σχήμα 4.1.1 Αναπαράσταση προφίλ απόστασης [31]

Στο σχήμα 4.1.2 φαίνεται περιγραφικά η όλη διαδικασία δημιουργίας του προφίλ πίνακα. Σχετίζεται πολύ με αυτά που προέκυψαν στο σχήμα 4.1.1. Επίσης, διακρίνονται τρεις πολύ βασικές οντότητες που έχουν αναφερθεί. Το προφίλ απόστασης, το προφίλ πίνακα και το προφίλ θέσης. Το κομμάτι που βρίσκεται τέρμα αριστερά δείχνει το προφίλ απόστασης, αφού έχει γίνει καταγραφή των πλήρη μετρήσεων. Μετά το προφίλ πίνακα είναι αμέσως επόμενο, στα δεξιά του, αποτελείται από τις πιο χαμηλές τιμές. Αυτό που δεξιά τελευταίο είναι το προφίλ θέσης.[31]



Σχήμα 4.1.2 Αναπαράσταση προφίλ πίνακα και θέσης [31]

4.2 Ο αλγόριθμος STAMP

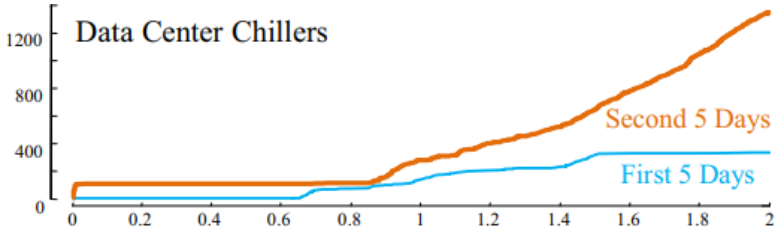
Ο αλγόριθμος ευρέως γνωστός ως STAMP υπολογίζει το προφίλ πίνακα[29] Σε μερικά ζητήματα ο αλγόριθμος που προτιμάται είναι ο STOMP. Χρησιμοποιείται σε κλάδους που έχουν μεγαλύτερες απαιτήσεις σε δεδομένα, συνεπώς χρειάζονται κάτι διαφορετικό. Αυτό το καταφέρνει με επιτυχία, καθώς σε σύγκριση με τον STAMP είναι πολύ πιο αποδοτικός. Ένας από αυτούς είναι και ο τομέας που μελετάει δεδομένα που αφορούν το έδαφος. Πιο συγκεκριμένα βοηθάει στην πρόβλεψη σεισμών.[30]

Η δημιουργία του STAMP δίνει λύση σε ένα πολύ σημαντικό θέμα. Αυτό εντείνεται στις χρονοσειρές, όπου από την φύση τους έχουν αρκετά στοιχεία. Το ζήτημα αφορά την εύρεση του πόσο απέχει μια υποσειρά με μια άλλη. Έχει να κάνει κυρίως με το ότι είναι μια δύσκολα εφαρμοστέα διαδικασία. Μάλιστα συνήθως παίρνει ένα μεγάλο χρονικό διάστημα για να εκτελεστεί. Αυτό καθιστά την εφαρμογή της σε οτιδήποτε σχετίζεται με την ανάλυση πολλών δεδομένων μη πρακτική. Έτσι αντί να περιμένει κάποιος για πολύ μέχρι να πάρει αποτελέσματα, του έρχονται μέσα σε λίγο χρόνο. Σίγουρα

αποτελεί μια καλή βελτίωση σε σύγκριση με πριν. Χάρη σε αυτό η εξαγωγή χρήσιμης πληροφορίας από τα δεδομένα καθιστάται μια πιο πρακτική διαδικασία.

Καταφέρνει να κάνει τα πάντα, από την αρχή εώς το τέλος, να τρέχουν αποδοτικά. Από εκεί που χρειαζόταν ένα μεγάλο χρονικό διάστημα, ξαφνικά με την υλοποίηση του ελαττώνεται σημαντικά. Επίσης, η διαδικασία είναι πιο απλή όταν το σύνολο δεδομένων είναι στατικό. Αυτό σημαίνει ότι η μικρότερη απόσταση θα εξάγονταν, κάνοντας υπολογισμούς μεταξύ υποσειρών. Ωστόσο, στην αντίθετη περίπτωση δημιουργείται μια ανησυχία. Ότι όταν αντλούνται πληροφορίες χωρίς διακοπή πως θα αλλάξει και το πώς κρίνεται ποια τμήματα μιας σειράς είναι παρόμοια. Σε αυτό ο αλγόριθμος υποστηρίζει ότι δεν χρειάζεται να γίνονται ολικές μεταποιήσεις της αρχικής έννοιας. Δηλαδή δεν είναι αναγκαίο να γίνονται όλες οι μετρήσεις από την αρχή. Ειδικά εφόσον υπάρχει ήδη μια βάση από προηγούμενες καταγραφές. Αυτό που προσφέρει ο STAMP λοιπόν είναι την ιδιότητα αναθεώρησης ώστε να μπορούν να γίνονται μικρές αλλαγές. Ακόμη, προτείνει την αφαίρεση της τιμής που δημιουργεί ένα καίριο θέμα. Προκύπτει από την ιδέα της σχηματικής συσχέτισης. Όταν δηλαδή παρατηρείται ότι τμήματα έχουν αντίστοιχη συμπεριφορά σε μια απεικόνιση της σειράς. Αυξάνεται η πολυπλοκότητα του θέματος, όταν καθορίζεται μια συνθήκη με βάση την οποία κρίνεται το παραπάνω. Ο λόγος πηγάζει από την ευαισθησία της τιμής που την ορίζει. Αυτό πρακτικά σημαίνει ότι ακόμα και όταν οι τιμές δεν έχουν μεταξύ τους μεγάλη διαφορά, θα επηρεαστούν πολύ τα αποτελέσματα. Βέβαια το συγκεκριμένο συμβαίνει κυρίως όταν γίνεται υλοποίηση της Ευκλείδειας. Το παραπάνω παρουσιάζεται στο σχήμα 4.2.1 Επιπλέον, ένα από τα προτερήματα του αλγορίθμου είναι ότι κάνει πολύ καλή διαχείριση της μνήμης μέσα στην οποία γίνεται η αποθήκευση. [29]

Ένα παράδειγμα στο οποίο μπορεί να υλοποιηθεί αυτό, είναι το εργοστάσιο. Όπως είναι λογικό, γίνεται κατακλυσμός δεδομένων σε ένα τέτοιο περιβάλλον. Άρα η αρχική χρονική σειρά θα περιλαμβάνει μέσα της πολλά στοιχεία, που θα είναι ανάλογα με το μέγεθός της. Εδώ η εύρεση των υποτμημάτων με τη μικρότερη απόσταση παρέχει και μια αξιόλογη πληροφορία. Θα υπήρχε μεγάλη επιβράδυνση, αν γινόταν εφαρμογή της πιο απλής μεθόδου. Καθώς θα χρειαζόταν να μετρηθούν από όλες τις υποσειρές οι τιμές που δείχνουν το πόσο μακριά ή όχι είναι μεταξύ τους. [29]



Σχήμα 4.2.1 Αναπαράσταση γραφημάτων ενός συστήματος ψύξης [29]

Στο σχήμα 4.2.1 φαίνονται δύο γραφικές αναπαραστάσεις. Μετράνε την ίδια ποσότητα, απλά σε διαφορετικό χρόνο. Τα δεδομένα που χρησιμοποιούνται είναι ενός συστήματος ψύξης. Επίσης το συνολικό διάστημα μέσα στο οποίο έγινε η καταγραφή ήταν 10 ημερών. Ο άξονας χ'χ δείχνει την αλλαγή της τιμής της παραμέτρου, ενώ ο y'y δείχνει τις τελικές συσχετίσεις ομοιότητας. Όπως διακρίνεται, η μπλε γραμμή μετράει την ποσότητα αυτή για τις 5 μέρες στην αρχή. Από την άλλη, η πορτοκαλί αναλαμβάνει τις άλλες 5. Μάλιστα, παρουσιάζει και τη μεγαλύτερη αύξηση όταν τη τιμή γίνει 0.6. Για την ίδια τιμή, η μπλε έχει εντελώς διαφορετική συμπεριφορά. [29]

4.2.1 Περιγραφή και ανάλυση του αλγορίθμου

Προτού γίνει οποιοσδήποτε υπολογισμός, πρέπει να έχουν οριστεί οι μεταβλητές που θα χρησιμοποιηθούν. Ουσιαστικά εισάγεται μια προσωρινή τιμή που στην πορεία θα αλλάξει. Αυτή η ανάθεση είναι απαραίτητη για την προετοιμασία του αλγόριθμου. Ακριβώς αυτό θα γίνει με τις παραμέτρους που δείχνουν το προφίλ πίνακα (Pab) και το προφίλ θέσης (Iab) αντίστοιχα. Πιο συγκεκριμένα στην αρχή τους ανατίθενται οι τιμές άπειρο για κάθε θέση στην πρώτη παράμετρο. Από την άλλη στη δεύτερη παίρνει σε όλο το μήκος του μηδέν. Επίσης η μεταβλητή idxes που φαίνεται στο σχήμα 4.2.1.1 δείχνει το σύνολο υποσειρών της χρονοσειράς B. Όσο για το n_B αποτελεί τον αριθμό που δείχνει πόσο μεγάλη είναι η σειρά.

Αφού έχει ολοκληρωθεί το αρχικό στάδιο, μετά ακολουθεί αυτό στο οποίο πραγματοποιούνται οι μετρήσεις. Το idxes που αναφέρθηκε δείχνει πόσες φορές θα πραγματοποιηθεί η επανάληψη. Αυτό εξαρτάται άμεσα από την B. Ανάλογα με από πόσες υποακολουθίες περιλαμβάνει. Όσο για το idx, προσδιορίζει τον δείκτη θέσης, ο οποίος φανερώνει σε τι φάση βρίσκεται η επαναληψη.

Μετά υπολογίζονται όλες οι αποστάσεις μεταξύ του κάθε τμήματος της A και ενός συγκεκριμένου από την B. Όταν ολοκληρωθεί αυτό, το αποτέλεσμα αποθηκεύεται στον χώρο που κρατούνται αυτά. Δηλαδή στο προφίλ απόστασης.

Σκοπός είναι σε όλη τη διαδικασία να μένουν οι πιο μικρές αποστάσεις. Γι' αυτό τον λόγο οι θέσεις του προφιλ αποστασης συγκρίνονται με τις αντίστοιχες στο προφίλ πίνακα. Για παράδειγμα η 0 του ενός με την 0 του άλλου κτλ. Μετά, από το κάθε σύνολο δύο στοιχείων, παίρνεται ο μικρότερος αριθμός. Σε αυτή την περίπτωση είναι που θα γίνει αλλαγή στο προφίλ πίνακα με αυτή την τιμή. Πέρα από αυτό, διαφορετική τιμή παίρνει και αυτή της παραμέτρου Iab που είχε οριστεί στην αρχή. Πιο συγκεκριμένα, θα πάρει το ίδιο το idx. Ωστόσο, αν ήταν μεγαλύτερο, τότε θα έμεναν όπως έχει. Κάθε φορά που γίνεται πάλι η επανάληψη, ο δείκτης αλλάζει ανάλογα. Συνεπώς, θα γίνουν πάλι τέτοιες μετρήσεις, απλώς χρησιμοποιώντας διαφορετική υποσειρά της B. [29]

Procedure STAMP(T_A, T_B, m)
Input: Two user provided time series, T_A and T_B, interested subsequence length m
Output: A matrix profile P_{AB} and associated matrix profile index I_{AB} of T_A join T_B, $\boldsymbol{J_{AB}} = \boldsymbol{A} \bowtie_{\theta 1nn} \boldsymbol{B}$

```
1  n_B ← Length(T_B)
2  P_AB ← infs, I_AB ← zeros, idxes ← 1:n_B-m+1
3  for each idx in idxes    // In any order, but random for anytime algorithm
4        D ← MASS(B[idx], T_A)    // [12]
5        P_AB, I_AB ← ElementWiseMin(P_AB, I_AB, D, idx)
6  end for
7  return P_AB, I_AB
```

Σχήμα 4.2.1.1 Αναπαράσταση αλγόριθμου STAMP[29]

4.2.2 Λειτουργία του MASS και του Sliding Dot

Μια πολύ καλή προσέγγιση στο ζήτημα της απόστασης, υλοποιείται από τον MASS. Είναι γνωστός ως αλγόριθμος εύρεσης ομοιότητας του MUEEN. Χάρη σε αυτό, επιτυγχάνονται υλοποιήσεις σε ποικίλες εφαρμογές. Όπως στην εύρεση shapelets. Αν και προτάθηκαν διάφοροι μέθοδοι για την ταυτοποίηση παρόμοιων στοιχείων, δεν είναι όλοι το ίδιο ικανοί. Μάλιστα, σε αυτή την κατηγορία κατατάσσονται και οι μέθοδοι ευρετηρίασης (indexing methods). Ένα από τα πλεονεκτήματα που διαθέτει είναι ότι θα βγάλει αποτελέσματα οτιδήποτε δεδομένα κι αν λάβει. Το παραπάνω αναδεικνύει την αποδοτικότητα του. Τα δεδομένα χρονοσειρών εμφανίζουν συνήθως αυξημένη πολυπλοκότητα. Ειδικά όταν δεν είναι και τόσο μικρά, το ζήτημα εντείνεται. Συνεπώς, υπάρχει ένας προβληματισμός στο κομμάτι της αντιμετώπισης τους, στο οποίο παρέχει λύσει ο MASS.

Εστιάζει στη βελτίωση του προφίλ απόστασης έχοντας σαν στόχο την ελάττωση του χρόνου μετρήσεων τιμών, που δείχνουν πόσο μακριά είναι υποσειρές μεταξύ τους.

Μεγάλο ρόλο σε αυτό έχει η χρήση της μεθόδου κινούμενου εσωτερικού γινομένου (Sliding dot product). Η λειτουργία της στηρίζεται πάνω στην υλοποίηση του γρήγορου μετασχηματισμού Φουριερ (FFT). Σε αυτό οφείλει την επίτευξη του αρχικού στόχου.

Το σχήμα 4.2.2.2 δείχνει ότι το πρώτο βήμα για την εκτέλεση του MASS, είναι η εφαρμογή του κινούμενου εσωτερικού γινομένου. Επιπλεον, ο STAMP χρησιμοποιεί τον MASS για την εύρεση του προφίλ πίνακα. Άρα βγαίνει το συμπέρασμα ότι τα τρία αυτά στοιχεία είναι αλληλένδετα.[41]

Procedure SlidingDotProduct(Q, T)	
Input: A query Q, *and* a user provided time series T	
Output: The dot product between Q and all subsequences in T	
1	$n \leftarrow$ Length(T), m $\leftarrow$ Length(Q)
2	$T_a \leftarrow$ Append T with n zeros
3	$Q_r \leftarrow$ Reverse(Q)
4	$Q_{ra} \leftarrow$ Append Q_r with $2n$-m zeros
5	$Q_{raf} \leftarrow$ FFT(Q_{ra}), $T_{af} \leftarrow$ FFT(T_a)
6	$QT \leftarrow$ InverseFFT(ElementwiseMultiplication(Q_{raf}, T_{af}))
7	**return** QT

Σχήμα 4.2.2.1 Αναπαράσταση SlidingDotProduct [41]

Τελικό βήμα του Sliding dot, είναι η υλοποίηση αντίστροφου Φούριερ στο προϊόν μεταξύ δύο σειρών που μετασχηματίστηκαν από τον γρήγορο μετασχηματισμό Φουριερ. Για να φτάσει μέχρι εκεί, υπάρχουν κάποια στάδια. Αρχικά, η υλοποίηση του Sliding dot προϋποθέτει την εισαγωγή δύο παραμέτρων. Μια από αυτές είναι η υποσειρά που θα χρησιμοποιηθεί. Η δεύτερη είναι η χρονοσειρά. Αφού έχουν τεθεί αυτά μπορεί να ξεκινήσει η διαδικασία. Αρχικά εξάγονται τα μεγέθη και των δύο παραμέτρων. Σκοπός είναι να έρθουν σε μια συγκεκριμένη μορφή. Αυτή περιλαμβάνει την εισαγωγή μηδενικών. Τοποθετούνται στο τέλος και των δύο. Όσον αφορά την χρονοσειρά, ο αριθμός είναι συνδεδεμένος με το μήκος της. Στην περίπτωση της υποσειράς, πρέπει πρώτα να γυρίσει ανάποδα. Δηλαδή, όλα της τα στοιχεία να πάρουν αντίθετη κατεύθυνση. Μετά, επισυνάπτονται στο τέλος της ποσό μηδενικών που προκύπτει από μια μαθηματική πράξη. Αυτή αναφέρεται στην αφαίρεση του μήκους υποακολουθίας από το μέγεθος της αλλαγμένης σειράς. Έπειτα στο καθένα από αυτά γίνεται η

εφαρμογή του γρήγορου μετασχηματισμού Φούριερ. Μέσω αυτού, πλέον μπορεί να πραγματοποιηθεί αντιστροφος Φούριερ, όπως αναφέρθηκε στην αρχή. [41]

Όπως έχει ήδη αναφερθεί υπάρχει στενή σύνδεση ανάμεσα στον MASS και στο SlidingDotProduct. Το καθήκον του αλγορίθμου, είναι η μέτρηση του προφίλ απόστασης. Αναλυτικότερα, θέλει να κάνει τη διαδικασία πιο αποδοτική. Αυτός είναι ο τύπος:

$$D[i] = \sqrt{2m\left(1 - \frac{QT[i] - m\mu_Q M_T[i]}{m\sigma_Q \Sigma_T[i]}\right)} \qquad (4.2.2.1)$$

Μέσα από αυτόν διακρίνονται ποια είναι τα στοιχεία που χρειάζονται για τον μέτρηση του. Για παράδειγμα το MT, το QT, το mσ κτλ. Όλα αυτά προέρχονται από την υλοποίηση κάποιων μεθόδων.

Το QT προκύπτει από το SlidingDotProducts. Αποτελεί το προϊόν που αναφέρθηκε πριν. Αντίστοιχα από την ComputeMeanStd πηγάζουν τα μQ, σQ, MT και ΣT. Όλα αποτελούν στατιστικούς όρους. Λόγω της συγκεκριμένης μεθόδου, γίνεται καλύτερη η παραγωγή των παραπάνω. Από αυτά, τα ΣT και σQ αφορούν τυπική απόκλιση χρονοσειράς και υποσειράς. Τα MT και μQ δείχνουν την μέση τιμή των ίδιων. Όταν παραχθεί το D, ο χώρος του θα είναι γεμάτος από Ευκλείδειες αποστάσεις. [41]

Procedure MASS(Q, T)
Input: A query Q, *and* a user provided time series T
Output: A distance profile D of the query Q

```
1 QT ← SlidingDotProducts(Q, T)
2 μ_Q, σ_Q, M_T, Σ_T ← ComputeMeanStd(Q, T)
3 D ← CalculateDistanceProfile(QT, μ_Q, σ_Q, M_T, Σ_T)
4 return D
```

Σχήμα 4.2.2.2 Αναπαράσταση MASS [41]

4.3 Ο αλγόριθμος Brute Force

Σε σύγκριση με άλλους αλγορίθμους όπως ο STAMP, ο brute-force υστερεί αρκετά. Γενικά αποτελεί μια μέθοδο με σημαντικά χαμηλή αντοχή στα πολλά δεδομένα. Παρουσιάζει μεγάλη καθυστέρηση στην παραγωγή αποτελεσμάτων. Κυμαίνεται στο O (n^2m) από την υλοποίηση του προφίλ πίνακα. Το n συμβολίζει το μήκος της σειράς και το m του παραθύρου. Συνεπώς, όσο πιο μεγάλο είναι το n, τόσο αυξάνεται η τιμή του O. Αυτό δεν είναι επιθυμητό σαν αποτέλεσμα. Γι' αυτό και δεν προτιμάται σαν λύση. Επίσης, δεν τον καθιστά κατάλληλο για εφαρμογές με χρονοσειρές. Λειτουργεί εξάγοντας τα προφίλ απόστασης από όλες τις υποακολουθίες. Έπειτα σε αυτό αναζητά για τα ελάχιστα. Στο τέλος τα βάζει στον χώρο γνωστό ως προφίλ πίνακα. [32]

```
1   Function [dist, loc ]= Brute_Force( T, n)
2    best_so_far_dist = 0
3    best_so_far_loc = NaN
4
5   For p = 1 to |T| - n + 1                          // Begin Outer Loop
6     nearest_neighbor_dist = infinity
7     For q = 1 to |T| - n + 1                        // Begin Inner Loop
8       IF | p - q | ≥ n                              // non-self match?
9         IF Dist ([t_p,..., t_p+n-1], [t_q,..., t_q+n-1]) < nearest_neighbor_dist
10          nearest_neighbor_dist = Dist (t_p,..., t_p+n-1, t_q,..., t_q+n-1)
11        End
12      End                                           // End non-self match test
13    End                                             // End Inner Loop
14    IF nearest_neighbor_dist > best_so_far_dist
15      best_so_far_dist = nearest_neighbor_dist
16      best_so_far_loc = p
17    End
18  End                                               // End Outer Loop
19  Return[ best_so_far_dist, best_so_far_loc ]
```

Σχήμα 4.3.1 Αναπαράσταση Brute-force για την εύρεση ασυμφωνιών [33]

Όπως και στο προηγούμενο παράδειγμα εφαρμογής του στο προφίλ πίνακα, έτσι κι εδώ στην αναζήτηση ασυμφωνιών, ο αλγόριθμος παρουσιάζει αρκετά κακή επίδοση. Μάλιστα το θέμα γίνεται χειρότερο από την αύξηση των δεδομένων. Για την υλοποίηση του δεν αξιοποιεί άλλες διαδικασίες ή στοιχεία. Η λειτουργία του βασίζεται στην χρήση εμφωλευμένων επαναλήψεων. Μάλιστα, η τιμή p αφορά την υποσειρά που συγκρίνεται, ενώ το q τις υπόλοιπες αυτής. Στο πρώτο σκέλος εντοπίζει τις ελάχιστες αποστάσεις ανάμεσα σε υποσειρές. Έπειτα, από αυτές διαλέγει την πιο υψηλή. Εκεί είναι που βρίσκεται η ασυμφωνία.[33]

Στην εύρεση shapelets, η χρήση του εμφανίζει διάφορα ζητήματα. Πιο συγκεκριμένα, αφορούν την χρονική επιβάρυνση στην εκτέλεση του και την κακή εκμετάλλευση μνήμης. Σε αυτό το παράδειγμα, έχουν χωριστεί σε δύο κλάσεις, τα δεδομένα που χρησιμοποιούνται. Πιο συγκεκριμένα, στην B και A. Επίσης, πριν ξεκινήσει η διαδικασία πρέπει να υπάρξει κάποια αντίστοιχη προετοιμασία. Γι' αυτό και οι παράμετροι που θα χρειαστούν ορίζονται. Μια από αυτές είναι και η bsf_gain. Μέσα σε αυτή, κρατιέται το μεγαλύτερο όφελος. Η δομή του αλγορίθμου περιλαμβάνει μια συνθήκη επανάληψης. Αυτή εκτελείται για κάθε υποσειρά που είναι σε μια συγκεκριμένη λίστα. Μέσα της υπάρχουν υποακολουθίες με μήκοι που κυμαίνονται από μια τιμή μέχρι μια άλλη. Οι παραπάνω αριθμοί δίνονται από τον χρήστη.

Μετά υπολογίζεται ο επικρατέστερος για να γίνει shapelet. Ανάμεσα σε όλες τις υποσειρές της λίστας, πρέπει να βρεθεί ποια είναι η πιο ικανή. Η καταλληλότητα κρίνεται από μια σύγκριση ανάμεσα στα κέρδη πληροφορίας. Πιο συγκεκριμένα, γίνεται ανάμεσα στον υποψήφιο και σε αυτόν που θεωρείται καλύτερος. Στο τέλος, αυτός που είχε το πιο υψηλό. θα επιστραφεί.[16]

```
FindingShapeletBF (dataset D, MAXLEN, MINLEN)
1   candidates ← GenerateCandidates(D, MAXLEN, MINLEN)
2   bsf_gain ← 0
3   For each S in candidates
4       gain ← CheckCandidate(D, S)
5       If gain > bsf_gain
6           bsf_gain ← gain
7           bsf_shapelet ← S
8       EndIf
9   EndFor
10  Return bsf_shapelet
```

Σχήμα 4.3.2 Αναπαράσταση Brute-force για την εύρεση shapelets [16]

4.4 Εφαρμογές του Brute Force

Μια νέα πρόταση φέρνει ο αλγόριθμος skip-brute-force. Καταφέρνει να μην βρίσκει προσεγγιστικά τα πρότυπα. Παράλληλα, επιτυγχάνεται καλή διαχείριση του χρόνου από τη χρήση του. Επειδή δε θα ψάξει όλη τη σειρά μέχρι να βρει μοτίβα. Σε αντίθεση με τον brute force, επικεντρώνεται στα σημαντικά τμήματα. Η υλοποίηση του στηρίζεται στην αξιοποίηση του πλέγματος υπολογιστών. Μέσω αυτού δεν αναλαμβάνει ένας υπολογιστής το βάρος της εκτέλεσης. Αντιθέτως, μοιράζεται σε μονάδες. Κάθε μια θα έχει και ένα πόσο επεξεργαστικής δύναμης. Με βάση αυτό έγινε η δημιουργία του EUMEDGRID. Αποτελεί την πραγματική υλοποίηση του πλέγματος στη Μεσόγειο. Αυτό χρησιμοποιεί ο αλγόριθμος για την εκτέλεση του. Μπορεί να μην συνιστάται η χρήση του κλασικού brute-force εξαιτιας του χρόνου που παίρνει, αλλά έχει κάποια υπέρ. Αν του δοθεί μια υπακολουθια, βρίσκει όλες τις όμοιες με αυτή. Ακόμη και αν δεν είναι εντελώς ίδιες. Να έχουν δηλαδή μικρές διαφορές μεταξύ τους. Αυτό τον καθιστά κατάλληλο στην αναζήτηση μοτίβων σε δεδομένα βιολογίας. Πιο συγκεκριμένα, στον κλάδο που ασχολείται με την διερεύνηση αλληλουχιών DNA. Η εφαρμογή του brute force εδώ είναι αποτελεσματική. [34]

Σε σύγκριση με άλλες προσεγγίσεις, φαίνεται πως μια καλή επιλογή είναι η χρήση αυτού του αλγορίθμου μαζί με επίθεση λεξικού στο κομμάτι της εύρεσης κωδικών πρόσβασης. Η συγκεκριμένη μέθοδος δεν εγγυάται ότι θα βρει τον κωδικό. Όμως, αν είναι επιτυχής, τότε ο χρόνος που θα κάνει είναι μικρός. Από την άλλη,μέχρι να εντοπιστεί, η λειτουργία του δεν σταματάει από τον brute force Όταν αποτελείται από συνδυασμό συμβόλων, έχει μια παραπάνω δυσκολία. Επίσης, στους μικρούς κωδικούς παίρνει λίγο χρόνο μέχρι να εντοπιστούν.Συνεπώς, το αν είναι μεγάλος προσθέτει μια επιβάρυνση στην εκτέλεση του. Όλα τα παραπάνω που αναφέρθηκαν, βοηθούν άμεσα στην ενίσχυση της ασφάλειας. Είναι σημαντική η συμβολή στην βελτίωση της προστασίας κωδικών από διάφορες επιθέσεις. Γι' αυτό μέσω αυτής της διαδικασίας αναδεικνύει τα χαρακτηριστικά που τον καθιστούν ευάλωτο. [35]

Ο αλγόριθμος βρίσκει ποια είναι η διαδρομή που συμφέρει έναν πλανόδιο πωλητή. Αυτή δηλαδή που θα του φέρει περισσότερα χρήματα. Όπως επίσης να έχει λιγότερα συνολικά έξοδα. Έχει να επισκεφθεί κάποια μέρη. Συνεπώς, υπάρχουν διάφοροι συνδυασμοί σημείων που μπορεί να πάρει. Η μικρότερη από αυτές είναι η πιο αποδοτική. Βέβαια το ζήτημα περιπλέκεται, όταν ο αριθμός τους είναι πάνω από δέκα. Διότι οι μετρήσεις που πρέπει να κάνει αυξάνονται. Το συγκεκριμένο θέμα

παρουσιάζει μεγάλο ερευνητικό ενδιαφέρον. Συγκεκριμένα, στο κομμάτι της εύρεσης της καλύτερης διαδρομής. [42]

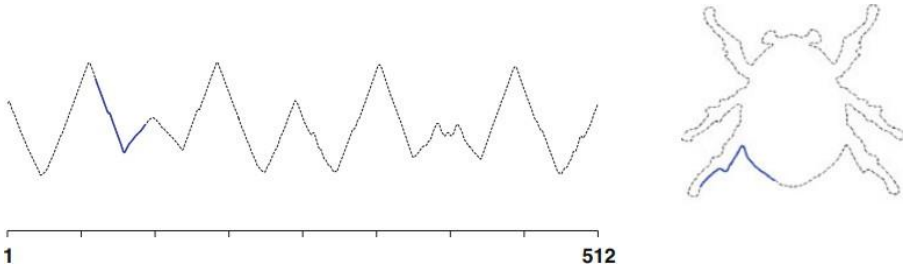
4.5 Επίλογος

Το Matrix Profile είναι ένα ισχυρό εργαλείο εύρεσης ομοιότητας ανάμεσα σε υποσειρές μεγαλύτερων σειρών. Αυτό επιτυγχάνεται υπολογίζοντας την Ευκλείδεια απόσταση των υποσειρών με τη χρήση ενός κυλιόμενου παραθύρου m που αυξάνεται κατά μια θέση τη φορά. Παρατηρώντας το Matrix Profile μπορούν να εξαχθούν ενδιαφέροντα πορίσματα. Στα σημεία όπου είναι αρκετά χαμηλά υπάρχουν μοτίβα, ενώ στα πολύ υψηλά ασυμφωνίες. Υπάρχουν διάφοροι αλγόριθμοι για την εύρεση του Matrix Profile. Ένας από αυτούς, ο STAMP είναι ο πιο αποδοτικός. Ωστόσο δε λειτουργεί το ίδιο καλά για πολύ μεγάλα σύνολα δεδομένων. Σε αυτή τη περίπτωση προτιμάται ο STOMP. Ο STAMP υλοποιεί τον MASS για την μέτρηση των αποστάσεων. Χάρη σε αυτόν, η εκτέλεση του STAMP γίνεται πιο γρήγορα. Μάλιστα,χρησιμοποιεί και το εσωτερικό γινόμενο για να το πετύχει αυτό και να βρει το προφίλ απόστασης. Ο αλγόριθμος Brute Force είναι η λιγότερο αποδοτική μέθοδος. Αποτελεί την πιο απλή εκδοχή εκτέλεσης κώδικα. Αυτό περιλαμβάνει την εύρεση των ασυμφωνιών, μοτίβων και των shapelets. Έχει την χειρότερη χρονική επίδοση σε σύγκριση με άλλους. Ωστόσο έχει κάποια θετικά. Όπώς για παράδειγμα, στον έλεγχο ανθεκτικότητας κωδικών πρόσβασης στη κυβερνασφάλεια, στο EUMEDGRID και στο πρόβλημα του πλανώδιου πωλητή.

Κεφάλαιο 5ο: Πειραματική Μεθοδολογία

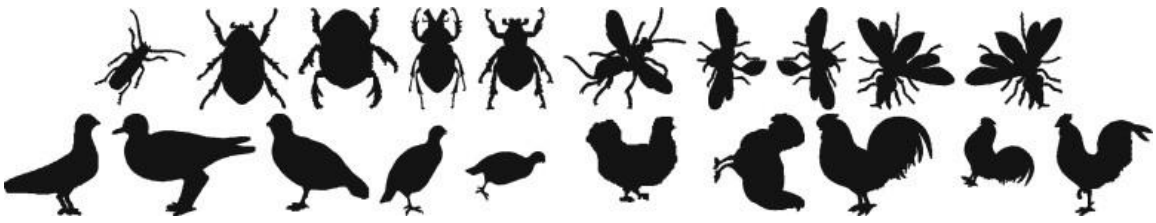
5.1 Σύνολο δεδομένων BeetleFly

Το σύνολο δεδομένων που χρησιμοποιείται στην εργασία αυτή ονομάζεται BeetleFly και βρίσκεται στην ιστοσελίδα www.timeseriesclassification.com. Τα δεδομένα αφορούν στην περιγραφή ενός συνόλου εντόμων με τη μορφή χρονοσειρών. Συγκεκριμένα, μία χρονοσειρά βγαίνει από τον μετασχηματισμό του περιγράμματος μιας μύγας ή ενός σκαθαριού. Στο σχήμα 5.1.1 φαίνεται η μετατροπή από το περίγραμμα ενός σκαθαρίου (δεξιά) σε μια χρονοσειρά (αριστερά).



Σχήμα 5.1.1 Μετατροπή από σχήμα σε χρονική σειρά [43]

Επίσης, το BeetleFly ανήκει στην ευρύτερη κατηγορία συνόλων που υλοποιούνται σε εφαρμογές ανάλυσης σχημάτων. Πέρα από το σκαθάρι και τη μύγα, διαθέτει πολλές άλλες κατηγορίες. Όπως για παράδειγμα πουλί και κότα. Στο σχήμα 5.1.2 φαίνονται τα στοιχεία αυτού του συνόλου. Η πάνω σειρά περιέχει ασπρόμαυρες εικόνες σκαθαριών στα αριστερά και μυγών στα δεξιά. Στην κάτω, αριστερά είναι τα πτηνά και δεξιά οι κότες. Στο επόμενο στάδιο μετατρέπονται σε χρονικές σειρές. Όλα αυτά τα σύνολα είναι μέρος μιας βάσης με όνομα MPEG-7 CE Shape-1 Part B. Υπάρχουν πολλοί μέθοδοι αναπαράστασης μορφής. Ο κάθε ένας επιτελεί και κάτι διαφορετικό. Μέσα στη βάση αυτή, τα περιεχόμενα χρησιμοποιούνται από τις MPEG-7 μεθόδους σαν αξιολόγηση. [43]



Σχήμα 5.1.2 Τα σύνολα δεδομένων Beetle/Fly και Bird/Chicken [43]

Το BeetleFly είναι χωρισμένο σε δύο σύνολα εκπαίδευσης και ελέγχου. Το κάθε ένα από αυτά περιέχει 10 χρονοσειρές που αντιστοιχούν σε μύγες και 10 χρονοσειρές που αντιστοιχουν σε σκαθάρια. Η κάθε μια χρονοσειρά είναι μονοδιάστατη μήκους 512 και συνοδεύεται από μια επιπλέον τιμή που δηλώνει την κλάση στην οποία ανήκει. Πιο συγκεκριμένα, όταν η τιμή είναι 1.0 αντιστοιχίζεται με το σκαθάρι, ενώ όταν είναι 2.0 με τη μύγα. Η τιμή αυτή είναι αποθηκευμένη στην πρώτη στήλη του κάθε συνόλου.

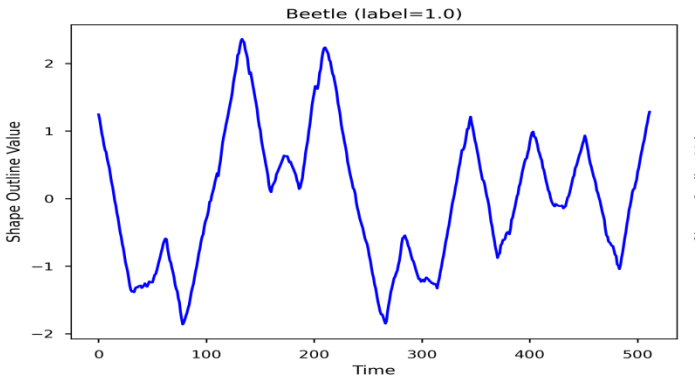
5.2 Προεπεξεργασία δεδομένων

Στην ιστοσελίδα κάθε σύνολο δεδομένων περιέχει μέσα του διάφορα αρχεία. Ένα από αυτά είναι για την εκπαίδευση του κατηγοριοποιητή. Το άλλο σχετίζεται με τον έλεγχο της λειτουργικότητας του. Αυτό που εξετάζει είναι το πόσο καλά προβλέπει. Βέβαια, το παραπάνω γίνεται σε μετέπειτα στάδιο. Το αρχείο για το BeetleFly έχει 20 γραμμές. Ωστόσο, επειδή αυτό είναι τύπου κειμένου χρειάστηκε να γίνει μια μετατροπή σε csv. Καθώς αυτό είναι πιο πρακτικό στη διαχείριση. Όπως φαίνεται και στον κώδικα της εικόνας 5.2.1, στη πρώτη γραμμή εισάγεται το αρχείο κειμένου. Μετά η βιβλιοθήκη Pandas που καλείται να το διαβάσει ως 'pd' το περνάει ως πίνακα. Η μεταβλητή στην οποία υπάρχει αυτό έχει όνομα 'df'. Έπειτα ο τύπος του κειμένου μετατρέπεται σε .csv. Αυτός είναι και ο τελικός στόχος.

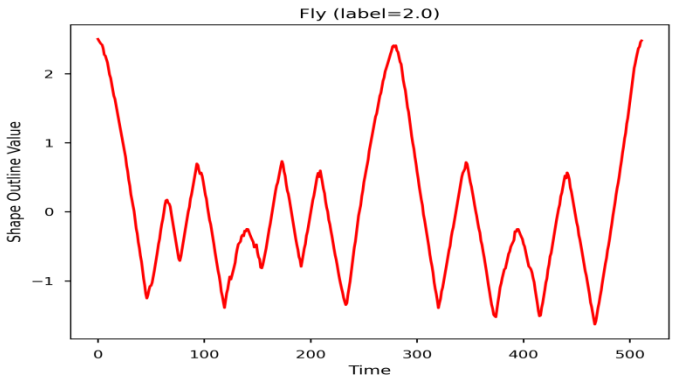
```
[3]: df = pd.read_csv('C:/Users/athen/Downloads/BeetleFly/BeetleFly_TRAIN.txt', sep=r'\s+', header=None)

     df.to_csv('C:/Users/athen/Downloads/BeetleFly/BeetleFly_TRAINNEW.csv', index=False)
```

Σχήμα 5.2.1 Μετατροπή από .txt σε .csv

Για τη σωστή λειτουργία της STUMPY πρέπει το dataset να είναι κανονικοποιημένο. Αυτό σημαίνει ότι το μέσος όρος των δειγμάτων πρέπει να είναι 0. Ακόμη, η τυπική απόκλιση να είναι 1. Σε αυτή την περίπτωση, το BeetleFly πληροί τα κριτήρια. Γι' αυτό και δεν έγινε κάποια προεπεξεργασία. Για να επιβεβαιωθεί αυτό έγινε χρήση της Pandas. Μέσω των συναρτήσεων mean() και std() βρέθηκαν οι παραπάνω τιμές.
Στο σχήμα 5.2.2, εμφανίζεται η γραφική αναπαράσταση ενός δείγματος σκαθαριού, ενώ στο 5.2.3 μιας μύγας.



Σχήμα 5.2.2 Οπτικοποίηση ενός δείγματος σκαθαριού



Σχήμα 5.2.3 Οπτικοποίηση ενός δείγματος μύγας

5.3 Εξαγωγή shapelets

Για το κομμάτι της υλοποίησης χρησιμοποιήθηκε η εξής διαδικασία:

1. Ο καθορισμός των απαραίτητων βιβλιοθηκών Python για τη διαχείριση και την ανάλυση των δεδομένων. Η εύρεση shapelets βασίζεται στη χρήση του προφίλ πίνακα (Matrix profile). Ο υπολογισμός του πραγματοποιείται από τη βιβλιοθήκη STUMPY. Για την υλοποίηση στο περιβάλλον Jupyter έγινε χρήση της Pandas, Numpy, Matplotlib, sklearn και SciPy. Η Matplotlib παρέχει τα κατάλληλα εργαλεία για την δημιουργία των οπτικών απεικονίσεων σειρών, όπως ορισμό ετικετών, τίτλων, κουκίδων και γραφημάτων. Στα πλαίσια της εργασίας η Pandas παράγει και διαχειρίζεται πλαίσια δεδομένων (Dataframes), - πίνακες με πάνω από δύο διαστάσεις. Οι πίνακες αυτοί γεμίζουν με στοιχεία που διαβάζονται από μια διαδρομή αρχείου. Η Numpy είναι φτιαγμένη για την επεξεργασία αριθμητικών δεδομένων σε έναν πίνακα, όπως τη μετατροπή ή την ταξινόμηση αυτών. Επίσης διαθέτει τη δημιουργία πινάκων τύπου numpy. Όσο για την sklearn, προκύπτει από την scikit-learn. Η συγκεκριμένη προσφέρει εργαλεία και τεχνικές που χρειάζονται στην κατηγοριοποίηση. Η SciPy, όπως η Numpy, χρησιμοποιείται για επιστημονικούς υπολογισμούς. Όλες οι βιβλιοθήκες που αναφέρθηκαν παίζουν βασικό ρόλο στην εύρεση shapelets. Τους δίνεται ένα εναλλακτικό όνομα για πρακτικούς λόγους, όπως, το όνομα np για τη Numpy ή το pd για την Pandas (Π1).

2. Στη συνέχεια εισάγεται το αρχείο που θα αποτελέσει το σύνολο δεδομένων εκπαίδευσης, το οποίο αποθηκεύεται σε ένα dataframe με τη χρήση της Pandas (Π2).
   Έπειτα, το σύνολο αυτό διασπάται σε δύο μικρότερα dataframes με βάση την κλάση. Η κλάση εξάγεται από την πρώτη στήλη του αρχείου. Αν η τιμή σε αυτή είναι 1.0, τότε η σειρά αντιστοιχεί σε σκαθάρι. Τα στοιχεία της υπόλοιπης σειράς αποθηκεύονται στο πρώτο dataframe. Ανάμεσα σε κάθε ακολουθία, προστίθεται μια τιμή NaN. Χάρη σε αυτό, η διάκριση ανάμεσα στις σειρές είναι ευκολότερη. Με ανάλογο τρόπο, οι σειρές που αντιστοιχούν στην κλάση 2.0 (μύγα) κρατιούνται στο δεύτερο dataframe (Π3).

3. Το επόμενο βήμα αφορά την εμφάνιση γραφημάτων. Πιο αναλυτικά, παρουσιάζονται δύο διακριτές γραφικές αναπαραστάσεις. Κάθε μια αντιστοιχεί σε ένα από τα dataframes που προαναφέρθηκαν. Το πρώτο από αυτά, του σκαθαριού, απεικονίζεται στο πάνω μέρος του συνολικού πλαισίου, ενώ της μύγας στο κάτω. Με αυτό τον τρόπο παρέχεται μια γενική εικόνα των σειρών. Ακόμη, έχει οριστεί γενικός τίτλος, αλλά και ξεχωριστοί για τους άξονες χ και y αντίστοιχα (Π4).

4. Ακολουθεί ο υπολογισμός δύο προφίλ πίνακα. Το πρώτο πηγάζει από το dataframe του σκαθαριού με τον εαυτό του. Το δεύτερο προκύπτει από αυτό του σκαθαριού με εκείνο της μύγας. Σημαντικός είναι ο ορισμός τιμής για το μήκος του παραθύρου m. Ο κατάλληλος αριθμός γι’ αυτό δεν είναι καταγεγραμμένος κάπου. Ωστόσο μπορεί να καθοριστεί εμπειρικά μετά από την εκτέλεση πειραμάτων. Εδώ χρησιμοποιήθηκε η τιμή 59.
   Αφού γίνει η εκτέλεση του προφίλ πίνακα για αυτά, από τις NaN προκύπτουν τιμές τύπου άπειρο. Πρέπει να μετατραπούν πάλι σε NaN καθώς μπορεί να προκληθούν θέματα (Π5).

5. Κατασκευάζεται η γραφική παράσταση των δυο προφίλ πίνακα σε ένα διάγραμμα (Π5).
   Συμπληρωματικά, υπολογίζεται η διαφορά Pdiff των δύο προφίλ πίνακα. Αυτού που

προέκυψε από τα dataframes του σκαθαριού και της μύγας μείον εκείνου από του σκαθαριού με τον εαυτό του. Μέσω της συνάρτησης find_peaks της SciPy βρίσκονται οι δέκα κορυφαίες τιμές σε αυτό. Στη συνέχεια αποθηκεύονται σε μια μεταβλητή θέσης idx. Τέλος δημιουργείται το γράφημα από την διαφορά και τις θέσεις. Χάρη σε αυτό, είναι πιο ευδιάκριτο που βρίσκονται τα shapelets (Π6).

6. Στη συνέχεια, αρχικοποιείται ένας πίνακας για τα shapelets σκαθαριού. Για κάθε τιμή μέσα στη μεταβλητή θέσης idx, από το dataframe σκαθαριού θα αποσπάται μια υποακολουθία μήκους m, η οποία αποτελεί ένα shapelet, και θα αποθηκεύεται στον πίνακα που αρχικοποιήθηκε προηγούμενως. Έτσι στην απεικόνιση του γραφήματος σκαθαριού, μπορούν να παρουσιάζονται και τα shapelets.

   Ο πίνακας αυτός περιλαμβάνει την πληροφορία που θα χρησιμοποιηθεί για την κατηγοριοποίηση των εντόμων (Π7).

5.4 Διαδικασία Κατηγοριοποίησης

Στην παρούσα ενότητα, περιγράφεται η διαδικασία κατηγοριοποίησης των εντόμων με τη χρήση των shapelets που έχουν εντοπιστεί. Πρέπει να αναφερθεί ότι το μήκος των shapelets αντιστοιχεί στην τιμή m. Ακόμη, τα shapelets αποτελούν στοιχεία που χαρακτηρίζουν τα αντικείμενα μιας κλάσης. Για παράδειγμα, ένα συγκεκριμένο shapelet μπορεί να είναι χαρακτηριστικό μιας μύγας, ενώ ένα διαφορετικό shapelet να είναι χαρακτηριστικό ενός σκαθαριού. Υπάρχουν πολλά υποψήφια τμήματα για να γίνουν shapelets. Όμως τα καλύτερα είναι αυτά που εμφανίζονται συχνότερα σε μια σειρά. Επίσης, πρέπει να είναι όσο πιο διαφορετικά γίνεται μεταξύ των δύο κλάσεων. Μάλιστα, αυτά είναι που παρουσιάζουν την μεγαλύτερη ικανότητα κατηγοριοποίησης (Π8).

Στην προηγούμενη ενότητα έχουν βρεθεί 10 shapelets, τα οποία θα ελεγχθούν για την ικανότητά τους στο διαχωρισμό των μυγών και των σκαθαριών. Η διαδικασία αυτή περιγράφεται στη συνέχεια.

1. Επιλέγεται ένα από τα 10 shapelets

2. Υπολογίζεται το προφίλ αποστάσεων μεταξύ του shapelet και κάθε υποακολουθίας για την πρώτη χρονοσειρά του συνόλου εκπαίδευσης. Αποθηκεύεται η μικρότερη απόσταση σε μια λίστα. Επαναλαμβάνεται η διαδικασία αυτή για τις υπόλοιπες χρονοσειρές του συνόλου εκπαίδευσης.Ο υπολογισμός των αποστάσεων πραγματοποιείται με χρήση της STUMPY και κλήση της MASS. Οι αποστάσεις δείχνουν το πόσο κοντά είναι το shapelet σε κάποια υποακολουθία κάθε χρονοσειράς.

3. Κατασκευάζεται ένα δέντρο αποφάσεων με βάση τη λίστα των αποστάσεων που υπολογίστηκαν στο προηγούμενο βήμα και την αντίστοιχη κλάση (ετικέτα). Με τον τρόπο αυτό υπολογίζεται μία τιμή διαχωρισμού σύμφωνα με την οποία μια χρονοσειρά θα κατηγοριοποιείται ως μύγα ή σκαθάρι ανάλογα με την απόσταση που υπολογίστηκε στο προηγούμενο βήμα.

4. Το δέντρο αποφάσεων αξιολογείται με βάση την ακρίβεια που παρουσιάζει στην κατηγοριοποίηση των χρονοσειρών του συνόλου ελέγχου. Εφόσον υπάρχουν 10 shapelets, θα κατασκευαστούν 10 δέντρα αποφάσεων και θα υπολογιστούν 10 ποσοστά ακρίβειας. Όσο πιο κοντά στο 100% είναι η τιμή της ακρίβειας, τόσο καλύτερος είναι ο κατηγοριοποιητής (shapelet).

5.5 Επίλογος

Για τις ανάγκες αυτής της πτυχιακής χρησιμοποιήθηκε το σύνολο δεδομένων BeetleFly. Βρίσκεται σε μια ιστοσελίδα που περιέχει διάφορα τέτοια σύνολα για κατηγοριοποίηση χρονοσειρών. Επειδή το συγκεκριμένο είναι τύπου εικόνας, για να γίνει χρονοσειρά ακολουθήθηκε μια διαδικασία. Αρχικά από τις εικόνες πάρθηκαν τα περιγράμματα με τη μορφή σκαθαριού και μύγας. Έπειτα, από αυτά έγινε ο μετασχηματισμός σε χρονοσειρά. Όλα τα σύνολα που βρίσκονται μέσα στην ίδια οικογένεια αποτελούνται από χρονοσειρές 512 μήκους, 20 σύνολο εκπαίδευσης και 20 σύνολο ελέγχου. Τια την εξαγωγή των shapelets έγινε χρήση των βιβλιοθηκών STUMPY για το matrix profile. Της Matplotlib για την οπτικοποίηση γραφημάτων. Η Pandas εφαρμόστηκε στη διαχείριση των dataframes. Η Numpy για τους αριθμητικούς υπολογισμούς. Όσο για την sklearn υλοποιήθηκε στο κομμάτι της κατηγοριοποίησης. Τέλος η SciPy για επιστημονικούς υπολογισμούς. Μετά από διάφορους πειραματισμούς, η τιμή m που επιλέχθηκε για το matrix profile ήταν η 59. Αφού βρεθούν τα shapelets για την κλάση beetle, υλοποιούνται στη διαδικασία κατηγοριοποίησης. Εφόσον υπάρχουν 10 shapelets, τόσοι θα είναι και οι κατηγοροποιητές δέντρων απόφασης. Εκεί είναι που γίνεται ο έλεγχος ακρίβειας τους.

Κεφάλαιο 6ο: Αποτελέσματα

6.1 Πειραματική επιλογή της παραμέτρου m

Η εύρεση του κατάλληλου m έρχεται με δύο τρόπους. Είτε με βάση τα δεδομένα, είτε από πειραματισμούς. Χάρη στις τιμές ακρίβειας, μπορεί να εξαχθεί μια γενική εικόνα για την ικανότητα κατηγοριοποίησης που διαθέτει το κάθε shapelet ανάμεσα στις κλάσεις beetle και fly. Παρακάτω δίνονται οι τιμές ακρίβειας για μερικά m που δοκιμάστηκαν:

Για m = 40 - οι πιο ανεβασμένη τιμή είναι εκείνη του πέμπτου shapelet με 0.8. Ακολουθεί του όγδοου με 0.7. Μετά του έβδομου και του δέκατο με 0.6 το καθένα.

```
Accuracy for shapelet 0 = 0.55
Accuracy for shapelet 1 = 0.45
Accuracy for shapelet 2 = 0.35
Accuracy for shapelet 3 = 0.5
Accuracy for shapelet 4 = 0.8
Accuracy for shapelet 5 = 0.4
Accuracy for shapelet 6 = 0.6
Accuracy for shapelet 7 = 0.7
Accuracy for shapelet 8 = 0.55
Accuracy for shapelet 9 = 0.6
```

Σχήμα 6.1.1 Αποτελέσματα ακρίβειας για m = 40

Για m = 50 - Οι μεγαλύτερες τιμές ήταν η 0.8, η 0.65 και η 0.6. Αυτό δείχνει ότι καλύτερη διακριτική ικανότητα έχει το πρώτο, το τρίτο και το τέταρτο. Όπως φαίνεται και στην εικόνα 6.1.2. Αντίστοιχα, το πιο χαμηλό σε τιμή θεωρείται το δεύτερο.

```
Accuracy for shapelet 0 = 0.8
Accuracy for shapelet 1 = 0.4
Accuracy for shapelet 2 = 0.65
Accuracy for shapelet 3 = 0.6
Accuracy for shapelet 4 = 0.55
Accuracy for shapelet 5 = 0.55
Accuracy for shapelet 6 = 0.5
Accuracy for shapelet 7 = 0.5
Accuracy for shapelet 8 = 0.5
Accuracy for shapelet 9 = 0.55
```

Σχήμα 6.1.2 Αποτελέσματα ακρίβειας για m = 50

Για m = 80 - Οι τιμές είναι πιο κάτω σε σύγκριση με το προηγούμενο. Μάλιστα κορυφαίες είναι οι 0.75, η 0.7 και η 0.75. Δηλαδή τα καλύτερα shapelets είναι το ένατο, το δεύτερο και το όγδοο. Αλλάζοντας το μήκος των shapelets, επηρεάζεται και η ακρίβεια.

```
Accuracy for shapelet 0 = 0.6
Accuracy for shapelet 1 = 0.7
Accuracy for shapelet 2 = 0.55
Accuracy for shapelet 3 = 0.75
Accuracy for shapelet 4 = 0.55
Accuracy for shapelet 5 = 0.5
Accuracy for shapelet 6 = 0.45
Accuracy for shapelet 7 = 0.65
Accuracy for shapelet 8 = 0.75
Accuracy for shapelet 9 = 0.35
```

Σχήμα 6.1.3 Αποτελέσματα ακρίβειας για m = 80

Για m = 59 - Αρκετές τιμές είναι σχετικά υψηλές. Τα καλύτερα είναι το έβδομο με 0.8 και το έκτο με 0.75. Πολλά έχουν το ίδιο ποσό 0.7. Πιο συγκεκριμένα, το πρώτο, δεύτερο, τρίτο, τέταρτο και ένατο.

```
Accuracy for shapelet 0 = 0.7
Accuracy for shapelet 1 = 0.7
Accuracy for shapelet 2 = 0.7
Accuracy for shapelet 3 = 0.7
Accuracy for shapelet 4 = 0.6
Accuracy for shapelet 5 = 0.75
Accuracy for shapelet 6 = 0.8
Accuracy for shapelet 7 = 0.6
Accuracy for shapelet 8 = 0.7
Accuracy for shapelet 9 = 0.35
```

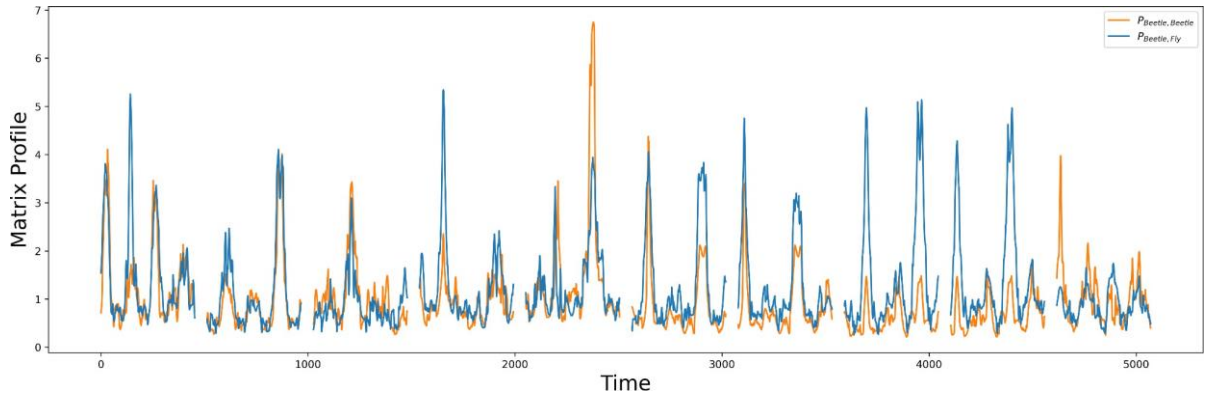
Σχήμα 6.1.4 Αποτελέσματα ακρίβειας για m = 59

Με βάση τις παραπάνω μετρήσεις, μπορούν να εξαχθούν κάποια συμπεράσματα. Όπως ότι όταν το m είναι 80, οι πιο μεγάλες τιμές για την ακρίβεια είναι το 0.75, 0.75 και 0.7. Για m ίσο με 40 η πρώτη τιμή είναι 0.8, ενώ οι άλλες 0.7 και 0.6. Αυτό δείχνει ότι κυμαίνονται κάπως ψηλά. Τέλος για m ίσον με 50, η μεγαλύτερη είναι η 0.8. Ωστόσο οι άλλες πάνε πολύ χαμηλά. Τέλος, όταν το m γίνει 59 υπάρχουν πολλά shapelets με ικανοποιητική τιμή ακρίβειας. Καθώς δύο από τις τρεις κορυφαίες τιμές είναι 8 και 0.75. Αυτό δείχνει ότι αρκετά shapelets έχουν ικανοποιητική ικανότητα κατηγοριοποίησης στις κλάσεις beetle και fly. Με βάση τα παραπάνω και για τις ανάγκες της παρούσας εργασίας, επιλέχθηκε το 59 ως τιμή για το m.

6.2 **Ανάλυση των shapelets του beetle dataframe**

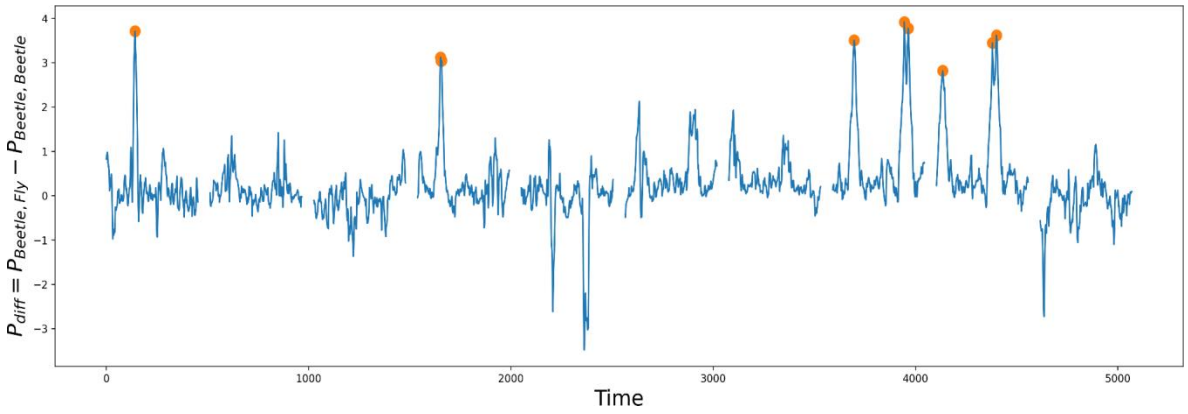
Η υλοποίηση του matrix profile είναι βασική στην εξαγωγή shapelets. Γι' αυτό βρίσκονται δύο διαφορετικά. Το πρώτο προκύπτει από την ένωση ομοιότητας του beetle dataframe με τον εαυτό του. Το δεύτερο, από το beetle και το fly data frame. Στην εικόνα 6.2.1 το Pbeetle,beetle συμβολίζεται με

πορτοκαλί, ενώ το άλλο με μπλε. Η τιμή m που χρησιμοποιήθηκε για την υλοποίηση τους είναι 59. Ένα πιθανό shapelet πρέπει να πληρεί κάποια κριτήρια. Αρχικά, την συχνή εμφάνιση στα υποτμήματα της κλάσης. Ακόμη, την ικανότητα διάκρισης από τα δείγματα της άλλης. Όπως αναφέρθηκε στο κεφάλαιο 4.1, οι χαμηλότερες τιμές στο matrix profile δείχνουν μοτιβα. Από την άλλη, οι πιο υψηλές αναπαριστούν ασυμφωνίες. Δηλαδή τιμές που είναι εκτός του κανονικού. Γι' αυτό τοποθετήθηκαν τα δύο matrix profile το ένα πάνω στο άλλο. Επειδή με αυτό τον τρόπο γίνεται ευκολότερη η αναγνώριση shapelets. Πιο συγκεκριμένα, γίνονται εμφανή στα σημεία με μεγάλη διαφορά. Αυτό προκύπτει όταν στο Pbeetle,beetle βρίσκονται χαμηλά το γράφημα. Ενώ στο Pbeetle,fly πηγαίνουν ψηλά. Στην 6.2.1 αυτό το φαινόμενο παρατηρείται στα τμήματα γύρω από το 4000. Επίσης στο σημείο που είναι κοντά στο 0. Ακόμη, εμφανίζεται σε αυτό που κυμαίνεται στο 1600-1700. Στα παραπάνω μέρη διακρίνονται οι μεγαλύτερες αποστάσεις μεταξύ των matrix profile. Σημαντικό ύψος έχει και το κομμάτι που βρίσκεται στη μέση. Αυτό ανάμεσα στο 2000 και στο 3000. Ωστόσο, το υψηλό Pbeetle,beetle δείχνει ότι αποτελεί ακραία τιμή για την κλάση. Συνεπώς, παρουσιάζει συμπεριφορά ασυμφωνίας. Επίσης, το μικρό Pbeetle,fly σημαίνει ότι έχει κακή ικανότητα στη διάκριση κλάσεων. Η ανάλυση της διαφοράς μεταξύ των matrix profile είναι αρκετά σημαντική. Διότι παρέχει μια ιδέα για το που βρίσκονται τα shapelets.



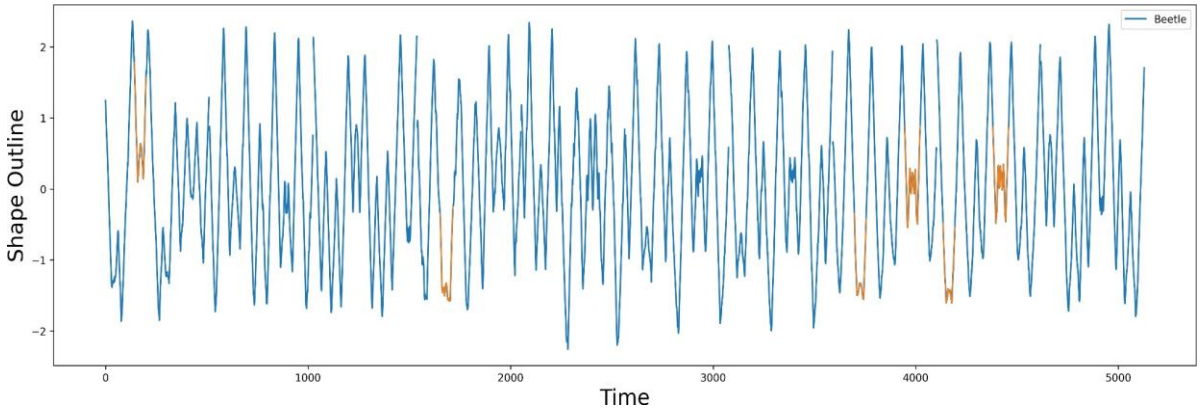
Σχήμα 6.2.1 Εμφάνιση των Matrix Profile Pbeetle,beetle και Pbeetle,fly.

Στη συνέχεια επιβεβαιώνεται αυτό που αναφέρθηκε πριν. Δηλαδή το ποια είναι τα πιθανά shapelets. Η εικόνα 6.2.2 δείχνει το γράφημα που προκύπτει από τη διαφορά των matrix profiles Pdiff. Πιο συγκεκριμένα της αφαίρεσης του Pbeetle,beetle από του Pbeetle,fly. Χάρη σε αυτό τα shapelets είναι πιο ευδιάκριτα. Συνολικά εμφανίζονται 10 πορτοκαλί κουκίδες στην οπτική αναπαράσταση. Κάθε μια από αυτές συσχετίζεται με ένα shapelet. Ο αριθμός τους μπορεί να αλλάξει. Για παράδειγμα αντί για 10, μπορεί να βρεθούν 8. Εξαρτάται από τις ανάγκες της ανάλυσης. Επιτυγχάνεται αλλάζοντας την τιμή κορυφαίων τιμών που θα βρεθούν στο Pdiff. Αφού γίνει αυτό, μετά αποθηκεύονται στη μεταβλητή idx. Ακόμη, στο σημείο ανάμεσα στο 2000 με 3000 υπάρχει ακραία διαφορά. Που σημαίνει ότι δεν είναι κατάλληλο να γίνει shapelet. Αυτό ισχύει και για τα υπόλοιπα μέρη όπου πέφτει απότομα το γράφημα. Όταν το διάγραμμα της διαφοράς κυμαίνεται σε χαμηλές τιμές δείχνει αδυναμία στη διάκριση κλάσεων. Όπως για παράδειγμα τα τμήματα κοντά στην 3000 στην εικόνα 6.2.2.



Σχήμα 6.2.2 Εμφάνιση των κορυφαίων 10 σημείων.

Για κάθε τιμή μέσα στο idx, εξάγεται ένα κομμάτι μήκους m από το beetle dataframe. Έπειτα κρατιούνται σε έναν πίνακα με shapelets. Τέλος, εμφανίζονται πάνω στο διάγραμμα. Εφόσον υπάρχουν 10 μεγαλύτερα σημεία, τόσα θα είναι και τα shapelets. Στην αναπαράσταση της 6.2.3 φαίνονται με πορτοκαλί πάνω στο γράφημα του beetle dataframe. Με αυτό τον τρόπο γίνεται πιο απλή η αναγνώριση μοτίβων με το μάτι. Για παράδειγμα τα shapelets ανάμεσα στο 4000 και στο 5000 έχουν μια ομοιότητα. Όλα παρουσιάζουν κοινά χαρακτηριστικά. Το ζήτημα είναι η αναγνώριση των καλύτερων. Αυτό κρίνεται από την ακρίβεια που αναφέρθηκε στο 6.1.

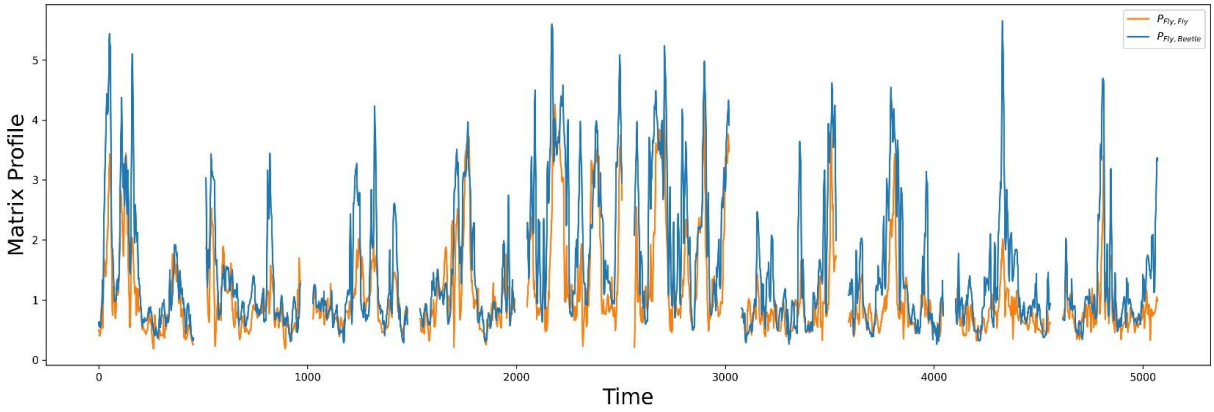


Σχήμα 6.2.3 Εμφάνιση των shapelets στο beetle_df

6.3 **Εξαγωγή shapelets από την κλάση fly**

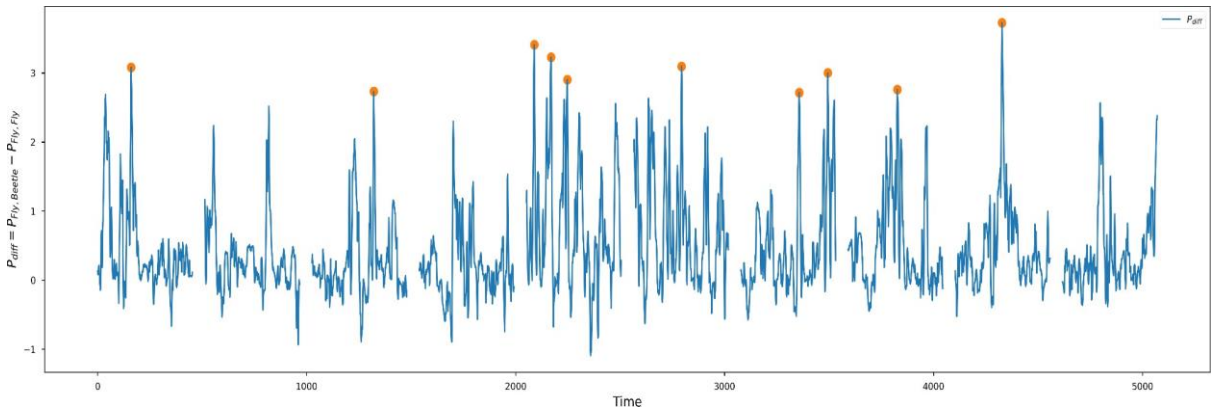
Η διαδικασία εύρεσης shapelets παραμένει ίδια με εκείνη για την κλάση beetle. Συνεχίζει να υλοποιείται για m = 59. Βέβαια υπάρχουν κάποιες διαφορές. Για παράδειγμα στα σημεία όπου χρησιμοποιούνταν το beetle dataframe μπαίνει το fly dataframe. Όπως και το ανάποδο. Η λογική ωστόσο που ακολουθείται δεν αλλάζει. Γίνεται ο υπολογισμός των δύο matrix profile Pfly,fly και Pfly,beetle. Μετά τοποθετείται το ένα πάνω στο άλλο. Ο λόγος είναι για την ανάδειξη των μεταξύ τους αποστάσεων. Στην οπτική αναπαράσταση της 6.3.1 φαίνονται δύο γραφήματα. Το μπλε αναφέρεται στο Pfly,beetle, ενώ το πορτοκαλί στο Pfly,fly. Κοιτώντας αυτά προκύπτουν ενδιαφέροντα συμπεράσματα. Για παράδειγμα στο 4327 διακρίνεται μια μεγάλη κορυφή του Pfly,beetle. Μάλιστα

έχει φτάσει το 5 στον άξονα y'y. Αυτό δείχνει ότι έχει καλή ικανότητα διαφοροποίησης των κλάσεων fly και beetle. Από την άλλη, το Pfly,fly έχει πέσει αρκετά κάτω. Στον άξονα y'y βρίσκεται στο 2 περίπου. Με αυτό φαίνεται ότι το shapelet εμφανίζεται συχνά στα δείγματα της κλάσης fly. Συνεπώς καθίσταται ως ένα καλό υποψήφιο shapelet. Το σημείο στη θέση 1321 παρουσιάζει όμοια συμπεριφορά. Δηλαδή σχετικά μεγάλο ύψος Pfly,beetle και χαμηλό Pfly,fly. Παρομοίως το ίδιο ισχύει και για τις θέσεις 3357, 3826, 2247, 3494, 2794, 2169, και 2089. Αυτά θεωρούνται ως τα κορυφαία υποψήφια shapelets.



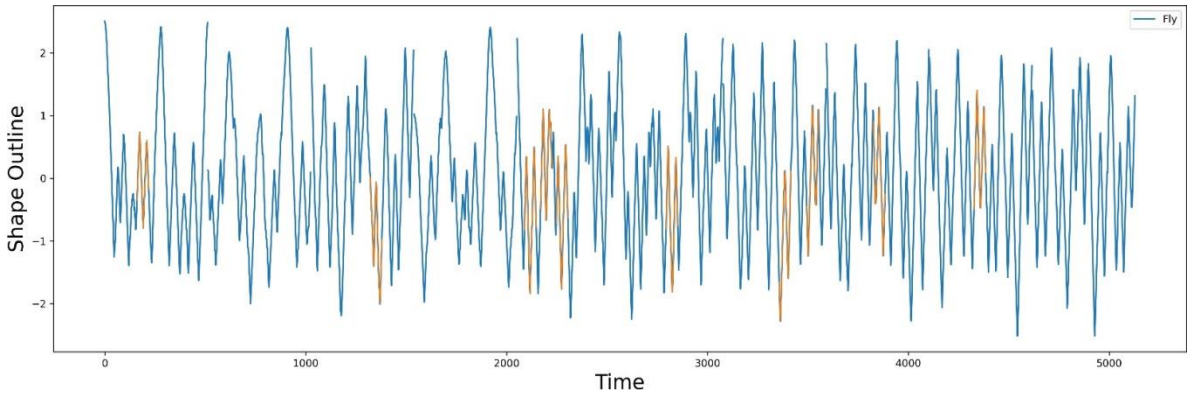
Σχήμα 6.3.1 Εμφάνιση matrix profile Pfly,fly και Pfly,beetle

Οι θέσεις που αναφέρθηκαν πριν χρησιμοποιούνται για τη δημιουργία γραφήματος στην 6.3.2. Σε αυτό συμμετέχει και το Pdiff. Μέσα του κρατάει τη διαφορά του Pfly,fly από το Pfly,beetle. Με αυτόν τον τρόπο παρουσίασης γίνεται πιο κατανοητό που βρίσκονται τα shapelets. Μάλιστα σε σύγκριση με εκείνα του beetle_df, αυτά είναι πιο διακριτά. Απλώνονται παραπάνω στο γράφημα. Που σημαίνει ότι ξεχωρίζεται που είναι το καθένα. Επίσης δείχνει ότι οι αποστάσεις ύψους των δύο matrix profile είναι πιο έντονες.



Σχήμα 6.3.2 Εμφάνιση κορυφαίων shapelets για την κλάση fly

Στο τέλος γίνεται η οπτική αναπαράσταση των shapelets πάνω στο fly_df. Για την επίτευξη αυτού υλοποιήθηκαν οι θέσεις των κορυφαίων υποψήφιων shapelets. Όπως φαίνεται και στην 6.3.3 δεν είναι ίδια μεταξύ τους σχηματικά. Παρουσιάζουν ομοιότητες, ενώ παράλληλα διατηρούν κάποιες διαφορές. Είναι δύσκολος ο εντοπισμός του καλύτερου διαισθητικά. Γι' αυτό υπάρχει η μέτρηση της ακρίβειας του κάθε shapelet.



Σχήμα 6.3.3 Αναπαράσταση των shapelets στο fly dataframe

Με βάση την εικόνα 6.3.4, τα shapelets εμφανίζονται αρκετά στα δείγματα της κλάσης fly. Επίσης είναι ικανά να διαφοροποιήσουν την fly από την beetle. Μάλιστα σε σύγκριση με τα shapelets της beetle είναι πιο ισχυρά. Αυτό φαίνεται από τις αυξημένες τιμές ακρίβειας. Τρεις από αυτές είναι 0.8, ενώ μια φτάνει στο 0.85. Η αμέσως επόμενη είναι 0.75 και άλλες τρεις 0.7. Άρα το καλύτερο shapelet θεωρείται το ένατο με 0.85. Αμέσως μετά ακολουθεί το τέταρτο, το όγδοο και το δέκατο με 0.8. Επόμενο στην κατάταξη είναι το δεύτερο, πρώτο, έκτο και το έβδομο.

```
Accuracy for shapelet 0 = 0.7
Accuracy for shapelet 1 = 0.75
Accuracy for shapelet 2 = 0.65
Accuracy for shapelet 3 = 0.8
Accuracy for shapelet 4 = 0.55
Accuracy for shapelet 5 = 0.7
Accuracy for shapelet 6 = 0.7
Accuracy for shapelet 7 = 0.8
Accuracy for shapelet 8 = 0.85
Accuracy for shapelet 9 = 0.8
```

Σχήμα 6.3.4 Μέτρηση ακρίβειας των shapelets στην κλάση fly

6.4 **Συνολική αξιολόγηση και παρατηρήσεις**

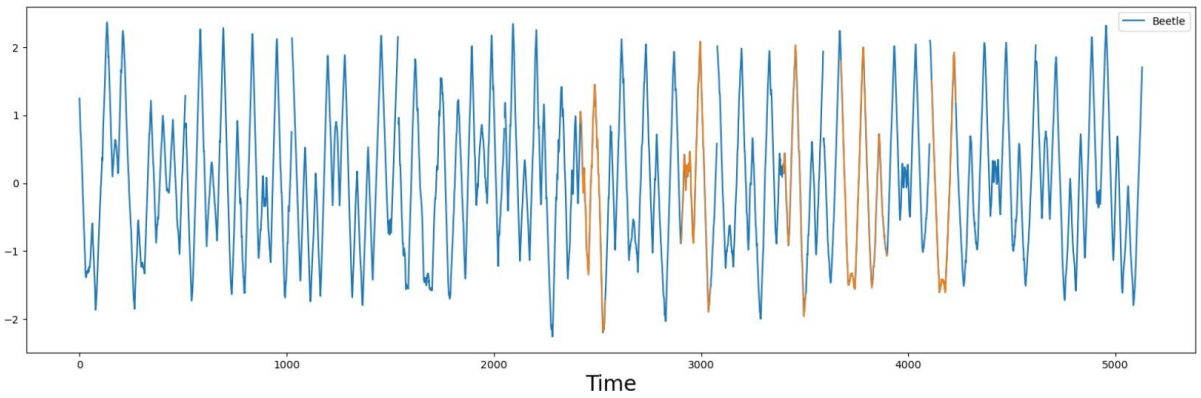
Με βάση την εικόνα 6.2.3 μπορούν να εξαχθούν κάποια πορίσματα. Στα shapelets που προκύπτουν παρατηρούνται δύο κύρια σχήματα. Καθένα από αυτά εμφανίζεται αρκετά μέσα στο beetle_df. Το πρώτο υπάρχει στο shapelet με τιμή 4000 στον x'x. Το συγκεκριμένο αποτελεί μοτίβο. Παρόμοιο του αποτελεί εκείνο στο 4500 και το άλλο κοντά στο 0. Όσο για άλλο, παρατηρείται στο shapelet περίπου στο 4100. Η μορφή του παρουσιάζει σαν μικρό δόντι ή λόφο. Έχει κοινά χαρακτηριστικά με τα δύο που η κορυφή τους προσεγγίζει το -1 με -2 στον y'y.

Βασικό ρόλο στην εύρεση των shapelets παίζει η τιμή m που θα δοθεί. Από αυτήν εξαρτώνται πολλά. Ένα από τα κυριότερα είναι το μήκος τους, που είναι ίδιο με το m. Η επιλογή αυτής στηρίζεται στον πειραματισμό. Βέβαια ανάλογα και με το σύνολο δεδομένων, η κατάλληλη τιμή διαφέρει. Για

παράδειγμα σε άλλες περιπτώσεις μπορεί να είναι 15, 100, 200 κτλ. Μια καλή ιδέα γι' αυτή δίνεται από την μελέτη της συμπεριφοράς των δειγμάτων. Επίσης, το m σχετίζεται άμεσα και με το συνολικό μήκος του dataframe. Όταν το m προσεγγίζει το μήκος προκύπτει η επικάλυψη shapelets. Δηλαδή το να είναι το ένα πάνω στο άλλο.

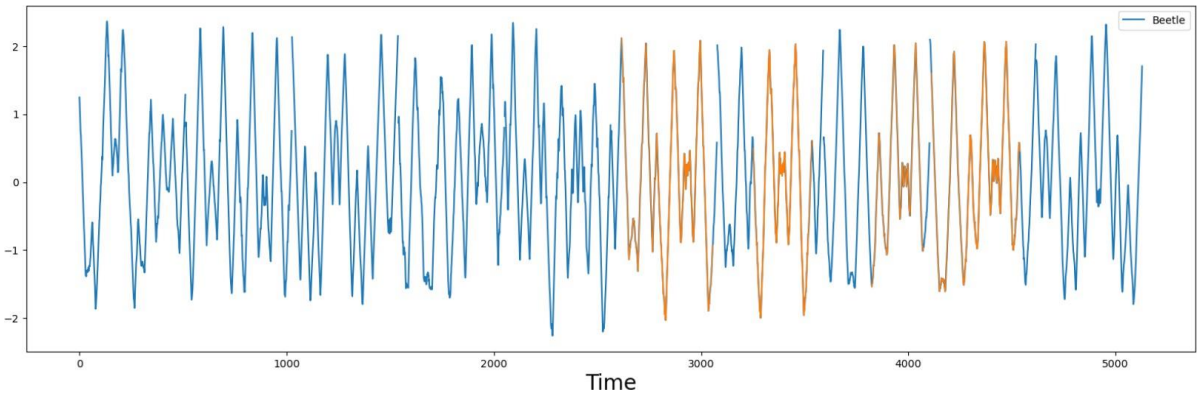
Στη συνέχεια θα δοθούν κάποιες τιμές για το m για την εξαγωγή shapelets. Στόχος είναι η κατανόηση της ευαισθησίας του.

Όταν m = 110. Όπως φαίνεται στο γράφημα 6.4.1 τα shapelets βγαίνουν μεγάλα. Ακόμη, το συνολικό μήκος του beetle dataframe είναι 512. Στο κομμάτι ανάμεσα από το 3700-4000 στον x'x παρατηρείται ταύτιση των shapelets. Ουσιαστικά δεν ξεχωρίζει κανένα καθώς φαίνονται σαν ένα ενιαίο κομμάτι. Το παραπάνω δυσκολεύει την ανάλυση τους



Σχήμα 6.4.1 Αναπαράσταση shapelets για m = 110

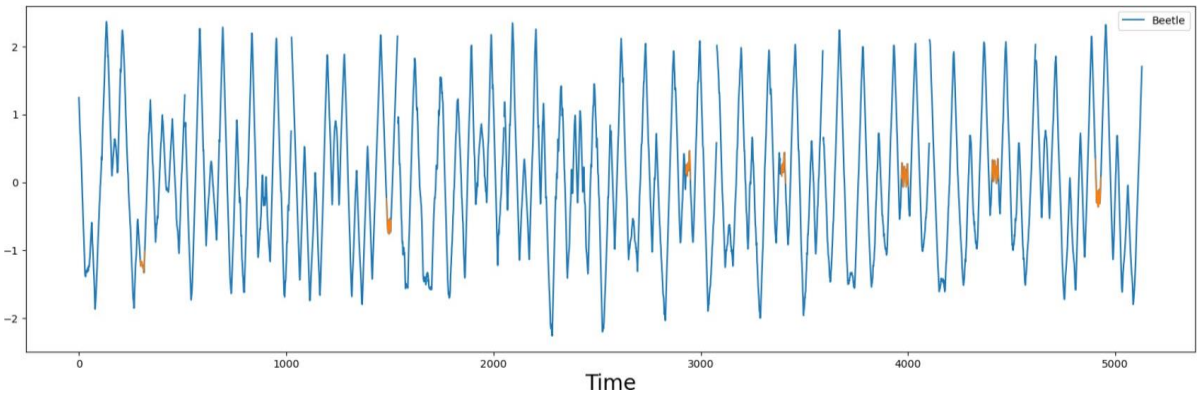
Όταν m = 250. Όσο αυξάνεται το m, τόσο χειροτερεύει το πρόβλημα. Είναι εμφανές από τα shapelets στην 6.4.2. Μάλιστα εκεί που στο γράφημα 6.4.1 βρέθηκαν 5, πλέον διακρίνονται 4. Φαίνεται ότι κάποια συγχωνεύτηκαν. Γι' αυτό μεγαλώνοντας το μήκος των shapelets χάνονται σημαντικά μοτίβα. Συνεπώς και χρήσιμη πληροφορία που θα είχε βρεθεί.



Σχήμα 6.4.2 Αναπαράσταση shapelets για m = 250

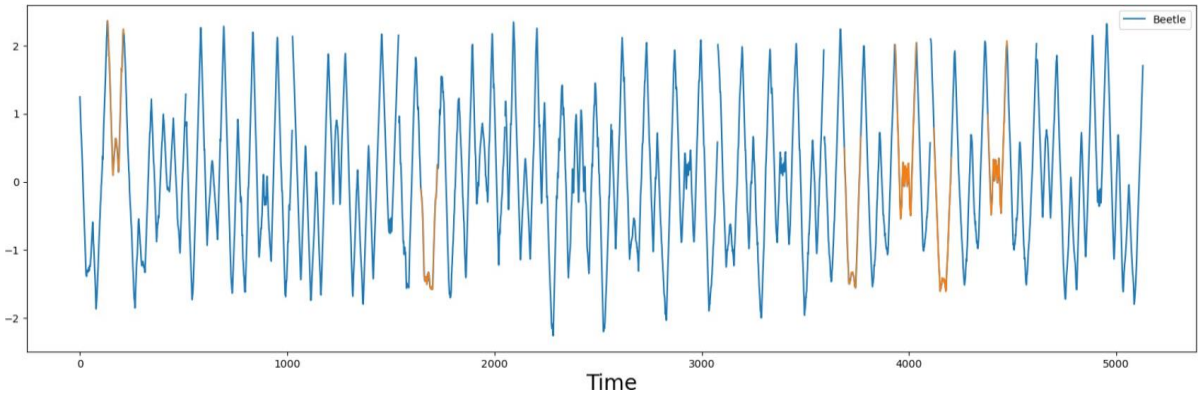
Όταν m = 23. Αλλάζοντας την τιμή σε μικρό αριθμό γίνονται πιο αντιληπτά τα shapelets. Μάλιστα προέκυψαν περισσότερα σε σύγκριση με πριν. Λογικό καθώς το χαμηλό m σημαίνει ευκολία στην

εύρεση μοτίβων. Ωστόσο σε τέτοιες τιμές υπάρχει κίνδυνος ότι τα shapelets είναι θόρυβος. Γι' αυτό είναι σημαντική η προεπεξεργασία του συνόλου δεδομένων. Στη συγκεκριμένη περίπτωση δεν χρειάστηκε κάτι τέτοιο.



Σχήμα 6.4.3 Αναπαράσταση shapelets για m = 23

Όταν m = 80. Σε αυτή την περίπτωση είναι εύκολα παρατηρήσιμα τα shapelets. Αν και μεγάλωσε το μήκος τους, δεν υπάρχει επικάλυψη. Επίσης το σχήμα τους είναι εντελώς διαφορετικό από των προηγούμενων. Αυτό προκύπτει αν συγκριθούν τα shapelets στην 6.4.4 με εκείνα των άλλων εικόνων. Γι' αυτό η επιλογή του m παίζει τόσο σημαντικό ρόλο. Τροποποιώντας την λίγο μπορεί να φέρει αλλιώτικα σε μορφή shapelets. Άρα και άλλα μοτίβα. Δηλαδή, το τι κρίνει ως όμοιο αλλάζει.



Σχήμα 6.4.4 Αναπαράσταση shapelets για m = 80

Μέσω των προηγούμενων παραδειγμάτων έγινε αντιληπτή η αξία της τιμής m. Για τα μικρά m, οι υπακολουθίες που βγαίνουν είναι απλωμένες. Η κάθε μια έχει το δικό της χώρο, χωρίς να συγχέονται μεταξύ τους. Όταν είναι υπερβολικά χαμηλή η τιμή, τα shapelets μπορεί να είναι θόρυβος. Ενώ όταν το m είναι μεγάλο, υπάρχει κίνδυνος της αλληλεπικάλυψης. Που σημαίνει ότι το ένα κρύβει το άλλο. Με αποτέλεσμα κάποια μοτίβα να φαίνονται σαν ένα ενιαία. Με αυτό τον τρόπο χάνεται πολύτιμη πληροφορία. Αυτό καθιστά τη διαδικασία αναγνώρισης shapelets λίγο δύσκολη.

Η επιλογή m = 59 για την εξαγωγή τους έγινε με βάση την συνολική απόδοση. Με αυτή στο fly dataframe η μεγαλύτερη τιμή ακρίβειας είναι το 0.85. Αμέσως μετά είναι η 0.8 που την είχαν δύο. Τρεις έφτασαν στο 0.7 και δυο στο 0.75. Αυτό δείχνει ότι τα μοτίβα που βρέθηκαν στη κλάση fly τη

διακρίνουν πολύ καλά από την beetle. Από την άλλη στο beetle dataframe η υψηλότερη ανέβηκε στο 0.8. Μετά ακολουθεί η 0.75. Έπειτα πέντε shapelets έχουν 0.7. Συγκριτικά με της fly, οι τιμές εδώ είναι πιο χαμηλές. Άρα τα shapelets της κλάσης fly είναι πιο ισχυρά από εκείνα της beetle.

6.5 Επίλογος

Μετά από διάφορους πειραματισμούς στη διαδικασία εύρεσης του καταλληλότερου m, διαπιστώθηκε ότι η τιμή αυτή είναι η 59. Αυτό φαίνεται από τις τιμές ακρίβειας που παρουσιάζουν τα shapelets για κάθε ένα διαφορετικό m. Βλεποντας την οπτικοποίηση των γραφημάτων από τα matrix profiles του Pbeetle,beetle και του Pbeetle,fly εξάγονται αξιόλογα συμπεράσματα. Τα κομμάτια που είναι τα καλύτερα υποψήφια shapelets είναι εκείνα οπού το Pbeetle,beetle είναι χαμηλό και το Pbeetle,fly υψηλό. Αυτό δείχνει ότι εμφανίζονται αρκετά συχνά στη κλάση beetle, ενώ παράλλ ηλα είναι διαφορετικά από της fly. Απομωνόνοντας τα δέκα μεγαλύτερα σημεία από όλα αυτά, βρίσκονται τα δέκα shapelets. Στη συνέχεια, για ευκολότερη κατανόηση οπτικοποιούνται και τα δέκα πάνω στο αρχικό γράφημα της κλάσης beetle. Μετά αξιολογείται η ακρίβεια του καθενός στην κατηγοριοποίηση των δειγμάτων από το σύνολο ελέγχου. Συνήθως αυτά με τις μεγαλύτερεςτιμές έχουν αυξημένη ικανότητα διάκρισης ανάμεσα στις κλάσεις beetle και fly.

Κεφάλαιο 7ο: Συμπεράσματα και προτάσεις βελτίωσης

7.1 Συμπεράσματα μελέτης

Σκοπός της μελέτης είναι η εξαγωγή shapelets στο σύνολο δεδομένων BeetleFly. Αυτό είναι σημαντικό καθώς βρίσκονται κάποια ενδιαφέροντα συμπεράσματα για τα δεδομένα. Μάλιστα, χάρη σε αυτά γίνονται ευδιάκριτα κάποια μοτίβα στη κλάση beetle. Επειδή το shapelet είναι κάτι σαν υπογραφή της κλάσης, διαθέτει ένα χαρακτηριστικό σχήμα που το κάνει να ξεχωρίζει από εκείνα της άλλης. Συνεπώς, αν μέσα στον χρόνο έρθει ένα καινούργιο αταξινόμητο δείγμα από δεδομένα εικόνας, κοιτάζοντας το σχήμα μπορεί να ανατεθεί σε μια από τις δύο κλάσεις. Επίσης, πρέπει να συνυπολογίζεται και η ακρίβεια που παρέχει στην κατηγοριοποίηση. Χάρη σε αυτή είναι πιο κατανοητή η επιλογή των καλύτερων shapelet. Δεν έχουν όλα την ίδια τιμή ακρίβειας. Αυτά που είναι θεμιτά πετυχαίνουν ένα ποσοστό πολύ υψηλό. Με βάση αυτές τις τιμές κινείται ο βαθμός διακριτότητας μεταξύ των κλάσεων beetle και fly. Αποτελεί έναν τρόπο ελέγχου της δύναμης του κάθε shapelet. Σε αυτό το κομμάτι είναι σημαντική η εύρεση του κατάλληλου αριθμού για το m του υπολογισμού matrix profile. Δεν υπήρχε κάποιο κείμενο που να έλεγε ποια είναι η καλύτερη τιμή γι' αυτό. Προήλθε μετά από πειραματισμούς με διάφορες ενναλακτικές και παρατηρώντας τα ποσά ακρίβειας. Το 59 επιλέχθηκε γιατί πέρα από μια καλή τιμή την 0.8, πετύχαινε σχετικά μεγάλα ποσά στις υπόλοιπες. Επίσης επειδή σε ένα σύνολο δεδομένων με μήκος 512 και 20 δεδομένα εκπαίδευσης ένα πολύ μικρό m θα έφερνε shapelets που μπορεί να ήταν θόρυβος. Δηλαδή μπορεί να μην έφερνε κάποια πληροφορία. Ακόμη, αν ήταν πολύ μεγάλο,τότε θα επικάλυπτε το ένα το άλλο. Η μέθοδος εύρεσης αποτελεσμάτων ήταν αρκετά αποδοτική στην εξαγωγή των shapelets. Βρέθηκαν επιτυχώς όλα και με γρήγορο ρυθμό Για την υλοποίηση έγινε χρήση της γλώσσας προγραμματισμού Python πάνω στο περιβάλλον Jupyter. Αποτελεί ένα πολύ καλό και εύκολο στην εφαρμογή εργαλείο. Είναι ένα μεγάλο υπέρ ότι ο χρήστης επιλέγει το κομμάτι κώδικα που θέλει να τρέξει και αμέσως να εμφανίζονται τα αποτελέσματα από κάτω. Αυτό καθιστά πιο απλές τις περισσότερες διαδικασίες που έτρεξαν, όπως η οπτικοποίηση των γραφημάτων. Στις περιπτώσεις όπου χρειάστηκε μικρή αλλαγή στο μέγεθος του τίτλού ή οπουδήποτε κατευθείαν βγήκαν τα αποτελέσματα, χωρίς να χρειαστεί να τρέξει από την αρχή όλος ο κώδικας. Χάρη στις βιβλιοθήκες Python που υλοποιήθηκαν, βγήκε επιτυχής όλη αυτή η μελέτη. Η STUMPY που υπολογίζει το matrix profile, ήταν πολύ αποδοτική σε χρόνο εκτέλεσης. Επίσης ήταν αρκετά εύχρηστη και κατανοητή.

7.2 Προτάσεις βελτίωσης

Μια πιθανή βελτίωση του πειραματικού στάδιου θα ήταν η χρήση παραπάνω από μια μεθόδων εύρεσης shapelets. Έτσι θα ήταν δυνατή η σύγκριση απόδοσης τους. Μια άλλη θα μπορούσε να γίνει στο κομμάτι του μεγέθους του συνόλου δεδομένων. Που σημαίνει με περίσσοτερες κατηγορίες και με θόρυβο. Με αυτό τον τρόπο θα ήταν εφικτός ο έλεγχος της ανθεκτικότητας της μεθόδου.Σε αυτή την περίπτωση, το BeetleFly ήταν ένα ήδη προετοιμασμένο σύνολο για κατηγοριοποίηση. Ήταν κανονικοποιημένο και δεν υπήρχε θόρυβος. Επομένως θα είχε παραπάνω ενδιαφέρον να υλοποιηθούν όχι και τόσο τέλεια σύνολα. Γιατί θα μπορούσε να δοκιμαστεί και η λειτουργικότητα της μεθόδου. Επιπλέον, αν το μέγεθος του συνόλου ήταν πολύ μεγαλύτερο και αυτού της εκπαίδευσης και του ελέγχου θα διακρίνονταν καλύτερα η επίδοση του. Για παράδειγμα, μια μέθοδος που λειτουργεί σε ένα σχετικά μικρό σύνολο δεδομένων δεν λέει κάτι σαν πληροφορία. Από την άλλη, αν καταφέρει να

τρέξει σε ένα τεράστιο, εκεί θα φανεί στην πραγματικότητα το πόσο καλά δουλεύει. Οπότε, αν και η επίδοση του στην πτυχιακή εργασία ήταν πολύ καλή, δεν μπορεί να συγκριθεί με κάτι. Ακόμη, θα είχε ιδιαίτερη αξία αν η εύρεση των shapelet υιοθετούσε στοιχεία από την μηχανική μάθηση. Όπως για παράδειγμα την εφαρμογή νευρωνικών δικτύων για την κατηγοριοποίηση. Τέλος, θα ήταν σημαντικό να βρεθούν και άλλοι τομείς της καθημερινότητας τιες οποιες βοηθάνε οι χρονοσειρές και η ανάλυση τους.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] J. Breckling, The Analysis of Directional Time Series: Applications to Wind Speed and Direction. New York: Springer, 1989.

[2] X. Zhang, T. Zhang, A. A. Young, and X. Li, "Applications and comparisons of four time series models in epidemiological surveillance data," *PLoS ONE*, vol. 9, no. 2, Art. no. e88075, Feb. 2014. [Online]. Available: https://doi.org/10.1371/journal.pone.0088075

[3] G. Kirchgässner, J. Wolters, and U. Hassler, *Introduction to Modern Time Series Analysis*. Berlin: Springer, 2012.

[4] T. W. Anderson, *The Statistical Analysis of Time Series*. New York: Wiley, 1994.

[5] P. J. Brockwell and R. A. Davis, *Time Series: Theory and Methods*, 2nd ed. New York: Springer, 2009.

[6] B. Chiu, E. Keogh, and S. Lonardi, "Probabilistic discovery of time series motifs," in Proc. 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '03), New York, USA, 2003, pp. 493–498. [Online]. Available: https://doi.org/10.1145/956750.956808

[7] A. Almeida, S. Bras, S. Sargento, and F. C. Pinto, "Time series big data: a survey on data stream frameworks, analysis and algorithms," Journal of Big Data, vol. 10, no. 83, pp. 1-37,2023. doi: 10. 1186/s40537-023-00760-1.

[8] C.-C. M. Yeh, Y. Zhu, L. Ulanova, N. Begum, Y. Ding, H. A. Dau, D. F. Silva, A. Mueen, and E. Keogh, "Matrix Profile I: All Pairs Similarity Joins for Time Series: A Unifying View That Includes Motifs, Discords and Shapelets," in Proc. 2016 IEEE 16th International Conference on Data Mining (ICDM), Barcelona, Spain, 2016, pp. 1317–1322, doi: 10.1109/ICDM.2016.0179.

[9] P. Esling and C. Agon, "Time-series data mining," *ACM Comput. Surveys*, vol. 45, no. 1, Art. 12, Nov. 2012, ,pp. 34. [Online]. Available: https://doi.org/10.1145/2379776.2379788

[10] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd ed. San Francisco, CA: Morgan Kaufmann, 2005.

[11] D. D. Pegarkov, *National Security Issues*. New York: Nova Publishers, 2006.

[12] J. Han, M. Kamber and J. Pei, *Data Mining:Concepts and Techniques*. 3rd ed. Waltham, MA: Morgan Kaufmann, 2011.

[13] R. Mercer, S. Alaee, A. Abdoli, S. Singh, A. Murillo, and E. Keogh, "Matrix Profile XXIII: Contrast Profile: A Novel Time Series Primitive that Allows Real World Classification," in *Proc. 2021 IEEE International Conference on Data Mining (ICDM)*, Auckland, New Zealand, 2021, pp. 1240–1245, doi: 10.1109/ICDM51629.2021.00151

[14] C.-C. M. Yeh, N. Kavantzas, and E. Keogh, "Matrix Profile VI: Meaningful Multidimensional Motif Discovery," in *Proc. 2017 IEEE International Conference on Data Mining (ICDM)*, New Orleans, LA, USA, 2017, pp. 565–574, doi: 10.1109/ICDM.2017.66.

[15] R. Guiotti and M. D'Onofrio, "Matrix Profile-Based Interpretable Time Series Classifier," *Frontiers in Artificial Intelligence*, vol. 4, 2021. doi: 10.3389/frai.2021.699448. [Online]. Available: https://www.frontiersin.org/journals/artificial-intelligence/articles/10.3389/frai.2021.699448

[16] L. Ye and E. Keogh, "Time series shapelets: a new primitive for data mining," in *Proc. of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '09)*, New York, NY, USA, 2009, pp. 947–956. doi: 10.1145/1557019.1557122.

[17] D. T. Anh, "An overview of similarity search in time series data," VNUHCM Journal of Science and Technology Development, vol. 14, no. 2, pp. 71–79, 2011. doi: 10.32508/stdj.v14i2.1911.

[18] D. Rafiei and A. Mendelzon, “Similarity-based queries for time series data,” In Proc. 1997 ACM SIGMOD International Conference on Management of Data (SIGMOD '97), New York, NY, USA, 1997, pp. 13–25, doi: 10.1145/253260.253264
[19] M. H Richardson, “Fundamentals of the Discrete Fourier Transform”, *Sound & Vibration Magazine*, Mar. 1978, Hwelett Packard Corporation, Santa Clara, CA. [Online]. Available: http://papers.vibetech.com/Paper09.pdf
[20] R. Giusti and G. E. A. P. A. Batista, “An Empirical Comparison of Dissimilarity Measures for Time Series Classification,” in Proc. 2013 Brazilian Conference on Intelligent Systems (BRACIS), Fortaleza, Brazil, 2013, pp. 82–88, doi: 10.1109/BRACIS.2013.22.
[21] B. Lkhagva, Y. Suzuki, and K. Kawagoe, “Extended SAX: extension of symbolic aggregate approximation for financial time series data representation,” in Proc. The 17th Data Engineering Workshop, IEICE, 2006
[22] P.Bloomfield, *Fourier Analysis of Time Series: An Introduction*, 2nd ed. Hoboken, NJ: John Wiley & Sons, 2004
[23] P. Chaovalit, A. Gangopadhyay, G. Karabatis, and Z. Chen, “Discrete wavelet transform-based time series analysis and mining,” ACM Comput. Surv., vol. 43, no. 2, Art. 6, Jan. 2011, 37 pages, doi: 10.1145/1883612.1883613
[24] A. I. Mees, P. E. Rapp, and L. S. Jennings, “Singular-value decomposition and embedding dimension,” *Phys. Rev. A*, vol. 36, no. 1, pp. 340–346, Jul. 1987. [Online]. Available: https://journals.aps.org/pra/pdf/10.1103/PhysRevA.36.340
[25] W. McKinney, *Python for Data Analysis: Data Wrangling with pandas, NumPy, and Jupyter*, 3rd ed. Sebastopol, CA: O'Reilly Media, Inc., 2022.
[26] S. M. Law, “STUMPY: A Powerful and Scalable Python Library for Time Series Data Mining,” *Journal of Open Source Software*, vol. 4, no. 39, p. 1504, 2019. [Online]. Available: https://doi.org/10.21105/joss.01504
[27] B. E. Granger and F. Perez, “Jupyter: Thinking and Storytelling With Code and Data,” Computing in Science & Engineering, vol. 23, no. 2, pp. 7-14, Mar.-Apr. 2021, doi: 10.1109/MCSE.2021.3059263
[28] K. Grotov, S. Titov, V. Sotnikov, Y. Golubev, and T. Bryksin, “A large-scale comparison of Python code in Jupyter notebooks and scripts,” In Proc. 19th International Conference on Mining Software Repositories (MSR ’22), New York, NY, USA, 2022, pp. 353–364, doi: 10.1145/3524842.3528447.
[29] C.-C. M. Yeh et al., “Matrix Profile I: All Pairs Similarity Joins for Time Series: A Unifying View That Includes Motifs, Discords and Shapelets,” in Proc. 2016 IEEE 16th International Conference on Data Mining (ICDM), Barcelona, Spain, 2016, pp. 1317–1322, doi: 10.1109/ICDM.2016.0179.
[30] Y. Zhu et al., “Matrix Profile II: Exploiting a Novel Algorithm and GPUs to Break the One Hundred Million Barrier for Time Series Motifs and Joins,” in Proc. 2016 IEEE 16th International Conference on Data Mining (ICDM), Barcelona, Spain, 2016, pp. 739–748, doi: 10.1109/ICDM.2016.0085.
[31] H. El Khansa, C. Gervet, and A. Brouillet, “Application of Matrix Profile Techniques to Detect Insightful Discords in Climate Data,” *International Journal on Soft Computing, Artificial Intelligence and Applications (IJSCAI)*, 2022, doi: 10.5121/ijscai.2021.11201.
[32] J. Shi, N. Yu, E. Keogh, H. K. Chen, and K. Yamashita, “Discovering and Labeling Power System Events in Synchrophasor Data with Matrix Profile,” in Proc. 2019 IEEE Sustainable Power and Energy Conference (iSPEC), Beijing, China, 2019, pp. 1827–1832, doi: 10.1109/iSPEC48194.2019.8975286.

[33] E. Keogh, J. Lin, S.-H. Lee, and H. Van Herle, “Finding the most unusual time series subsequence: algorithms and applications,” *Knowledge and Information Systems*,vol. 11,no. 1, pp. 1-27, 2006, doi: 10.1007/s10115-006-0034-6

[34] H. M. Faheem, “Accelerating motif finding problem using grid computing with enhanced Brute Force,” in Proc. The 12th International Conference on Advanced Communication Technology (ICACT), Gangwon, Korea (South), 2010, pp. 197–202

[35] I. Alkhwaja, M. Albugami, A. Alkhwaja, M. Alghamdi, H. Abahussain, F. Alfawaz, A. Almurayh, and N. Min-Allah,, “Password Cracking with Brute Force Algorithm and Dictionary Attack Using Parallel Programming,” *Appied. Sciences*, vol. 13, no. 10, p. 5979, 2023. [Online]. Available: https://doi.org/10.3390/app13105979

[36] Z. Wu, N. E. Huang, S. R. Long, and C.-K. Peng, “On the trend, detrending, and variability of nonlinear and nonstationary time series,” *Proc. Natl. Acad.Sci. U.S.A.*, vol. 104, no. 38, pp. 14889–14894, Sep. 2007.

[37] E. R. Rosca, “Stationary and Non-Stationary Time Series,” *The USV Annals of Economics and Public Administration,* vol. 10, no. 1, pp. 177–186, 2010.

[38] E. Ogasawara, L. C. Martinez, D. de Oliveira, G. Zimbrão, G. L. Pappa, and M. Mattoso, “Adaptive Normalization: A novel data normalization approach for non-stationary time series,” in Proc. The 2010 International Joint Conference on Neural Networks (IJCNN), Barcelona, Spain, 2010, pp. 1–8, doi: 10.1109/IJCNN.2010.5596746

[39] R. M. Warner, Spectral Analysis of Time-series Data. New York, NY: Guilford Press, 1998.

[40] M. K. Bhatia, “Data analysis and its importance,” International Research Journal of Advanced Engineering and Science, vol. 2, no. 1, pp. 166–168, 2017

[41] C.-C. M. Yeh, Y. Zhu, L. Ulanova, N. Begum, Y. Ding, H. A. Dau, Z. Zimmerman, D. F. Silva, A. Mueen, and E. Keogh, “Time series joins, motifs, discords and shapelets: a unifying view that exploits the matrix profile,” Data Mining and Knowledge Discovery, vol. 32, pp. 83–123, 2018, doi: 10.1007/s10618-017-0519-9.

[42] E. Kolog, “Dynamic Programming using Brute Force Algorithm for a Traveling Salesman Problem,” ResearchGate, 2015. [Online]. Available: https://doi.org/10.13140/RG.2.1.2304.2727

[43] J. Hills, J. Lines, E. Baranauskas, J. Mapp, and A. Bagnall, “Classification of time series by shapelet transformation,” Data Mining and Knowledge Discovery, vol. 28, pp. 851–881, 2014. doi: 10.1007/s10618-013-0322-1.

ΠΑΡΑΡΤΗΜΑ: ΠΕΡΙΓΡΑΦΗ ΚΩΔΙΚΑ

Π1. Εισαγωγή βιβλιοθηκών Python στο Jupyter

```
%matplotlib inline


import stumpy
import pandas as pd
import numpy as np
import  matplotlib.pyplot  as  plt
from sklearn import tree
from sklearn import metrics
from scipy.signal import find_peaks

plt.style.use('seaborn-v0_8-notebook')
```

Πριν ξεκινήσει η οποιαδήποτε διαδικασία είναι σημαντική η φόρτωση των βιβλιοθηκών που θα χρειαστούν. Αυτές είναι η STUMPY, η Pandas, NumPy, Matplotlib, sklearn και τέλος η SciPy. Για λόγους πρακτικότητας, δίνονται εναλλακτικές ονομασίες. Η Pandas αντιστοιχεί στο pd, η NumPy στο np ενώ η Matplotlib στο plt. Με αυτό τον τρόπο γίνεται πιο εύκολη η αναφορά τους. Κάθε μια από αυτές παίζει και από έναν σημαντικό ρόλο στη εκπλήρωση του στόχου. Δηλαδή, την εξαγωγή των shapelets και στη κατηγοριοποίηση. Πιο συγκεκριμένα, η STUMPY ευθύνεται για τον υπολογισμό του matrix profile. Η Pandas είναι βασική στην ανάλυση δεδομένων χρονοσειρών λόγω διαχείρισης dataframe. Όσο για την NumPy, είναι υπεύθυνη για τις μαθηματικές πράξεις. Καλώντας την pyplot, η Matplotlib έχει στην ευχέρεια της πολλά είδη γραφημάτων για οπτικές αναπαραστάσεις. Επίσης παρέχει διάφορα καλούπια με προσαρμοσμένα αισθητικά χαρακτηριστικά. Ανάλογα με ποιο επιλεγεί, αλλάζει και η εμφάνιση των αναπαραστάσεων. Άλλα μπορεί να διαθέτουν πιο σκοτεινά χρώματα, ενώ άλλα πιο έντονα. Είναι στην κρίση του κάθε χρήστη η τελική απόφαση. Αφού έγινε δοκιμή πολλών τέτοιων προτύπων, κρίθηκε ότι το καλύτερο είναι το seaborn-v0_8-notebook. Μέσω του tree της sklearn, μπορεί να χρησιμοποιηθεί ένας κατηγοριοποιητής δέντρου απόφασης. Χάρη σε αυτόν μπορούν να αξιολογηθούν τα shapelets στην ικανότητα διάκρισης ανάμεσα στις κλάσεις beetle και fly. Το metrics του sklearn παρέχει αυτά που χρειάζονται για τις μετρήσεις ακριβείας. Από την scipy έγινε χρήση του signal. Πιο συγκεκριμένα, της find_peaks για την εύρεση των κορυφαίων σημείων.

Π2 Φόρτωση του συνόλου δεδομένων train

```
train_df=pd.read_csv('C:/Users/athen/Downloads/BeetleFly/BeetleFly_T
```

```
RAINNEW.csv')
```

Με βάση μια προκαθορισμένη διαδρομή, το αρχείο τύπου .csv κρατιέται μέσα στη μεταβλητή train_df. Ωστόσο, για να γίνει αυτό πρέπει πρώτα να αλλάξει μορφή σε dataframe. Η διαδικασία αυτή πραγματοποιείται από τη βιβλιοθήκη Pandas, την pd.

Π3 Διάσπαση του train

```
beetle_df = train_df[train_df['0'] == 1.0].iloc[:,1:].reset_index(
drop=True)

beetle_df = (beetle_df.assign(NaN=np.nan)

                .stack(dropna=False).to_frame().reset_index(drop=True)

                .rename({0: "Shape Outline"}, axis='columns')

            )

fly_df = train_df[train_df['0'] ==
2.0].iloc[:,1:].reset_index(drop=       True)

fly_df = (fly_df.assign(NaN=np.nan)

            .stack(dropna=False).to_frame().reset_index(drop=True)

                .rename({0: "Shape Outline"}, axis='columns')

            )
```

Το train_df διασπάται σε δύο επιμέρους dataframe, το beetle_df και το fly_df. Ο διαχωρισμός γίνεται με βάση το περιεχόμενο της στήλης που δείχνει την ετικέτα. Σε αυτή την περίπτωση είναι η πρώτη στήλη. Αν η τιμή είναι 1.0 τότε αποθηκεύεται στο beetle_df, ενώ αν είναι 2.0 στο fly_df. Όσον αφορά το train_df[train_df['0'] == 1.0] στο beetle_df , εδώ επιλέγονται οι τιμές 1.0 από την πρώτη στήλη του train_df. Στο .iloc[:, 1:] κρατάει όλα τα δεδομένα χωρίς την ετικέτα. Έπειτα το .reset_index(drop=True) επαναφέρει το index του dataframe στην αρχική του κατάσταση. Άρα στην ουσία όλα τα δείγματα που ξεκινούν με 1.0 θα κρατηθούν στο beetle_df. Στη συνέχεια, προστίθεται σε αυτό μια νέα στήλη τύπου NaN .assign(NaN=np.nan.). Έπειτα τοποθετείται το ένα δείγμα κάτω από το άλλο με το stack(dropna =False). Συνεπώς μετά από κάθε δείγμα που εισάγεται, θα υπάρχει η τιμή NaN. Πραγματοποιείται ώστε τα δείγματα να είναι ευδιάκριτα μεταξύ τους. Για να φαίνεται δηλαδή που αρχίζει το ένα και τελειώνει το άλλο. Έπειτα η στήλη μετατρέπεται σε dataframe με το to_frame(). Μετά γίνεται επαναφορά του index με το .reset_index(drop=True). Τέλος το όνομα της στήλης από 0 γίνεται Shape Outline με το .rename({0: "Shape Outline"}, axis='columns'. Η διαδικασία είναι εντελώς ίδια και για το fly_df.

Π4 Εμφάνιση των χρονοσειρών beetle και fly

```
plt.rcParams['xtick.direction'] = 'out'
```

```
fig,axs = plt.subplots(2,sharex=True,figsize=(20, 6),gridspec_kw =
{ 'hspace': 0})

plt.suptitle('beetle vs. fly', fontsize='30')

plt.xlabel('Time', fontsize ='20')

fig.text(0.09, 0.5, 'Shape Outline', va='center',
rotation='vertical', fontsize='20')
axs[0].plot(beetle_df, color="C1" ,label="beetle", linewidth = 1.5)

axs[0].legend()

axs[0].set_ylim(-3,3)


axs[1].plot(fly_df, label="fly", linewidth = 1.5)

axs[1].legend()

axs[1].set_ylim(-3,3)

plt.show()
```

Το κομμάτι αυτό αφορά την οπτική αναπαράσταση των beetle_df και fly_df. Πολύ βασική για την υλοποίηση είναι η βιβλιοθήκη Matplotlib ή αλλιώς plt. Παρέχει ποικίλα εργαλεία που βοηθούν στην επιτέλεση αυτού.

Στην αρχή αυτά που γίνονται είναι πιο γενικά . Με το plt.rcParams['xtick.direction'] = 'out' ορίζονται μικρές γραμμές πάνω σε κάθε τιμή του άξονα χ'χ. Μάλιστα το 'out' δείχνει ότι κατευθύνονται προς τα έξω. Στη συνέχεια, φτιάχνει έναν χώρο(fig) μέσα στον οποίο υπάρχουν σε δύο μικρότερα plots. Το axs δείχνει τον αριθμό των κάθετων διαγραμμάτων που προέκυψαν. Υλοποιείται με την plt.subplots(2, sharex=True, figsize=(20, 6) ,gridspec_kw={'hspace': 0}). Το figsize =(20, 6) καθιστά ότι το μέγεθος του καθένα θα είναι 20 πλάτος και 6 ύψος. Όσο για το gridspec_kw={'hspace': 0}, καλύπτει ότι μεταξύ των plots δε θα υπάρχει κενό. Θα είναι δηλαδή ενιαίο. Ακόμη, το sharex=True σημαίνει ότι τα γραφήματα θα έχουν τον ίδιο άξονα χ'χ. Στην επόμενη γραμμή, με το plt.suptitle('beetle vs. fly', fontsize='30') γίνεται η ανάθεση γενικού τίτλου με μέγεθος 30. Αντίστοιχα δίνεται τίτλος με μέγεθος 20 στον άξονα χ'χ με το plt.xlabel('Time', fontsize ='20'). Ο τίτλος για τον άξονα y'y παρέχεται από το fig.text(0.09, 0.5, 'Shape Outline', va='center', rotation='vertical', fontsize='20'). Όπως φαίνεται εδώ δόθηκαν παραπάνω τροποποιήσεις. Πιο συγκεκριμένα, το κείμενο του τίτλου να είναι στοιχισμένο στο κέντρο, να είναι κάθετο, να έχει μέγεθος 20 με 0.09 απόσταση από τον πλαίσιο και 0.5 από τη μέση του.

Τα δύο μικρότερα plots που προέκυψαν αναφέρονται ως axs[0] το πάνω και axs[1] το κάτω. Και τα δύο ακολουθούν τα ίδια βήματα όσον αφορά την οπτικοποίηση των beetle_df και fly_df. Το πρώτο, plot(beetle_df, color="C1" ,label="beetle", linewidth = 1.5) θέτει τα χαρακτηριστικά της γραφικής αναπαράστασης του beetle_df. Του προσθέτει πορτοκαλί χρώμα C1, μέγεθος γραμμής 1.5. Τέλος, βάζει μια ετικέτα που περιγράφει το όνομα της κλάσης για λόγους πρακτικότητας. Μέσω του .legend() εμφανίζεται το παραπάνω. Για να είναι ισάξια η σύγκριση των γραφημάτων, προσθέτει έναν

περιορισμό στις τιμές του άξονα y'y. Αυτό επιτυγχάνεται με το .set_ylim(-3, 3), που θέτει ότι τα όρια θα ξεκινούν από το 3 και θα τελειώνουν στο -3.

Π5 Υπολογισμός και Οπτικοποίηση Matrix Profile

```
m = 59

P_beetle_beetle = stumpy.stump(beetle_df["Shape Outline"],m)[:,0
].astype(float)

P_beetle_fly = stumpy.stump(beetle_df["Shape Outline"],m,fly_df["
Shape Outline"],ignore_trivial=False)[:, 0].astype(float)
```

Το επόμενο στάδιο στην εύρεση shapelet είναι ο υπολογισμός του matrix profile. Δεν είναι υλοποιήσιμο χωρίς την επιλογή τιμής m. Αυτή είναι που ορίζει το μέγεθος των shapelets. Μετά από διάφορες δοκιμές αποφασίστηκε ότι η καταλληλότερη είναι η 59. Σε αυτό το σημείο σημαντική θέση έχει η βιβλιοθήκη STUMPY. Χάρη σε αυτή πραγματοποιείται η εύρεση των matrix profile P_beetle_beetle και P_beetle_fly. Από το πρώτο θα φανεί η ομοιότητα ανάμεσα στην κλάση beetle και στον εαυτό της. Ένα shapelet είναι αντιπροσωπευτικό της κλάσης του. Που σημαίνει ότι πρέπει να εμφανίζεται συχνά μέσα σε αυτή. Ενώ το δεύτερο δείχνει την ομοιότητα μεταξύ της beetle και της fly. Για την εξαγωγή shapelets στόχος είναι το P_beetle_beetle να είναι όσο μικρότερο γίνεται. Αυτό θα δείξει ότι εμφανίζονται συχνά στην κλάση. Από την άλλη το P_beetle_fly πρέπει να είναι ψηλά. Γιατί αυτό σημαίνει ότι τα shapelets της beetle δεν αντιστοιχούν με τμήματα του fly. Γι' αυτό το λόγο γίνεται υπολογισμός των δύο αυτών matrix profiles. Η γνώση αυτή θα βοηθήσει στην συνέχεια.

Για το P_beetle_beetle, η stump() παίρνει δύο παραμέτρους. Το beetle_df dataframe με την στήλη Shape Outline και το m. Το [:, 0] σημαίνει ότι θα κρατήσει την χαμηλότερη απόσταση. Με το .astype(float) μετατρέπεται σε τύπου float. Δηλαδή για κάθε υποσειρά μεγέθους 59, θα βρίσκει την πιο όμοια στην ίδια την κλάση.

Με ανάλογο τρόπο πραγματοποιείται και η εύρεση του P_beetle_fly. Όμως αντί για μια παράμετρο όπως με το P_beetle_beetle, εδώ η stump() θα πάρει τρεις. Αρχικά, το beetle_df με τη στήλη Shape Outline. Έπειτα την τιμή m και τέλος το fly_df dataframe. Η διαφορά σε σχέση με πριν είναι ότι εδώ προστίθεται αυτό το ignore_trivial=False. Αυτό δίνει μια οδηγία στο stump() ότι δεν πρέπει να αγνοεί κανέναν αποτέλεσμα. Όπως και πριν, το [:, 0] δείχνει ότι μένει η πιο μικρή απόσταση. Αντίστοιχα και με το .astype(float) το κάνει να αλλάξει σε float.

```
P_beetle_beetle[P_beetle_beetle==np.inf]=np.nan
P_beetle_fly[P_beetle_fly==np.inf]=np.nan
```

Τόσο στο beetle_df όσο και στο fly_df υπήρχαν NaN τιμές. Συνεπώς, με τον υπολογισμό των matrix profile κάποιες από αυτές έγιναν τύπου inf. Γι' αυτό θέλοντας να διατηρηθεί η αρχική δομή, μετατρέπονται σε NaN.

Το [P_beetle_beetle == np.inf] = np.nan, σημαίνει ότι όσες τιμές από το P_beetle_beetle είναι inf να

γίνουν NaN μέσω της NumPy. Έπειτα το αποτέλεσμα κρατιέται στο P_beetle_beetle.

Αντίστοιχα το [P_beetle_fly == np.inf] = np.nan, δείχνει ότι για κάθε inf που βρίσκει μέσα στο P_beetle_fly θα μετατρέπεται σε NaN. Αυτό που προκύπτει μένει στο P_beetle_fly.

```
plt.rcParams['xtick.direction'] = 'out'
plt.figure(figsize=(20, 6))
plt.plot(P_beetle_beetle,color="C1",label="$P_{Beetle,Beetle}$",
linewidth = 1.5)
plt.plot(P_beetle_fly,label="$P_{Beetle,Fly}$",linewidth=1.5)
plt.xlabel("Time", fontsize="20")
plt.ylabel("Matrix Profile", fontsize="20")
plt.legend()
plt.show()
```

Προκειμένου να γίνει αντιληπτή η διαφορά στο ύψος μεταξύ των δύο matrix profiles τοποθετούνται το ένα πάνω από το άλλο. Στα μέρη όπου υπάρχουν μεγάλες αποστάσεις δείχνει πιθανά shapelets. Αρχικά με το['xtick.direction'] = 'out' ορίζεται ως έξω η κατεύθυνση που κοιτάνε οι γραμμές των ποσών στον άξονα χ'χ. Με το figsize=(20, 6) τίθεται ότι το μέγεθος του πλαισίου θα έχει 20 πλάτος και 6 ύψος. Για την οπτικοποίηση του P_beetle_beetle, με την plot(), του δίνεται χρώμα πορτοκαλί color="C1" και πάχος γραμμής 1.5, linewidth = 1.5. Ακόμη με το label="$P_{Beetle,Beetle}$", προστίθεται μια ετικέτα πάνω δεξιά στο γράφημα με κείμενο P_Beetle,Beetle.

Ακριβώς η ίδια διαδικασία εφαρμόζεται και για την οπτικοποίηση του P_beetle_fly. Δηλαδή ετικέτα πάνω δεξία αλλά με το P_Beetle,Fly και πάχος γραμμής 1.5. Μόνο που δεν ορίζεται κάποιο χρώμα. Γι' αυτό παίρνει το προκαθορισμένο μπλε.

Με την xlabel() προστίθεται τίτλος για τον άξονα χ'χ, Time με μέγεθος 20, fontsize="20".
Με την ylabel() δίνεται τίτλος για τον άξονα y'y Matrix Profile με μέγεθος 20 fontsize="20".
Χάρη στη legend() γίνεται η εμφάνιση όλων των ετικετών.
Η show() δείχνει το τελικό γράφημα.

Π6 Εύρεση και εμφάνιση κορυφαίων σημείων

```
P_diff = P_beetle_fly - P_beetle_beetle

peaks, _ = find_peaks(P_diff)

idx = peaks[np.argsort(P_diff[peaks])][-10:]

plt.rcParams['xtick.direction']='out'
plt.figure(figsize=(20,6))
plt.suptitle("",fontsize="30")
plt.xlabel("Time", fontsize="20")
```

```
plt.ylabel("$P_{diff}=P_{Beetle,Fly}-P_{Beetle,Beetle}$",
fontsize="20")
plt.plot(idx,P_diff[idx],color="C1", marker="o", linewidth=0,
markersize=10)
plt.plot(P_diff,label="$P_{diff}$",linewidth=1.5)
plt.legend()
plt.show()
```

Για να γίνει πιο ευδιάκριτη η απόσταση ύψους ανάμεσα στα matrix profiles P_beetle_fly και P_beetle_beetle παίρνεται η διαφορά τους. Έπειτα αποθηκεύεται στο dataframe P_diff. Μέσα σε αυτό οι 10 υψηλότερες τιμές δείχνουν καλά υποψήφια shapelets. Η εύρεση των μεγαλύτερων τιμών υλοποιείται από την find_peaks() της SciPy που παίρνει σαν παράμετρο το P_diff και τα κρατάει στο peaks. Έπειτα με την peaks[np.argsort()][-10:], τα peaks τοποθετούνται από το μικρότερο στο μεγαλύτερο. Αυτή η ταξινόμηση πραγματοποιείται από τη NumPy. Μετά διαλέγονται οι 10 κορυφαίες τιμές, [-10:] και αποθηκεύονται στο idx.

Στο κομμάτι της οπτικοποίησης, γίνεται η αρχή με το ['xtick.direction'] = 'out' που δείχνει ότι θα είναι γυρισμένες προς τα έξω οι γραμμές στον άξονα χ'χ. Με το figsize=(20, 6) δίνεται το μέγεθος του πλαισίου figure για το γράφημα ύψος 6 και πλάτος 20. Το suptitle() παρέχει τον γενικό τίτλο και έχει δοθεί μέγεθος 30, fontsize="30".

Με την xlabel() ορίζεται τίτλος για τον άξονα χ'χ. Πιο συγκεκριμένα, το Time με μήκος 20, fontsize="20".
Με την plt.ylabel() αντιστοίχως δίνεται ο τίτλος, αλλά για τον άξονα y'y. Το P_diff = P_Beetle,Fly - P_Beetle, Beetle με μέγεθος 20, fontsize="20".
Η plot() χρησιμοποιείται για την μορφοποίηση του γραφήματος που θα εμφανιστεί για το P_diff. Μάλιστα, με το idx τοποθετεί τα 10 σημεία στο διάγραμμα, P_diff[idx]. Επίσης, τους δίνεται χρώμα πορτοκαλί, color="C1", με μορφή κύκλου, marker="o", πάχος 0, linewidth=0 και μέγεθος 10, markersize=10. Αμέσως μετά, με το plot() εμφανίζει και το γράφημα του P_diff με πάχος 1.5, linewidth=1.5. Ακόμη, προσθέτει μια ετικέτα P_diff πάνω δεξιά, label="P_{diff}". Με το legend() την παρουσιάζει. Το show() είναι αυτό που δείχνει όλη την οπτική αναπαράσταση.

Π7 Παρουσίαση των shapelets στο beetle

```
beetle_shapelets = []
for i in idx:
    shapelet=beetle_df.iloc[i:i+m,0] beetle_shapelets.append(shapelet)

plt.figure(figsize=(20,6))
plt.rcParams['xtick.direction']='out'
plt.xlabel("Time",fontsize="20")
plt.ylabel('Shape Outline', fontsize='20')
plt.plot(beetle_df, label="Beetle", linewidth = 1.5)
for i, shapelet in zip(idx, beetle_shapelets):
    plt.plot(range(i, i + m), shapelet, color="C1", linewidth=1)
```

```
plt.legend()
plt.show()
```

Το πρώτο βήμα αποτελεί η αρχικοποίηση ενός άδειου πίνακα με όνομα beetle_shapelets. Μέσα σε αυτόν θα κρατηθούν τα shapelets που βρίσκονται. Σε αυτό το σημείο ξεκινάει μια δομή επανάληψης που περνάει από όλα τα idx. Δηλαδή και για τα 10 κορυφαία σημεία με τα υποψήφια shapelets. Για κάθε ένα από αυτά, η .iloc[] κόβει μια υποσειρά από το beetle_df από το i μέχρι το i+m. Το 0 σημαίνει ότι θα χρησιμοποιηθεί η πρώτη στήλη από το beetle_df. Έπειτα αποθηκεύει την υπακολουθία στη μεταβλητή shapelet. Το m συμβολίζει το μέγεθος παραθύρου που επιλέχθηκε για τον υπολογισμό του matrix profile. Αν το idx του πρώτου shapelet ήταν το 20, τότε το iloc[] θα έκοβε ένα κομμάτι από 20 μέχρι και το 20+59= 79. Στη συνέχεια, με το .append() το περιεχόμενο της shapelet προστίθεται στον κενό πίνακα beetle_shapelets. Έτσι στο τέλος της επανάληψης, ο πίνακας θα είναι γεμάτος με 10 shapelet που αντιστοιχούν στα σημεία idx.

Η οπτικοποίηση αυτών πάνω στο διάγραμμα της ακολουθίας beetle είναι απλή διαδικασία. Με το ['xtick.direction'] = 'out' ορίζεται ότι οι γραμμές στον άξονα χ'χ θα βλέπουν προς τα έξω. Χάρη στο figure() τίθεται το μέγεθος του γενικού πλαισίου με πλάτος 20 και ύψος 6, figsize=(20, 6)

Το xlabel() ορίζει τίτλο προσαρμοσμένο για τον άξονα χ'χ. Ειδικότερα, το Time με μέγεθος 20, fontsize="20".
Το ylabel() υλοποιείται για τις μορφοποιήσεις του άξονα y'y. Του δίνεται ο τίτλος Shape Outline με μέγεθος 20, fontsize='20'. Η οπτική αναπαράσταση του γραφήματος γίνεται από την plt.show().
Την ετοιμασία του γραφήματος αναλαμβάνει η plot(). Χρησιμοποιούνται τα δεδομένα του beetle_df με πάχος γραμμής 1.5, linewidth = 1.5. Ακόμη μπαίνει μια ετικέτα Beetle στο δεξί μέρος του γραφήματος, label="Beetle".

Στη συνέχεια για κάθε τιμή i στο shapelet που είναι μέσα στον πίνακα beetle_shapelets, zip(idx, beetle_shapelets) θα γίνονται επαναλήψεις. Συνεπώς, και για τα 10 shapelets που βρέθηκαν. Με την plot() θα εμφανιστούν όλα πάνω στο γράφημα του beetle_df. Έχοντας συγκεκριμένο πεδίο από i μέχρι και i+m. Επίσης με πορτοκαλί χρώμα, color="C1" και πάχος γραμμής 1, linewidth=1. Μέσω της legend() παρουσιάζονται οι ετικέτες.

Π8 Αξιολόγηση των shapelets

```
test_df=df=
pd.read_csv('C:/Users/athen/Downloads/BeetleFly/BeetleFly_TESTNEW.csv')

y_train = train_df.iloc[:, 0]
y_test = test_df.iloc[:, 0]
```

Στο προηγούμενο στάδιο βρέθηκαν 10 shapelets. Ωστόσο δεν είναι όλα το ίδιο ικανά. Γι' αυτό πρέπει να γίνει η αξιολόγηση της διακριτικής ικανότητας των shapelets. Δηλαδή το πόσο καλά διαχωρίζουν ανάμεσα στις κλάσεις beetle και fly. Αυτό θα επιτευχθεί χτίζοντας 10 κατηγοριοποιητές δέντρου απόφασης.Βέβαια πρέπει να πρώτα να εκπαιδευτούν και μετά να αξιολογηθούν. Σε αυτό αξιοποιούνται το train set, train_df και το test set. Εισάγεται με βάση μια συγκεκριμένη διαδρομή το

αρχείου «pd.read_csv('C:/Users/athen/Downloads/BeetleFly/BeetleFly _TESTNEW.csv')». Η Pandas, pd το μετατρέπει σε dataframe και το αποθηκεύει στο test_df.

Μετά η iloc[:, 0] παίρνει μόνο την πρώτη στήλη, 0 του train_df και την κρατάει στο y_train. Έπειτα η iloc[:, 0] κρατάει πάλι την πρώτη στήλη, 0 αλλά από το test_df που εισήχθη πριν. Στη συνέχεια την αποθηκεύει στο y_test. Να σημειωθεί ότι η πρώτη στήλη έχει τις ετικέτες 1 και 2 .

```
def   distance_to_shapelet(data,   shapelet):

    data = np.asarray(data)
    X = np.empty(len(data))
    for i in range(len(data)):
        D  =  stumpy.mass(shapelet,  data[i])
        X[i] = D.min()

    return X.reshape(-1, 1)
```

Σε αυτό το στάδιο βρίσκεται η ακρίβεια των shapelets στη διαφοροποίηση των δύο κλάσεων. Επιτυγχάνεται υπολογίζοντας την Ευκλείδεια απόσταση ανάμεσα σε κάθε shapelet και στα δείγματα. Συνεπώς η συνάρτηση distance_to_shapelet που το υλοποιεί παίρνει παραμέτρους το data και το shapelet. Η NumPy μετατρέπει το data που ήρθε σε numpy array, np.asarray(data). Έπειτα φτιάχνει έναν άδειο πίνακα με μήκος όσο και το πλήθος του data, np.empty(len(data)). Κρατάει το αποτέλεσμα στο X. Ο πίνακας θα χρησιμεύσει στην πορεία, γι' αυτό αρχικοποιείται από τώρα. Στη συνέχεια ξεκινάει μια δομή επανάληψης. Αυτή θα πραγματοποιείται για κάθε δείγμα data. Μέσα σε αυτή βρίσκεται η απόσταση του shapelet και κάθε θέσης του data[i]. Για την υλοποίηση ευθύνεται η συνάρτηση MASS, stumpy.mass(shapelet, data[i]). Χάρη σε αυτή διαδικασία εκτέλεσης είναι γρήγορη. Μετά όλες οι αποστάσεις που προκύπτουν αποθηκεύονται στη μεταβλητή D. Στο επόμενο στάδιο από την D παίρνεται η μικρότερη τιμή και εισάγεται στον πίνακα X, X[i] = D.min(). Τέλος, επιστρέφεται το X, αφού πρώτα έχει μετατραπεί σε πίνακα δύο διαστάσεων. Η βιβλιοθήκη sklearn για την εκτέλεση του κατηγοριοποιητή δέντρου απόφασης πετάει λάθος αν ο πίνακας είναι μιας διάστασης. Γι' αυτό χρειάζεται το .reshape().

```
clf = tree.DecisionTreeClassifier()

for i, shapelet in enumerate(beetle_shapelets):
    X_train  =  distance_to_shapelet(train_df.iloc[:,  1:],  shapelet)
    X_test  =  distance_to_shapelet(test_df.iloc[:,  1:],  shapelet)
    clf.fit(X_train, y_train)
    y_pred  =  clf.predict(X_test.reshape(-1,  1))
    print(f"Accuracy for shapelet {i} =
{round(metrics.accuracy_score(y_test,  y_pred),  3)}")
```

Με το tree.DecisionTreeClassifier() φτιάχνεται ο κατηγοριοποιητής δέντρων απόφασης με όνομα clf. Αυτός θα πραγματοποιήσει την εκπαίδευση αλλά και την αξιολόγηση των shapelets. Μάλιστα για την εκπαίδευση χρησιμοποιεί την απόσταση από ένα shapelet.

Το παρακάτω θα εκτελείται για κάθε shapelet i μέσα στον πίνακα beetle_shapelets. Δηλαδή, τα αποτελέσματα θα αφορούν και τα 10 shapelets. Αρχικά καλείται η distance_to_shapelet() με την οποία υπολογίζεται η απόσταση του συνόλου εκπαίδευσης train_df από το shapelet. Το .iloc[:, 1:] τονίζει ότι θα παρθούν όλα τα δεδομένα από το train_df εκτός της πρώτης στήλης. Η ετικέτα δεν χρειάζεται για την ώρα. Το αποτέλεσμα αποθηκεύεται στο X_train. Η ίδια διαδικασία γίνεται και για το X_test. Με την distance_to_shapelet βρίσκεται πόσο απέχουν τα δείγματα του test_df από το shapelet.

Στο επόμενο στάδιο, με τη fit() εκπαιδεύει τον κατηγοριοποιητή clf χρησιμοποιώντας το X_train και y_train. Εφόσον τελεστεί αυτό, ήρθε η ώρα της πρόβλεψης. Στην ουσία αξιολογούνται τα shapelets ως προς την διακριτική τους ικανότητα. Αυτό πραγματοποιείται μέσω της predict(), εισάγοντας το X_test. Πρέπει να μετατραπεί σε 2 διαστάσεις, γι' αυτό και το .reshape(-1, 1). Έπειτα, με την print() εκτυπώνονται οι τιμές ακρίβειας για κάθε shapelet i. Μέσω της sklearn είναι δυνατή η εύρεση αυτών, την metrics.accuracy_score(). Σε αυτό εισάγονται τα y_test και y_pred. Σκοπός είναι η σύγκριση ανάμεσα στην πραγματική ετικέτα y_test και την εκτιμώμενη y_pred.

Η ίδια διαδικασία χρησιμοποιείται και για την εύρεση των shapelets της κλάσης Fly. Η μόνη διαφορά είναι εκεί υπάρχει beetle_df γίνεται fly_df και το αντίστροφο. Επίσης, τα αντίστοιχα matrix profiles θα είναι τα P_Fly_Beetle και P_Fly_Fly.