ΑΛΕΞΑΝΔΡΕΙΟ Τ.Ε.Ι ΘΕΣΣΑΛΟΝΙΚΗΣ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Πτυχιακή Εργασία

ΕΞΟΡΥΞΗ ΠΛΗΡΟΦΟΡΙΑΣ ΑΠΟ ΧΡΟΝΟΣΕΙΡΕΣ:ΠΡΟΓΡΑΜΜΑΤΙΣΤΙΚΗ ΥΛΟΠΟΙΗΣΗ ΜΕΤΡΩΝ ΟΜΟΙΟΤΗΤΑΣ

Της φοιτήτριας
Κάτσιανου Μαριλένας
Αρ.Μητρώου: 2379/03

Επιβλέπων καθηγητής
Καραμητόπουλος Λεωνίδας

Θεσσαλονίκη 2011

ΠΡΟΛΟΓΟΣ

Οι εξελίξεις στην τεχνολογία των ηλεκτρονικών υπολογιστών αλλά και ο σύγχρονος τρόπος λειτουργίας των επιχειρήσεων και οργανισμών, μας οδηγούν στο συμπέρασμα ότι ζούμε στην κοινωνία της πληροφορίας, όπου η μετατροπή των δεδομένων σε πληροφορία απαιτείται να οδηγεί στη μετατροπή της πληροφορίας σε γνώση. Η συλλογή τεράστιου όγκου δεδομένων κάθε είδους (ήχος, βίντεο, εικόνα κτλ), τα οποία λαμβάνονται μέσω πειραμάτων και παρατηρήσεων βρίσκουν ένα κύριο σύμμαχο στις μεγαλύτερης χωρητικότητας μνήμες, στους μεγαλύτερης ταχύτητας επεξεργαστές (hard disk, CD, DVD, USB) και στην ανάπτυξη διαφόρων οργάνων λεπτομερής καταγραφής μετρήσεων που έχουν κάνει την εμφάνισή τους. Το υλικό που συγκεντρώνεται καταγράφεται διαρκώς, με αποτέλεσμα τη δημιουργία πολύ μεγάλων βάσεων δεδομένων. Το ζήτημα που προκύπτει είναι εάν μπορούμε να εκμεταλλευτούμε αποτελεσματικά και αποδοτικά τέτοιες βάσεις δεδομένων.

Την τελευταία δεκαετία, αναπτύχθηκαν νέες, περισσότερο αποδοτικές μέθοδοι ανάλυσης δεδομένων, οι οποίες βοηθούν στην γρήγορη και αποτελεσματική εξερεύνηση τεράστιων βάσεων δεδομένων μέσω αυτόματων διαδικασιών με απώτερο σκοπό την ανακάλυψη της γνώσης. Το επιστημονικό πεδίο που ασχολείται με το αντικείμενο αυτό ονομάζεται Εξόρυξη Πληροφορίας (Data Mining). Σε πολλές εφαρμογές διαφορετικών δραστηριοτήτων, όπως για παράδειγμα το χρηματιστήριο, η βιομηχανία, η μετεωρολογία, η βιοπληροφορική, παράγονται δεδομένα σε μορφή χρονοσειρών. Η παρούσα εργασία επικεντρώνεται στην διαδικασία Εξόρυξης Πληροφορίας μέσα από Χρονοσειρές (Time Series Data Mining).

ΠΕΡΙΛΗΨΗ

Στη σύγχρονη εποχή η απόκτηση πληροφορίας και η εξόρυξη γνώσης από αυτή παίζει κυρίαρχο ρόλο. Η απεικόνιση της πληροφορίας με τη μορφή χρονοσειρών συναντάται σε πολλούς τομείς όπως η βιομηχανία, οι επιχειρήσεις, η ιατρική, η φυσική, η ψυχαγωγία. Στην παρούσα εργασία διερευνούμε διάφορες τεχνικές που χρησιμοποιούνται για την αποτελεσματική και αξιόπιστη εξόρυξη πληροφορίας από χρονοσειρές. Οι κύριες ενέργειες που πραγματοποιούνται με την εφαρμογή των τεχνικών αυτών είναι η συσταδοποίηση, η κατηγοριοποίηση, η ανακάλυψη κανόνων συσχέτισης και η ανάκτηση όμοιων εγγραφών. Στο επίκεντρο των ενεργειών αυτών βρίσκεται η έννοια της ομοιότητας και της μέτρησής της. Η χρονική διάσταση των δεδομένων όμως θέτει δυο βασικά ζητήματα τα οποία θα πρέπει να λαμβάνονται σοβαρά υπόψη κατά την αναζήτηση ομοιοτήτων. Το πρώτο ζήτημα είναι η επιλογή ενός κατάλληλου μέτρου ομοιότητας το οποίο θα επιτρέπει τον εντοπισμό όμοιων χρονοσειρών, οι οποίες δεν ταυτίζονται απαραιτήτως. Το δεύτερο ζήτημα αφορά στην αναπαράσταση των χρονοσειρών με στόχο τη μείωση της υψηλής διαστατικότητας τους (dimensionality).

Η πτυχιακή εργασία αυτή εστιάζεται κυρίως στη μέτρηση της ομοιότητας μεταξύ των χρονοσειρών με την εφαρμογή ενός μέτρου ομοιότητας, όπως είναι η Ευκλείδεια απόσταση, το Dynamic Time Warping (DTW) με ή χωρίς τους περιορισμούς Sakoe/Chiba και Itakura. Με την χρήση της βιβλιοθήκης ελεύθερου λογισμικού *java-ml* και στο περιβάλλον ανάπτυξης λογισμικού *Eclipse* τροποποιήσαμε τον αλγόριθμο DTW, υλοποιήσαμε τα φίλτρα Sakoe/Chiba και Itakura και τα ενσωματώσαμε στη βιβλιοθήκη java-ml. Επίσης, πραγματοποιήσαμε ένα σύνολο πειραμάτων για να αξιολογήσουμε την αποδοτικότητα και την αξιοπιστία αυτών των τεχνικών αναζήτησης ομοιότητας χρησιμοποιώντας τον αλγόριθμο του Εγγύτερου Γείτονα (1-Nearest Neighbor) της java-ml. Η πειραματική αξιολόγηση αφορούσε τους αλγόριθμους της Ευκλείδειας απόστασης, του Dynamic Time Warping, του Sakoe/Chiba και του Itakura, ώστε να εντοπίσουμε τις διαφορές και τις ομοιότητες τους σε σχέση με τα ποσοστά εσφαλμένης κατηγοριοποίησης και τους χρόνους απόκρισης τους κατά τη διάρκεια της αναζήτησης όμοιων χρονοσειρών.

ABSTRACT

In our time and age, the acquisition of knowledge as well as knowledge mining plays a dominant role. The representation of information in the form of time series occurs in several fields such as industry, business, medicine, science and entertainment. In this project various techniques which are used in the efficient and reliable data mining are studied. The main tasks that are committed with the application of these techniques are clustering, classification, association rule discovery and query by content. At the core of these tasks lies the concept and measure of similarity. However, the time dimension poses two basic issues which should be taken under consideration during the process of similarity search. The first issue is the selection of the appropriate measure of similarity that allows the detection of similar time series, which are not necessarily identical. The second issue refers to the representation of time series in order to reduce their intrinsically high dimensionality.

The current paper mainly focuses on the issue of measuring similarity among time series with the utilization of similarity measures such as the Euclidean distance, the Dynamic Time Warping (DTW) with and without the corresponding restrictions of Sakoe/Chiba and Itakura. We utilized the library of the open source java-ml, and in the environment of the Eclipse software, we modified the existing DTW algorithm, we developed the codes for the Sakoe/Chiba and Itakura restrictions, and we incorporated them to the java-ml library. In addition to that, we conducted a number of experiments, in order to evaluate the efficiency and reliability of the above techniques with respect to similarity search, by utilizing the 1-Nearest Neighbor algorithm of java-ml. The experimental evaluation involved the algorithms for the Euclidean distance, the Dynamic Time Warping, the Sakoe/Chiba and the Itakura restrictions, in order to identify their differences and similarities with respect to classification error and to their run-time during the process of searching similar time series.

ΕΥΧΑΡΙΣΤΙΕΣ

Θα ήθελα να ευχαριστήσω ιδιαίτερα τον καθηγητή μου Λεωνίδα Καραμητόπουλο για την επίβλεψη αυτής της πτυχιακής εργασίας και την καθοριστική συμβολή του στην ολοκλήρωσή της.

Επίσης θερμά ευχαριστώ στον φίλο και συνάδελφο Κώστα Γεωργιάδη για τις συμβουλές του και τη πολύτιμη βοήθεια του κατά τη διάρκεια εκπόνησης της εργασίας αυτής.

Τέλος, ευχαριστώ την οικογένεια μου για την αμέριστη συμπαράσταση , υποστήριξη και βοήθεια όλα αυτά τα χρόνια.

ΠΕΡΙΕΧΟΜΕΝΑ

ΚΕΦΑΛΑΙΟ 1ο

ΕΞΟΡΥΞΗ ΠΛΗΡΟΦΟΡΙΑΣ ΑΠΟ ΧΡΟΝΟΣΕΙΡΕΣ

ΚΕΦΑΛΑΙΟ 2ο

ΑΝΑΖΗΤΗΣΗ ΟΜΟΙΩΝ ΧΡΟΝΟΣΕΙΡΩΝ

ΚΕΦΑΛΑΙΟ 3ο

ΜΕΤΡΑ ΑΠΟΣΤΑΣΗΣ

ΚΕΦΑΛΑΙΟ 1ο

ΕΞΟΡΥΞΗ ΠΛΗΡΟΦΟΡΙΑΣ ΑΠΟ ΧΡΟΝΟΣΕΙΡΕΣ

1.1 ΕΙΣΑΓΩΓΗ

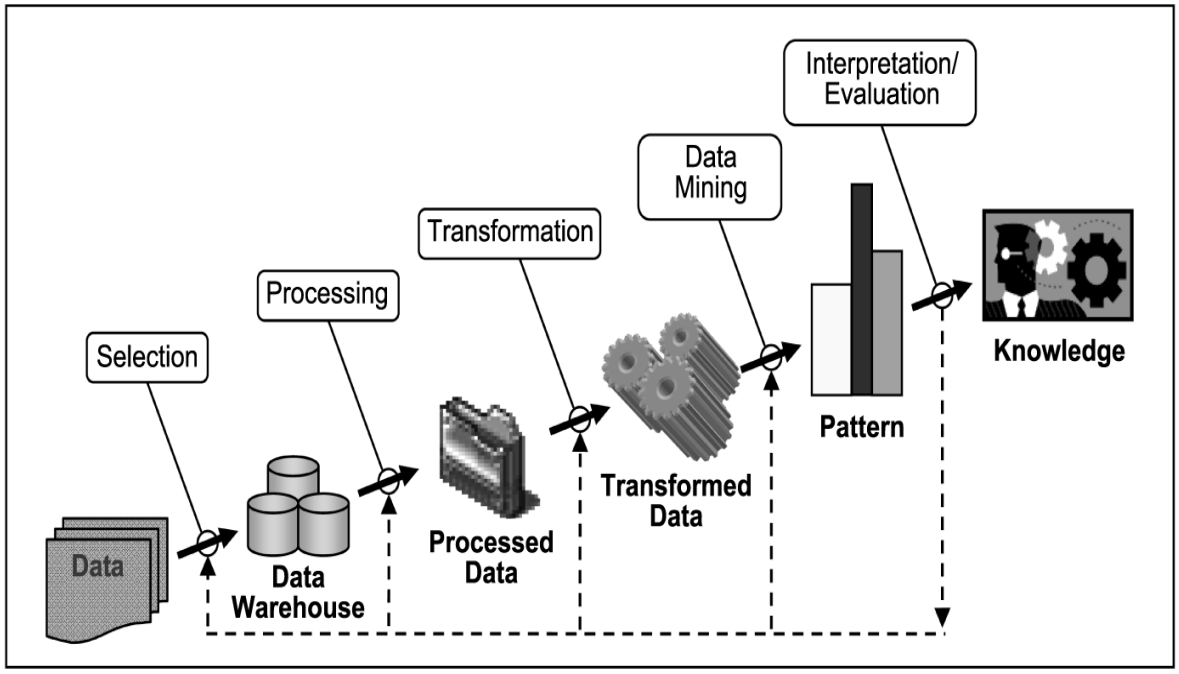
Στο κεφάλαιο αυτό πραγματοποιείται μία εισαγωγή στη διαδικασία ανακάλυψης γνώσης σε μια βάση δεδομένων και στη διαδικασία εξόρυξης πληροφορίας από αυτή. Προσεγγίζεται η έννοια των χρονοσειρών και περιγράφονται τα στάδια και οι στόχοι της διαδικασίας εξόρυξης πληροφορίας μέσω της ανάλυσης χρονοσειρών. Κατόπιν αναφέρονται διάφοροι τομείς της σημερινής εποχής όπου γίνεται ευρεία χρήση χρονοσειρών καθώς και τα ιδιαίτερα ζητήματα που εγείρονται στην διαδικασία εξόρυξης πληροφορίας μέσω ανάλυσης χρονοσειρών.

1.2 ΔΙΑΔΙΚΑΣΙΑ ΑΝΑΚΑΛΥΨΗΣ ΓΝΩΣΗΣ ΣΕ ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ

Η ανακάλυψη γνώσης σε βάσεις δεδομένων (Knowledge Discovery in Databases) είναι μια αυτοματοποιημένη διαδικασία μέσω της οποίας γίνεται προσπάθεια διερευνητικής ανάλυσης και μοντελοποίησης τεράστιων αποθηκών δεδομένων. Πρόκειται για μια συγκροτημένη μεθοδολογία αναγνώρισης έγκυρων και πρωτότυπων προτύπων μέσα από μεγάλους και περίπλοκους πίνακες δεδομένων, με στόχο τα πρότυπα που θα προκύψουν να είναι χρήσιμα και κατανοητά. Σύμφωνα με τον Fayyad et al., 1996 [9] η «*Ανακάλυψη Γνώσης σε Βάσεις Δεδομένων*» αποτελεί μία συγκροτημένη μεθοδολογία αναγνώρισης έγκυρων, καινοτόμων, εν δυνάμει χρήσιμων και σε τελευταία ανάλυση κατανοητών προτύπων μέσα σε δεδομένα.

Η *Εξόρυξη Πληροφορίας* (Data Mining) αποτελεί το σημαντικότερο βήμα στην παραπάνω διαδικασία και συνίσταται στην ανάπτυξη μεθόδων και αλγορίθμων, ικανών να επεξεργαστούν αποδοτικά τεράστιες βάσεις δεδομένων με σκοπό την ανακάλυψη προτύπων τα οποία πιθανόν να αποτελούν γνώση. Σύμφωνα με τον Kamath [12], η διαδικασία *Εξόρυξης Πληροφορίας* αποτελεί μια ημι-αυτοματοποιημένη επαναληπτική διαδικασία με απώτερο σκοπό την ανακάλυψη ενδιαφερόντων προτύπων συμπεριφοράς.

Υπάρχει σαφής διαφορά μεταξύ των όρων Knowledge Discovery in Databases (KDD) και Data Mining (DM). Ο όρος KDD χρησιμοποιείται για την περιγραφή ολόκληρης της διαδικασίας ανακάλυψης γνώσης από ένα σύνολο δεδομένων, ενώ ο όρος DM αναφέρεται στις τεχνικές που χρησιμοποιούνται για την ανακάλυψη πληροφορίας (Εικόνα 1.1).



Εικόνα 1.1 Διαδικασία KDD

Πηγή: http://www.emeraldinsight.com/content_images/fig/2630240410005.png

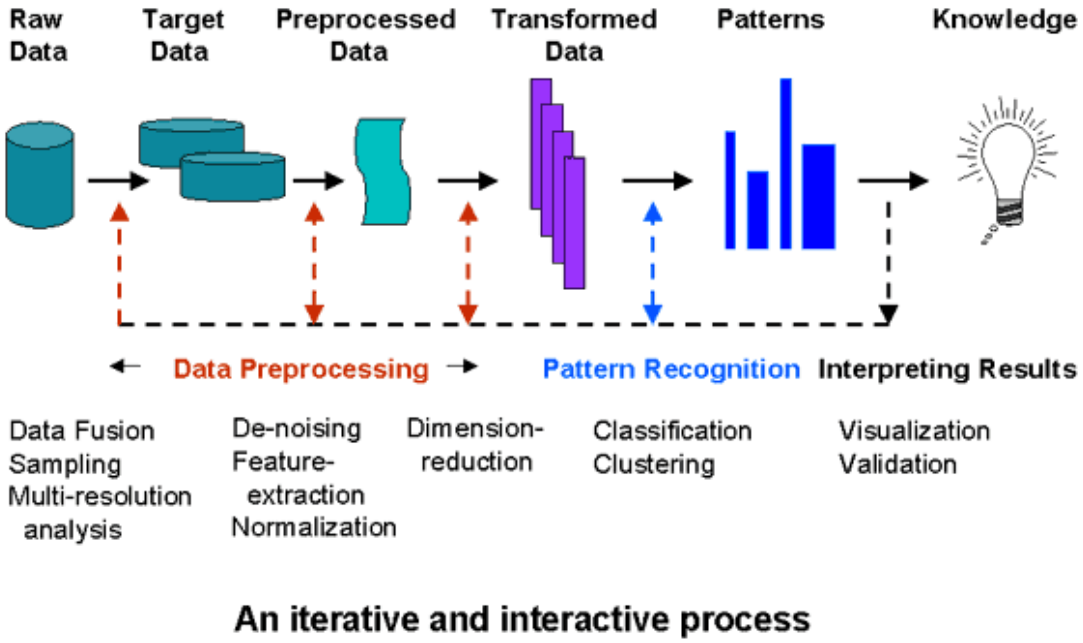
Η ανίχνευση προτύπων συμπεριφοράς αποτελεί μια εργασία με ιδιαίτερη σημασία καθώς μας επιτρέπει να δούμε μια μεγάλη βάση δεδομένων με περιληπτικό τρόπο και μέσα απ' αυτή να οδηγηθούμε στην ανακάλυψη νέας γνώσης. Η νέα αυτή γνώση επιτρέπει την επίτευξη κάποιων προκαθορισμένων στόχων της διαδικασίας εξόρυξης δεδομένων. Οι σημαντικότερες ενέργειες που πραγματοποιούνται με την εφαρμογή μεθόδων εξόρυξης δεδομένων σύμφωνα με τον Fayyad et. al (1996) [9] είναι οι εξής:

- Συσταδοποίηση (clustering)
- Κατηγοριοποίηση (classification)
- Εντοπισμός Παρεκκλίσεων / Μεταβολών (anomaly detection)
- Σύνοψη / Περιγραφή
- Μοντελοποίηση Σχέσεων / Εξαρτήσεων(association rules)

Βασικοί στόχοι της εξόρυξης γνώσης είναι η περιγραφή και η πρόβλεψη. Δηλαδή η αναγνώριση των προτύπων που επικρατούν σε ένα μεγάλο σύνολο δεδομένων και η δημιουργία προβλέψεων των τιμών και των συμπεριφορών κάποιων μεταβλητών. Η αναγνώριση των προτύπων γίνεται με την εφαρμογή πολύπλοκων και αρκετά εξειδικευμένων αλγορίθμων. Παρά την ύπαρξη πολλών τεχνικών που βοηθούν σ' αυτή τη διαδικασία, δεν υπάρχει ένα μοντέλο ή μια προσέγγιση που να τα περιλαμβάνει όλα. Σημαντική ανάπτυξη θα ήταν η δημιουργία μιας εξειδικευμένης «γλώσσας ερωτήσεων» η οποία θα περιελάμβανε τις παραδοσιακές SQL συναρτήσεις όπως επίσης και τις πιο πολύπλοκες ερωτήσεις, τα OLAP αιτήματα. Ωστόσο μερικές από τις πιο συχνά χρησιμοποιούμενες τεχνικές στη διαδικασία της εξόρυξης γνώσης είναι:

- Δημιουργία δέντρων απόφασης. Πρόκειται για σχέδια αναπαράστασης της γνώσης που προκύπτει, βασισμένα σε δενδρικές δομές που αντιπροσωπεύουν σύνολα αποφάσεων. Η προσέγγιση των δέντρων απόφασης είναι πολύ χρήσιμη στα προβλήματα κατηγοριοποίησης νέων δεδομένων (classification).
- Μέθοδος Εγγύτερου Γείτονα (Nearest Neighbor). Είναι μια τεχνική κατηγοριοποίησης, η οποία ταξινομεί ένα αντικείμενο σε μία ομάδα, από ένα σύνολο ομάδων, με βάση ποιο σύνολο απ' αυτά έχει παρόμοια χαρακτηριστικά με το αρχικό αντικείμενο. Η πολυπλοκότητα αυτής της μεθόδου εξαρτάται από το πλήθος των εγγραφών.
- Επαγωγή κανόνων. Πρόκειται για τεχνικές που περιλαμβάνουν την άντληση των χρήσιμων if-then κανόνων από δεδομένα.
- Δημιουργία νευρωνικών δικτύων. Είναι μη-γραμμικά μοντέλα πρόβλεψης που η δομή τους μοιάζει με τα βιολογικά νευρικά δίκτυα. Διαθέτουν την ικανότητα να αντλούν νόημα από περίπλοκα ή ασαφή στοιχεία και βοηθούν στην ανακάλυψη προτύπων και συμπεριφορών των δεδομένων που μελετούνται. Παρουσιάζουν υψηλό αριθμό ακρίβειας αλλά είναι αρκετά πιο δύσκολα στην κατανόηση σε σχέση με τα δέντρα απόφασης.
- Γενετικοί αλγόριθμοι. Είναι τεχνικές βελτιστοποίησης που χρησιμοποιούν το γενετικό συνδυασμό, τη μεταλλαγή, τη φυσική επιλογή, και άλλες έννοιες της φυσικής εξέλιξης.

Συμπερασματικά η διαδικασία εξόρυξης γνώσης αποτελείται από αποδοτικές τεχνικές που μας βοηθούν να αναλύσουμε πολύ μεγάλες συλλογές από δεδομένα και να εξάγουμε χρήσιμες πληροφορίες, τις οποίες είναι δύσκολο να διακρίνει κανείς εξ' αρχής λόγω του μεγάλου όγκου δεδομένων. Στην πράξη η εξόρυξη γνώσης δεν συνίσταται απλά στην εφαρμογή ενός από τους παραπάνω αλγόριθμους (Εικόνα 1.2). Τα δεδομένα συχνά περιέχουν θόρυβο ή είναι ημιτελή και αν αυτό δεν γίνει αντιληπτό και δεν διορθωθεί, τότε είναι πιθανό πολλά ενδιαφέροντα πρότυπα να μην ανιχνευτούν, ενώ η αξιοπιστία των εντοπισμένων προτύπων να είναι χαμηλή.



Εικόνα 1.2 Διαδικασία Data Mining
Πηγή: http://www.dwreview.com/Data_mining/Images/Data_Mine_Steps.gif

1.3 ΕΙΣΑΓΩΓΗ ΣΤΙΣ ΧΡΟΝΟΣΕΙΡΕΣ

Με τον όρο ***χρονοσειρά (Χ-Σ)*** ή ***χρονική σειρά (time series)*** εννοούμε μια ακολουθία από μετρήσεις/παρατηρήσεις X_t ενός χαρακτηριστικού Χ μέσα σε μία χρονική περίοδο.

Συνήθως μια χρονοσειρά συμβολίζεται ως:

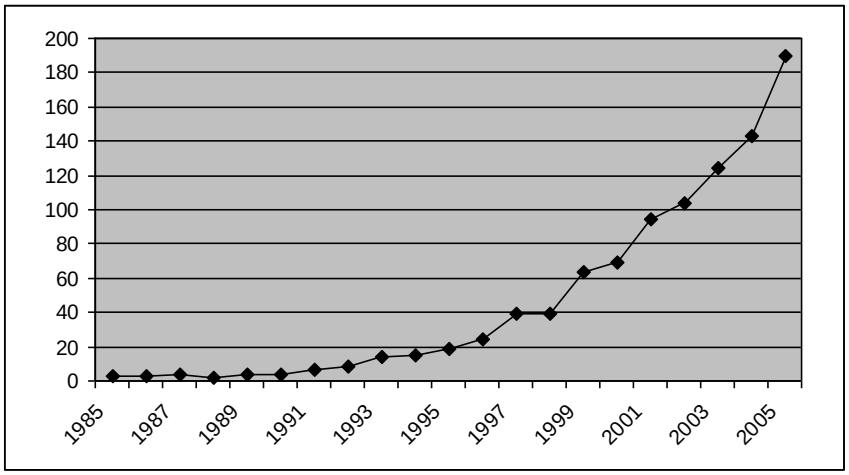
$X_1, X_2, \ldots\ldots, Xn$ n: μέγεθος Χ-Σ ή

X_t, t = 1,2,3, …..n

Αναπαριστώνται συχνά με διαγράμματα, όπου στον οριζόντιο άξονα xx΄ αναπαριστάται ο χρόνος t και στον κατακόρυφο άξονα yy΄ καταγράφονται οι μετρήσεις του χαρακτηριστικού / μεταβλητής που μελετάται τις αντίστοιχες χρονικές στιγμές (Εικόνα 1.3).

Οι χρονοσειρές διακρίνονται σε δυο κατηγορίες:

- Συνεχείς χρονοσειρές: Η χρονική περίοδος είναι συνεχής
- Διακριτές χρονοσειρές: Η χρονική περίοδος περιλαμβάνει διακριτές τιμές



Εικόνα 1.3: Πλήθος άρθρων που περιλαμβάνουν τον όρο "Time Series" στον τίτλο τους.

Παρακάτω αναφέρονται ορισμένα ποιοτικά χαρακτηριστικά των χρονοσειρών (Ζήσος, 2006) [1]:

1. *Στασιμότητα (Stationary).* Μια χρονοσειρά θεωρείται στάσιμη, όταν οι στατιστικές ιδιότητες της χρονοσειράς δεν αλλάζουν με το χρόνο. Ειδικότερα μια χρονοσειρά θα είναι στάσιμη αν έχει μέση τιμή και διακύμανση μη μεταβαλλόμενα με το χρόνο.

2. *Τάση (Trend).* Το στοιχείο της τάσης περιγράφει τη μακροχρόνια συμπεριφορά μιας χρονοσειράς. Ειδικότερα, αν για μία μακρά χρονική περίοδο οι τιμές μιας χρονοσειράς τείνουν να αυξάνονται ή να μειώνονται, τότε λέμε ότι η σειρά των παρατηρήσεων παρουσιάζει μακροχρόνια τάση.

3. *Περιοδικότητα ή Εποχικότητα (Seasonal).* Το στοιχείο της περιοδικότητας ή εποχικότητας περιγράφει κανονικές επαναλαμβανόμενες διακυμάνσεις των τιμών μιας χρονοσειράς σε κάποια χρονική περίοδο που αυτές μπορεί να αντιστοιχούν σε ένα χρόνο, μια εποχή του χρόνου, ένα μήνα ή και μια βδομάδα. Οι εποχικές

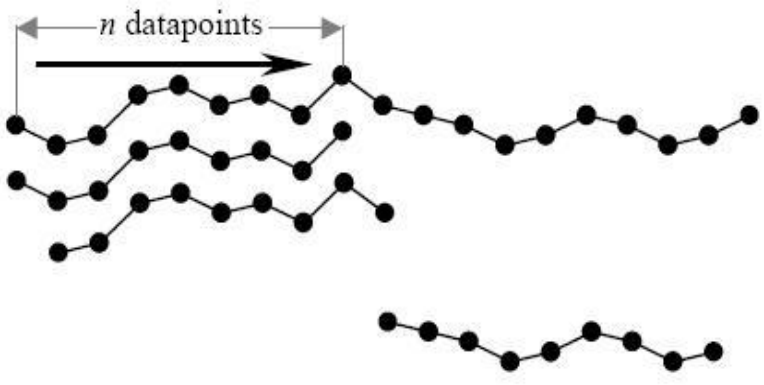
μεταβολές είναι ρυθμικές, επαναλαμβανόμενες στο χρόνο και φέρονται να αναφέρονται σε κάποιο πρότυπο, που ακολουθεί το
μετρούμενο μέγεθος, το οποίο αποτυπώνεται στις τιμές της χρονοσειράς κατά τη διάρκεια αντίστοιχων μηνών σε διαδοχικά έτη.

4. *Κυκλικότητα (Cyclical).* Το στοιχείο της κυκλικότητας αναφέρεται σε μακράς περιόδου ταλαντεύσεις των τιμών, γύρω από μία γραμμή ή καμπύλη τάσης. Οι κυκλικές μεταβολές διαφέρουν από τις περιοδικές, καθώς είναι μεγαλύτερης διαρκείας και δεν παρουσιάζουν μεγάλη περιοδικότητα.

5. *Ασυνέχειες (Discontinuity):* Ασυνήθιστες τιμές (Outliers). Οι ασυνέχειες αποτελούν «παραμορφώσεις» των χρονοσειρών σε ορισμένα σημεία και οφείλονται κυρίως σε σφάλματα των οργάνων μέτρησης, των μετρούμενων μεγεθών, τις συγκεκριμένες χρονικές στιγμές. Οι τιμές της χρονοσειράς στα σημεία αυτά αποτελούν σφάλματα οργάνων. Επειδή οι τιμές-σφάλματα οργάνων αποτελούν πρόβλημα στην επεξεργασία των χρονοσειρών, απαλείφονται ή αντικαθίστανται, προκειμένου να επιτευχθεί εξομάλυνση της χρονοσειράς.

6. *Τυχαιότητα (Randomization).* Η τυχαιότητα έχει να κάνει με απροσδόκητες διακυμάνσεις των τιμών της χρονοσειράς που έχουν να κάνουν είτε με φυσικά αίτια, είτε με ξαφνικά και απρόβλεπτα συμβάντα.

Βασικό χαρακτηριστικό των χρονοσειρών είναι το μεγάλο μέγεθος τους γεγονός το οποίο δυσκολεύει τις διαδικασίες ανάλυσης. Γι' αυτό σε κάποιες περιπτώσεις είναι απαραίτητη η τμηματοποίηση των προς ανάλυση χρονοσειρών σε μικρότερα μεγέθη τα οποία ονομάζονται υπό-ακολουθίες (Εικόνα 1.4).



Πηγή: Keogh E. & Pazzani M. (2000).“A Simple Dimensionality Reduction Technique for Fast Similarity Search in Large Time Series Databases” In Proceedings of the 4th Pacific-Asia Conference on Knowledge Discovery and Data Mining. Kyoto.

Εικόνα 1.4 Υπό-ακολουθίες μιας χρονοσειράς

1.4 ΕΞΟΡΥΞΗ ΠΛΗΡΟΦΟΡΙΑΣ ΑΠΟ ΧΡΟΝΟΣΕΙΡΕΣ: ΔΙΑΔΙΚΑΣΙΑ - ΣΤΟΧΟΙ

Η διαδικασία εξόρυξης πληροφορίας μέσω ανάλυσης χρονοσειρών συνίσταται στην ανάπτυξη και προσαρμογή μεθόδων και αλγορίθμων εξόρυξης πληροφορίας έτσι ώστε να μελετάται η τιμή ενός γνωρίσματος καθώς μεταβάλλεται στο χρόνο και να καθίσταται δυνατή η εξαγωγή συμπερασμάτων για τη συμπεριφορά και εξέλιξη του γνωρίσματος αυτού.

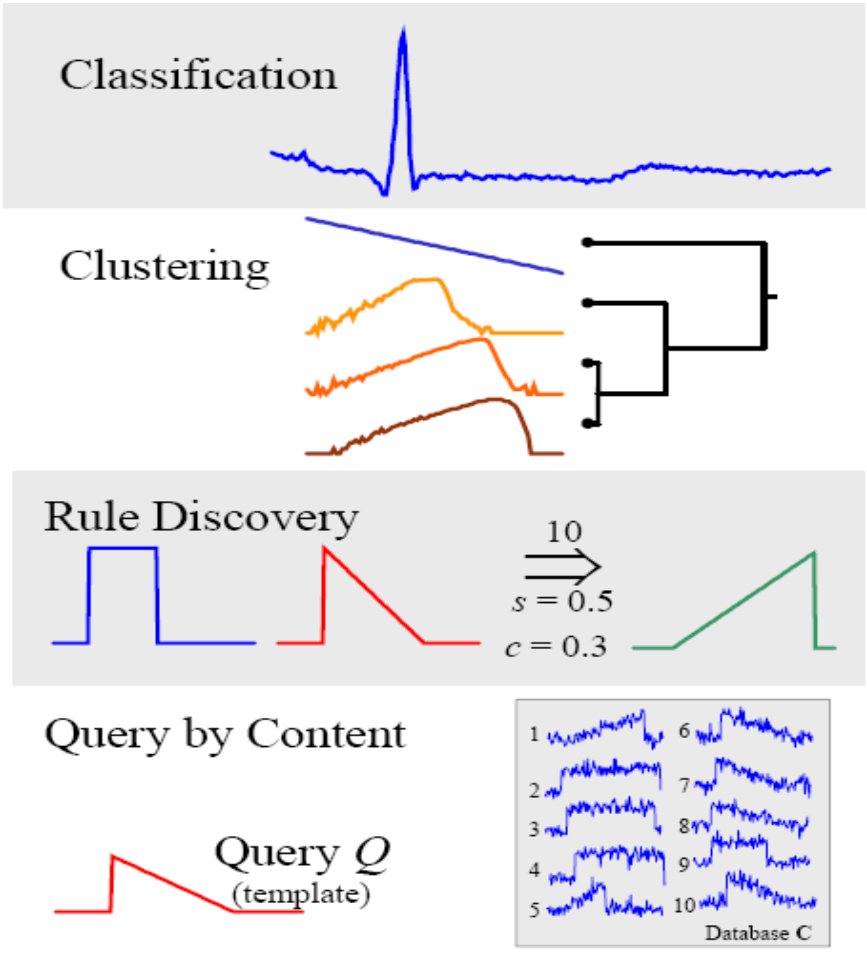
Η διαδικασία εξόρυξης γνώσης μέσω ανάλυσης χρονοσειρών περιλαμβάνει τα εξής βήματα σύμφωνα με τον Ramakrishnan(2002) [29]:

1. *Καθορισμός στόχων* (Lin, Keogh και Lonardi 2005) [26]. Οι σημαντικότεροι στόχοι αυτής της διαδικασίας είναι οι εξής (Εικόνα 1.5):

- *Κατηγοριοποίηση / Ταξινόμηση:* Ταξινόμηση νέων αντικειμένων σε ήδη υπάρχουσες κατηγορίες (classification). Στην κατηγοριοποίηση έχουμε διαθέσιμο ένα σύνολο κατηγοριών (classes) και μας ζητείται να εντάξουμε ένα νέο αντικείμενο σε μία από τις υπάρχουσες κατηγορίες. Στην περίπτωσή μας, ένα αντικείμενο είναι μία χρονοσειρά. Έτσι κάθε νέα χρονοσειρά ταξινομείται στις ήδη γνωστές κατηγορίες χρονοσειρών που υπάρχουν σε μια βάση δεδομένων. Οι κατηγορίες αυτές προσδιορίζονται από τα χαρακτηριστικά των αντικειμένων που είναι ήδη γνωστό ότι ανήκουν σ' αυτές. Μια τέτοια διαδικασία μπορεί να εφαρμοστεί για παράδειγμα σε έναν σταθμό ελέγχου ασφαλείας αεροδρομίου ώστε να ελεγχθεί αν οι επιβάτες είναι πιθανοί τρομοκράτες ή εγκληματίες. Για να γίνει αυτό, σαρώνεται με ειδικό σαρωτή το πρόσωπο κάθε επιβάτη και αναγνωρίζεται το βασικό του πρότυπο (απόσταση μεταξύ ματιών, μέγεθος και σχήμα στόματος, σχήμα κεφαλιού, κλπ). Αυτό το πρότυπο συγκρίνεται με μια βάση δεδομένων για να διαπιστωθεί εάν ταιριάζει με κάποια πρότυπα που σχετίζονται με γνωστοποιημένους παραβάτες (Margaret H.Dunham 2004) [28].
- *Συσταδοποίηση (clustering).* Η Συσταδοποίηση είναι παρόμοια με την κατηγοριοποίηση μόνο που εδώ οι κατηγορίες δεν είναι προκαθορισμένες αλλά δημιουργούνται κατά την εφαρμογή της. Το ζητούμενο είναι να δημιουργήσουμε ομάδες (συστάδες) όμοιων χρονοσειρών χωρίς καμία γνώση για προϋπάρχουσες κατηγορίες. Για παράδειγμα κάποια αλυσίδα πολυκαταστημάτων θέλει να δημιουργήσει ειδικούς καταλόγους που

στοχεύουν σε διάφορες δημογραφικές ομάδες με βάση γνωρίσματα όπως το εισόδημα, ο τόπος διαμονής, η ηλικία, το βάρος των δυνητικών πελατών τους. Προκειμένου να καθορίσει σε ποιους από τους πελάτες θα σταλεί διαφημιστικό υλικό και για να δημιουργηθούν καινούργιοι και πιο συγκεκριμένοι κατάλογοι, η εταιρεία κάνει ομαδοποίηση των πιθανών πελατών βασιζόμενη στις προκαθορισμένες τιμές των γνωρισμάτων τους. Τα αποτελέσματα της ομαδοποίησης χρησιμοποιούνται στη συνέχεια από τη διεύθυνση ώστε να δημιουργηθούν ειδικοί κατάλογοι στο κατάλληλο τμήμα του πληθυσμού (Margaret H.Dunham 2004) [28].

- *Ανακάλυψη κανόνων συσχέτισης (Association rule discovery).* Η ανακάλυψη κανόνων συσχέτισης είναι μια αρκετά δημοφιλή μέθοδος στο πεδίο της Εξόρυξης Πληροφορίας, η οποία βοηθά στο να αποκαλυφθούν κρυμμένες συσχετίσεις μεταξύ των χαρακτηριστικών μιας μεγάλης βάσης δεδομένων. Οι κανόνες αυτοί χρησιμεύουν στην κατανόηση των δεδομένων που εξετάζονται καθώς επίσης και στην πρόβλεψη μελλοντικών συμπεριφορών αυτών. Ένα παράδειγμα κανόνα συσχέτισης είναι το εξής: όταν η τιμή κλεισίματος μιας συγκεκριμένης μετοχής παρουσιάζει ένα συγκεκριμένο πρότυπο πτωτικής τάσης σε μία περίοδο δέκα ημερών, είναι πολύ πιθανό μετά από τρεις ημέρες να παρουσιάσει ένα συγκεκριμένο (διαφορετικό) πρότυπο απότομης ανόδου για τις επόμενες δύο ημέρες.
- *Ανάκτηση ομοίων αντικειμένων (Query by content / Similarity Search).* Πρόκειται για μια διαδικασία η οποία έχει ευρεία εφαρμογή σε βάσεις δεδομένων χρονοσειρών. Εστιάζει στην ανάκτηση χρονοσειρών από μια βάση δεδομένων ομοίων με μια δοθείσα χρονοσειρά (query). Για παράδειγμα, μία επενδυτική εταιρεία επιθυμεί να εντοπίσει όλες εκείνες τις μετοχές που έχουν παρόμοια διαχρονική εξέλιξη με μία συγκεκριμένη μετοχή (query).



Εικόνα 1.5 Στόχοι της διαδικασίας εξόρυξης γνώσης μέσω ανάλυσης χρονοσειρών

Πηγή : Lin J., Keogh E., Lonardi S., (2005). "Visualizing and Discovering Nontrivial Patterns In Large Time Series Databases", Information Visualization, Vol 4, No. 2

2. *Δειγματοληψία (sampling)*. Αφορά την επιλογή των δεδομένων που θα μελετηθούν, ποιες παράμετροι θα ληφθούν υπόψη όπως επίσης και την επιλογή του κατάλληλου αλγορίθμου ώστε να επιτευχθούν οι αρχικοί στόχοι.

3. *Προ-επεξεργασία δεδομένων (data preprocessing).* Αφού επιλεχθούν τα προς ανάλυση δεδομένα είναι απαραίτητο να περάσουν από μια επεξεργασία καθώς το πιο πιθανό είναι να περιέχουν θόρυβο ή/και άλλες στρεβλώσεις, οι οποίες αν δεν ανιχνευτούν έγκαιρα μπορεί να δώσουν αναξιόπιστα πρότυπα συμπεριφοράς κατά τη διαδικασία εξόρυξης πληροφορίας.

4. *Ανακάλυψη προτύπων συμπεριφοράς (pattern recognition)*. Αποτελεί το κυρίως στάδιο της διαδικασίας εξόρυξης γνώσης στο οποίο ανιχνεύονται τα πρότυπα συμπεριφοράς των προς μελέτη δεδομένων, ενώ ελέγχεται επίσης το κατά πόσο

είναι έγκυρα. Αυτό συμβαίνει ώστε να βεβαιωθεί ότι τα αποτελέσματα που προέκυψαν είναι όσο το δυνατόν πιο αξιόπιστα.

5. *Παρουσίαση και αξιολόγηση των αποτελεσμάτων (interpretation of results)*. Αποτελεί το τελικό στάδιο της διαδικασίας εξόρυξης γνώσης όπου επιτυγχάνεται ο κύριος σκοπός δηλαδή, η συγκεκριμενοποίηση της αποκτηθείσας γνώσης, την οποία μπορεί να χρησιμοποιήσει κατάλληλα ο άνθρωπος για να επιτύχει τους εκάστοτε στόχους του.

1.5 ΕΞΟΡΥΞΗ ΠΛΗΡΟΦΟΡΙΑΣ ΑΠΟ ΧΡΟΝΟΣΕΙΡΕΣ : ΧΡΗΣΙΜΟΤΗΤΑ

Οι χρονοσειρές στη σημερινή εποχή συναντώνται σχεδόν σε όλους τους τομείς της ανθρώπινης δραστηριότητας, όπως η οικονομία, η ιατρική, η κοινωνία, η επιστήμη, η μουσική κ.α.

Στον τομέα της οικονομίας οι χρονοσειρές έχουν ευρεία χρήση αναπαριστώντας για παράδειγμα την εξέλιξη μιας μετοχής κατά τη διάρκεια ενός εξαμήνου ή τις τιμές πώλησης ενός προϊόντος σε διάστημα δυο ετών. Η μελέτη και ανάλυση των παραπάνω δεδομένων έχει ως απώτερο σκοπό την απόκτηση ανταγωνιστικού πλεονεκτήματος στην αγορά για την εκάστοτε επιχείρηση.

Επίσης ιδιαίτερα χρήσιμη είναι η χρήση των χρονοσειρών στον τομέα της ιατρικής. Τα καρδιογραφήματα των ασθενών και οι μετρήσεις της πίεσης του αίματος κατά τη διάρκεια μιας χρονικής περιόδου αναπαριστώνται ως χρονοσειρές των οποίων η ανάλυση και μελέτη είναι χρήσιμη για τη διάγνωση ασθενειών. Εκτός από την ιατρική και στον τομέα της βιολογίας συναντάμε συχνά χρονοσειρές των οποίων η μελέτη βοηθά στην ανακάλυψη νέων φαρμάκων, (πχ. γονιδιακές ακολουθίες DNA).

Μεγάλη αύξηση της χρήση των χρονοσειρών παρουσιάζεται τελευταία και σε μελέτες κοινωνικών και πολιτικών φαινομένων, όπως μηνιαία ποσοστά ανεργίας, εγκληματικότητα ενός νομού ή μιας χώρας, δυσαρέσκεια για την πολιτική της εκάστοτε κυβέρνησης. Στοιχεία τα οποία αν αναλυθούν συμβάλλουν στην κατανόηση των αιτιών καθώς και στη λήψη αποδοτικών μέτρων.

Ιδιαίτερο ενδιαφέρον παρουσιάζει η χρήση χρονοσειρών για την αναπαράσταση ενός μουσικού κομματιού ανάλογα με τις τιμές έντασης του ήχου, μιας εικόνας ανάλογα με τη φωτεινότητα των χρωμάτων της ή ακόμα και ενός κειμένου ανάλογα με το πλήθος εμφάνισης κάποιων χαρακτήρων. Η μελέτη και

ανάλυση τέτοιου είδους δεδομένων υπό τη μορφή χρονοσειρών μπορεί να βοηθήσει πολύ σε περιπτώσεις υποκλοπών και πλαστογραφήσεων.

Τέλος, εκτεταμένη χρήση χρονοσειρών παρατηρείται και στην αναπαράσταση γεωλογικών και φυσικών φαινομένων. Όπως η θερμοκρασία ενός μήνα σε μια περιοχή, η ετήσια χιονόπτωση, ή η σεισμική συχνότητα της περιοχής, μετρήσεις οι οποίες βοηθούν τους επιστήμονες στην εξαγωγή χρήσιμων συμπερασμάτων για την εξέλιξη των φαινομένων αυτών.

1.6 ΕΞΟΡΥΞΗ ΠΛΗΡΟΦΟΡΙΑΣ ΑΠΟ ΧΡΟΝΟΣΕΙΡΕΣ: ΕΙΔΙΚΑ ΖΗΤΗΜΑΤΑ

Κατά τη διάρκεια της διαδικασίας εξόρυξης πληροφορίας από χρονοσειρές, εγείρονται μερικά ζητήματα, τα οποία μπορούν εν δυνάμει να επηρεάσουν δραματικά την αποτελεσματική ανάλυση των χρονοσειρών Τα σημαντικότερα ζητήματα είναι τα εξής:

- *Υποκειμενικότητα*. Το σημαντικότερο ζήτημα σε μία διαδικασία εξόρυξης πληροφορίας μέσω ανάλυσης χρονοσειρών είναι η υποκειμενικότητα της έννοιας της ομοιότητας. Έτσι, όπως θα δούμε και στη συνέχεια της παρούσας εργασίας είναι απαραίτητο να καθορίζεται ένα μέτρο ομοιότητας βάσει του οποίου να προσδιορίζεται με αντικειμενικό τρόπο κατά πόσο δυο χρονοσειρές μοιάζουν / διαφέρουν μεταξύ τους.
- *Θόρυβος*. Μία χρονοσειρά, από την φύση της, εμπεριέχει θόρυβο σε κάποιο βαθμό, ο οποίος μπορεί δημιουργείται είτε από τις διάφορες στις συνθήκες που επικρατούν τη στιγμή της μέτρησης είτε από σφάλματα στις μετρήσεις. Επειδή ο θόρυβος μπορεί να μας οδηγήσει σε μη αξιόπιστα αποτελέσματα, πρέπει να εφαρμόζονται στα δεδομένα μας τεχνικές προ-επεξεργασίας για την απομάκρυνση/περιορισμό του θορύβου πριν προχωρήσουμε σε περαιτέρω μελέτες και έρευνες.
- *Υψηλή Διαστατότητα.* Οι χρονοσειρές στην πλειοψηφία τους έχουν αρκετά μεγάλο μέγεθος, γεγονός το οποίο τις καθιστά αντικείμενα πολλών διαστάσεων. Για την αντιμετώπιση αυτού του ζητήματος έχουν προταθεί πολλές μέθοδοι μείωσης της διάστασης των χρονοσειρών και διαχείρισης αυτού του μεγάλου όγκου δεδομένων (indexing), ώστε να έχουμε μικρότερους χρόνους απόκρισης στις διάφορες εργασίες που πραγματοποιούνται πάνω στις χρονοσειρές.

- *Διαφορετικό μέγεθος χρονοσειρών*. Τα δεδομένα που μελετώνται κατά τη διάρκεια της διαδικασίας εξόρυξης γνώσης μέσω ανάλυσης χρονοσειρών έχουν συνήθως διαφορετικό μέγεθος, γεγονός που δυσκολεύει τη σύγκριση τους. Για τον λόγο αυτό πρέπει να εφαρμοστούν πάνω στα δεδομένα τεχνικές οι οποίες να τα καθιστούν συγκρίσιμα.

Όλα τα παραπάνω ζητήματα οδήγησαν τον χώρο της Εξόρυξης Πληροφορίας από Χρονοσειρές στην έρευνα για την αναζήτηση μεθόδων και τεχνικών που θα παρέχουν αποτελεσματικούς τρόπους αντιμετώπισής τους.

ΚΕΦΑΛΑΙΟ 2^{ο}

ΑΝΑΖΗΤΗΣΗ ΟΜΟΙΩΝ ΧΡΟΝΟΣΕΙΡΩΝ

2.1 ΕΙΣΑΓΩΓΗ

Στο κεφάλαιο αυτό πραγματοποιείται αρχικά μια εισαγωγή στη διαδικασία αναζήτησης όμοιων χρονοσειρών από μεγάλες βάσεις δεδομένων, μια διαδικασία στην οποία εστιάζεται το μεγαλύτερο μέρος των εφαρμογών της διαδικασίας Εξόρυξης Πληροφορίας από Χρονοσειρές. Στη συνέχεια, παρουσιάζονται οι κυριότερες τεχνικές προ-επεξεργασίας χρονοσειρών για την απομάκρυνση τυχόν στρεβλώσεων που δυσκολεύουν το έργο της διαδικασίας αναζήτησης όμοιων χρονοσειρών. Σκοπός επίσης του κεφαλαίου αυτού είναι η κατανόηση της σπουδαιότητας της επιτάχυνσης της διαδικασίας εξόρυξης γνώσης μέσω ανάλυσης χρονοσειρών. Επίσης, παρατίθενται οι κυριότερες μέθοδοι μείωσης των διαστάσεων μίας χρονοσειράς, ώστε να επιτευχθεί η επιτάχυνση αυτή και παρουσίαση της τεχνικής PAA. Κατόπιν γίνεται μια εισαγωγή στην χρησιμότητα της τεχνικής του κατώτερου φράγματος.

2.2 Η ΔΙΑΔΙΚΑΣΙΑ ΑΝΑΖΗΤΗΣΗΣ ΟΜΟΙΩΝ ΧΡΟΝΟΣΕΙΡΩΝ

Την έννοια της ομοιότητας τη συναντάμε πολύ συχνά στην καθημερινή μας ζωή. Την χρησιμοποιούμε σε πρακτικά προβλήματα, όπως για την κατασκευή όμοιων σχημάτων ή για το πόσο ένας άνθρωπος μοιάζει με κάποιον άλλο ή για το πόσο μια εικόνα είναι όμοια με μια άλλη.

Όλες οι ενέργειες-στόχοι της διαδικασίας εξόρυξης πληροφορίας από χρονοσειρές, οι οποίες περιγράφηκαν στο Κεφάλαιο 1, εμπεριέχουν την έννοια της ομοιότητας, αφού απαιτούν την αναζήτηση όμοιων προτύπων. Ένα βασικό πρόβλημα στην αναζήτηση αυτή είναι η υποκειμενικότητα με την οποία ορίζεται κάθε φορά η έννοια της ομοιότητας. Δυο χρονοσειρές μπορούν να θεωρηθούν παρόμοιες όταν παρουσιάζουν παρόμοια μορφή. Όμως, σε διαφορετικές εφαρμογές μπορεί να νοείται διαφορετικά το τι σημαίνει *παρόμοια μορφή*. Για

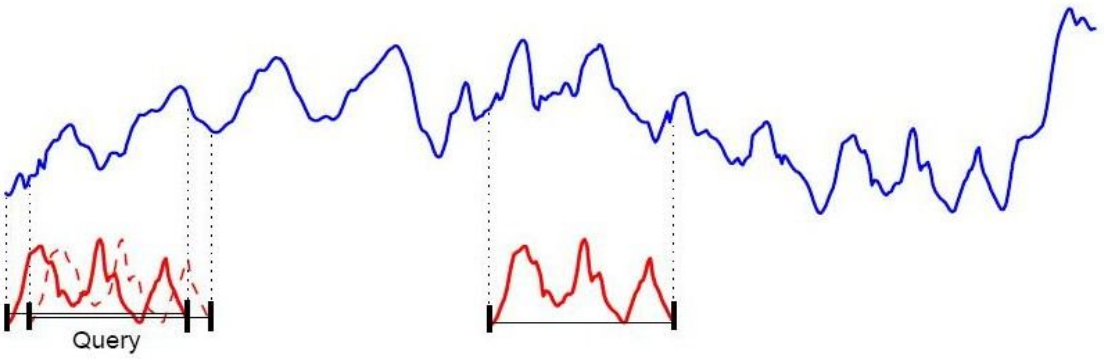
παράδειγμα, σε κάποιες εφαρμογές η τάξη μεγέθους των τιμών μπορεί να είναι σημαντική, ενώ σε άλλες εφαρμογές να θεωρείται σημαντική μόνο η μορφή της χρονοσειράς (ανεξάρτητα από την τάξη μεγέθους των τιμών). Για παράδειγμα ένας ασθενής επισκέπτεται τον γιατρό του για να κάνει κάποιο καρδιογράφημα και στη συνέχεια ο γιατρός συγκρίνει τα αποτελέσματα του ασθενή βάση των υπαρχόντων δεδομένων (ποιο καρδιογράφημα μοιάζει με ποιο) γεγονός που θα τον καθοδηγήσει σε πιο σωστή διάγνωση. Μόνο ο γιατρός αυτός μπορεί να ορίσει το πότε δύο καρδιογραφήματα θεωρούνται όμοια στην Ιατρική.

Επίσης, η παρουσία θορύβου ή διαφορετικού ρυθμού εξέλιξης του φαινομένου οδηγεί σε ασαφής αντιστοιχίες μεταξύ χρονοσειρών. Ως εκ τούτου, είναι αναγκαίο να καθοριστεί το κατάλληλο μέτρο ομοιότητας, αφού η έννοια της ομοιότητας συνεπάγεται ένα βαθμό υποκειμενικότητας που μπορεί να επηρεάσει το τελικό αποτέλεσμα.

Ακόμη, η εγγενής υψηλή διαστατότητα των χρονοσειρών επιβάλλει τον μετασχηματισμό τους σε χρονοσειρές χαμηλότερης διαστατότητας με σκοπό την αποδοτικότερη ανάλυσή τους.

Οι τεχνικές οι οποίες χρησιμοποιούνται στη διαδικασία ανάκτησης όμοιων χρονοσειρών είναι οι εξής (Vlachos, 2004) [32]:

- *Whole matching (πλήρης).* Στην τεχνική αυτή αναζητούνται οι χρονοσειρές που μοιάζουν με μία δεδομένη χρονοσειρά (query Q). Βασική προϋπόθεση αποτελεί όλες οι χρονοσειρές που εξετάζονται να έχουν το ίδιο μέγεθος με την query Q. Παράδειγμα είναι η ανάκτηση όμοιων χρονοσειρών με τη χρονοσειρά που αναπαριστά τα μηνιαία κέρδη μιας επιχείρησης ώστε να εντοπιστούν όλες οι επιχειρήσεις των οποίων τα κέρδη παρουσιάζουν την ίδια διαχρονική εξέλιξη.
- *Subsequence matching (τμηματική).* Σύμφωνα μ' αυτή την τεχνική αναζητούνται οι υπό-ακολουθίες των χρονοσειρών οι οποίες είναι όμοιες με τη χρονοσειρά query Q (Εικόνα 2.1). Εδώ δεν είναι ανάγκη όλες οι χρονοσειρές που εξετάζονται να έχουν το ίδιο μέγεθος απλά θα πρέπει η query Q να είναι μικρότερη σε μέγεθος από τις χρονοσειρές των οποίων αναζητούνται οι υπό-ακολουθίες. Ένα τέτοιο παράδειγμα είναι η ανάκτηση υπό-ακολουθιών στις DNA ακολουθίες ομοίων με την ακολουθία DNA (query Q), ενός ασθενούς με σοβαρή πάθηση προκειμένου ο γιατρός να καταλήξει σε σωστή διάγνωση.



Εικόνα 2.1 Παράδειγμα *Subsequence matching*

Πηγή: Vlachos M.(2004). "Similarity and Indexing in Multidimensional Spaces" California, September

Στη διαδικασία ανάκτησης όμοιων χρονοσειρών από μια βάση δεδομένων διακρίνουμε δυο διαδικασίες αναζήτησης (Vlachos, 2004) [32]:

- *Range Query*, όπου ζητείται να ανακτηθούν όλες οι χρονοσειρές οι οποίες διαφέρουν από την query Q κατά ένα ποσοστό p. Το p θεωρείται η παράμετρος βάση της οποίας θεωρούνται κάποιες χρονοσειρές όμοιες ή όχι μεταξύ τους. Η τιμή της παραμέτρου p προσδιορίζεται από το χρήστη, γεγονός το οποίο δεν είναι πολύ εύκολο γ' αυτό δεν χρησιμοποιείται τόσο συχνά η μέθοδος αυτή.
- *k-Nearest Neighbor (kNN) Search*, όπου ζητείται από μια βάση δεδομένων να ανακτηθούν οι k περισσότερο όμοιες χρονοσειρές με τη χρονοσειρά query Q. Η τιμή του k μπορεί να είναι ένας οποιοσδήποτε θετικός ακέραιος αριθμός, αν το k είναι ίσο με 1 τότε ανακτάται μόνο η χρονοσειρά, η οποία είναι περισσότερο όμοια με την query Q.

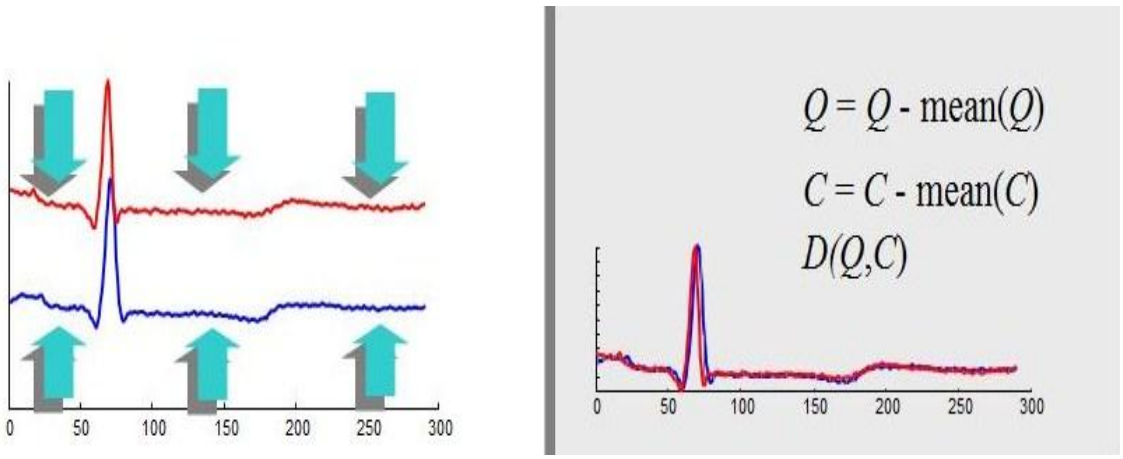
2.3 ΤΕΧΝΙΚΕΣ ΠΡΟ-ΕΠΕΞΕΡΓΑΣΙΑΣ ΧΡΟΝΟΣΕΙΡΩΝ

Η διαδικασία εξόρυξης πληροφορίας μέσω ανάλυσης χρονοσειρών δεν είναι πάντα μια εύκολη διαδικασία. Ανάλογα με το πως ορίζεται η έννοια της ομοιότητας σε μία εφαρμογή, θα πρέπει να εφαρμοστούν (ή να μην εφαρμοστούν)

διάφορες τεχνικές προ-επεξεργασίας των χρονοσειρών. Για παράδειγμα, η παρουσία *θορύβου* στα δεδομένα, συνήθως, είναι ανεπιθύμητη. Ο θόρυβος αναφέρεται σε σφάλματα στην καταγραφή τιμών, τιμές δηλαδή που δεν ανταποκρίνονται στις πραγματικές λόγω δυσμενών περιβαλλοντικών συνθηκών ή λόγω σφάλματος της συσκευής καταγραφής. Ο θόρυβος δημιουργεί προβλήματα καθώς μπορεί να οδηγήσει σε εσφαλμένη μέτρηση της ομοιότητας μεταξύ δύο χρονοσειρών. Οι Keogh και Pazzani (1999) [21] παραθέτουν τέσσερις βασικές περιπτώσεις *στρεβλώσεων* που ενδέχεται να υπάρχουν στα δεδομένα και τεχνικές αντιμετώπισής τους. Συγκεκριμένα, οι *στρεβλώσεις* αυτές είναι οι εξής: Offset Translation, Amplitude Scaling, Linear Trend, Noise.

2.3.1 OFFSET TRANSLATION

Στη διαδικασία ανάκτησης όμοιων χρονοσειρών από μια βάση δεδομένων είναι αρκετά συχνό φαινόμενο να παρουσιάζονται χρονοσειρές, οι οποίες να είναι όμοιες μεταξύ τους, οι τιμές τους όμως να είναι διαφορετικής τάξης μεγέθους (Offset Translation). Στην γραφική αναπαράσταση, οι χρονοσειρές αυτές παρουσιάζονται να είναι παράλληλα μετατοπισμένες ως προς τον άξονα στον yy΄. Μία τεχνική προ-επεξεργασίας των δεδομένων για την αντιμετώπιση αυτού του ζητήματος είναι η εξής: από κάθε μία τιμή της χρονοσειράς να αφαιρεθεί η μέση τιμή των τιμών της. Αυτό έχει ως αποτέλεσμα, η μετασχηματισμένη χρονοσειρά να έχει μέση τιμή ίση με το μηδέν (0) (Keogh, 1997) [13].



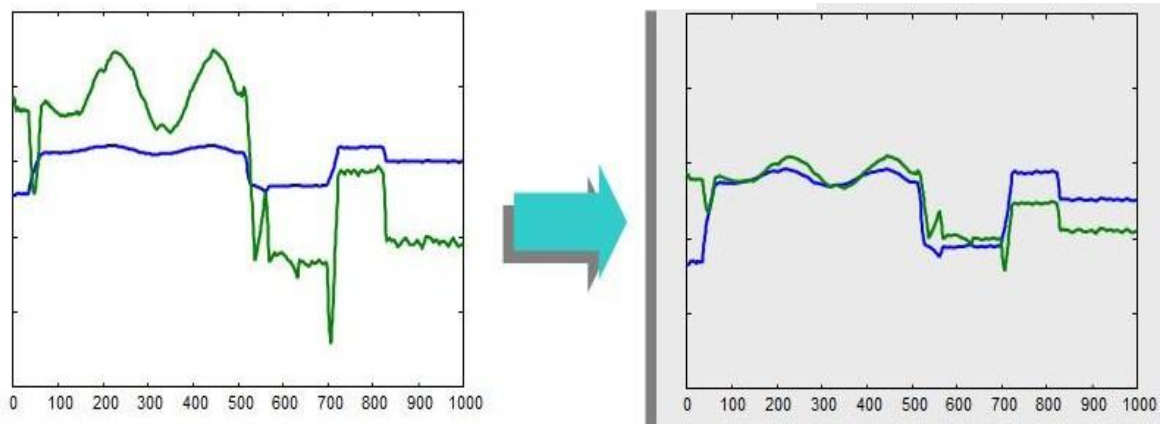
Εικόνα 2.2 Τεχνική προ-επεξεργασίας στην περίπτωση Offset Translation

Πηγή : Keogh, E. J. (2001). "A Tutorial on Indexing and Mining Time Series Data", In Proceedings of IEEE International Conference on Data Mining, San Jose, November 29.

2.3.2 AMPLITUDE SCALING

Μία δεύτερη περίπτωση είναι η εξής: δύο χρονοσειρές, οι οποίες είναι όμοιες μεταξύ τους, οι τιμές τους να βρίσκονται σε διαφορετική κλίμακα (Amplitude Scaling). Αυτό στην γραφική αναπαράσταση των χρονοσειρών παρουσιάζεται σαν κάποιες χρονοσειρές να είναι συμπιεσμένες στον yy΄ και άλλες αρκετά απλωμένες στον yy΄. Στην περίπτωση αυτή, η τεχνική προ-επεξεργασίας που εφαρμόζεται είναι ο μετασχηματισμός των τιμών της χρονοσειράς που επιτυγχάνεται διαιρώντας κάθε μία τιμή με την αντίστοιχη τυπική απόκλιση των τιμών της χρονοσειράς. Αν εφαρμοστούν και οι δυο προηγούμενες τεχνικές προ-επεξεργασίας, τότε η μετασχηματισμένη χρονοσειρά θα έχει μέση τιμή ίση με το μηδέν (0) και τυπική απόκλιση ίση με τη μονάδα (1) (Εικόνα 2.3).

$$Q = (Q - \text{mean}(Q)) / \text{std}(Q)$$
$$C = (C - \text{mean}(C)) / \text{std}(C) \qquad (2.1)$$
$$D(Q,C)$$



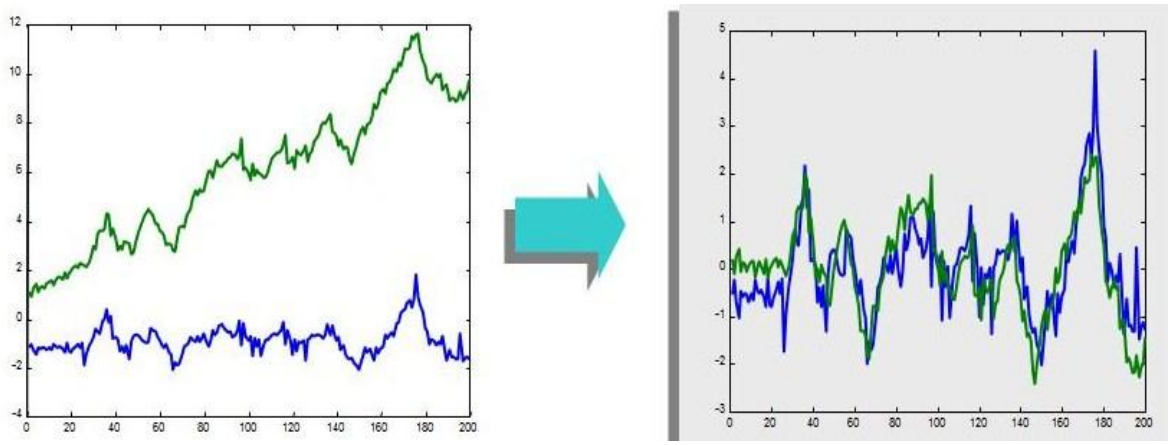
Εικόνα 2.3 Τεχνική προ-επεξεργασίας στην περίπτωση Amplitude Scaling

Πηγή : Keogh, E. J. (2001). "A Tutorial on Indexing and Mining Time Series Data", In Proceedings of IEEE International Conference on Data Mining, San Jose, November 29.

2.3.3 LINEAR TREND

Αυτή η τεχνική χρησιμοποιείται ιδιαίτερα σε περιπτώσεις ανάλυσης χρονοσειρών όπου οι εξωγενείς παράγοντες παίζουν αρκετά σημαντικό ρόλο. Για παράδειγμα στην περίπτωση σύγκρισης χρονοσειρών που αναπαριστούν τις

πωλήσεις εταιρείας παιδικών ενδυμάτων σε διάφορες χώρες ίδιας οικονομικής κατάστασης και με ίδια κουλτούρα βάσει της Ευκλείδειας Απόστασης, είναι πιθανόν να προκύψει ότι οι χρονοσειρές δεν είναι όμοιες μεταξύ τους. Αυτό μπορεί να οφείλεται στο ότι σε κάποιες χώρες ο πληθυσμός αυξάνεται με σταθερό ρυθμό ενώ σε κάποιες άλλες όχι. Η μοντελοποίηση της τάσης των προς ανάλυση χρονοσειρών πραγματοποιείται με τον καθορισμό εκείνης της ευθείας γραμμής η οποία αναπαριστά με τον καλύτερο δυνατό τρόπο την τάση της εκάστοτε χρονοσειράς που εξετάζεται (Keogh, 1997)[13]. Μέσω της εξίσωσης που προκύπτει από την παραπάνω ευθεία, υπολογίζεται μια προσεγγιστική τιμή ανά χρονική στιγμή η οποία ύστερα αφαιρείται από την αντίστοιχη πραγματική τιμή και έτσι προκύπτει η νέα επεξεργασμένη τιμή (Εικόνα 2.4).



Εικόνα 2.4 Τεχνική προ-επεξεργασίας στην περίπτωση Linear Trend.

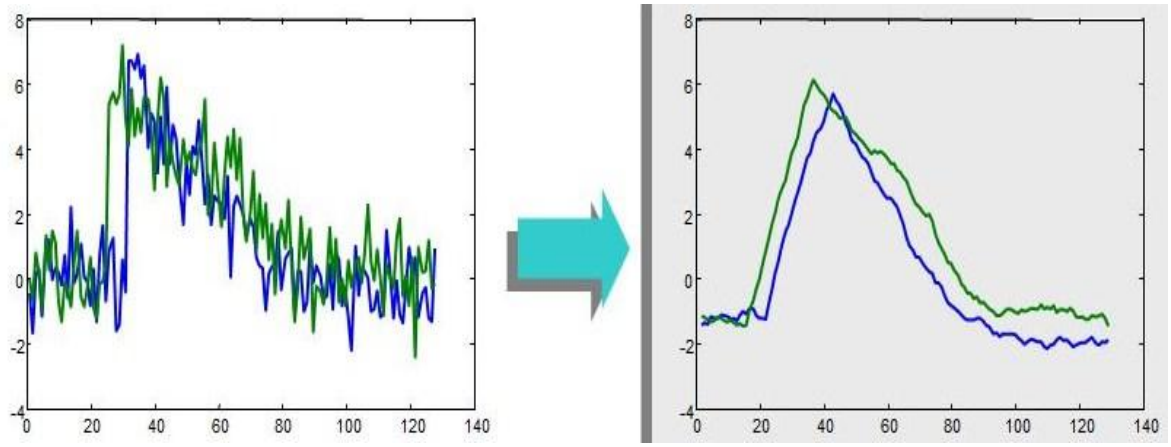
Πηγή : Keogh, E. J. (2001). “A Tutorial on Indexing and Mining Time Series Data”, *In Proceedings of IEEE International Conference on Data Mining*, San Jose, November 29.

2.3.4 Noise

Για την εξάλειψη των ασυνεχειών στις γραφικές παραστάσεις των χρονοσειρών οι οποίες προκύπτουν από την παρουσία θορύβου και δυσχεραίνουν τη διαδικασία ανάλυσης χρονοσειρών χρησιμοποιείται ευρέως η μέθοδος Moving Average Smoothing (Keogh, 1997) [13]. Σύμφωνα μ’ αυτή την τεχνική κάθε μέτρηση μιας χρονοσειράς αντικαθιστάται με τον μέσο όρο των n

προηγούμενων μετρήσεων, των n επόμενων μετρήσεων και της συγκεκριμένης τιμής. Συνεπώς οι χρονοσειρές που προκύπτουν έχουν μικρότερο μέγεθος.

$$Q = \text{smooth}(Q)$$
$$C = \text{smooth}(C)\ ,\ D(Q, C) \qquad (2.2)$$



Εικόνα 2.5 Τεχνική προ-επεξεργασίας στην περίπτωση Moving Average Smoothing
Πηγή : Keogh, E. J. (2001). "A Tutorial on Indexing and Mining Time Series Data", In Proceedings of IEEE International Conference on Data Mining, San Jose, November 29.

2.4 ΑΝΑΠΑΡΑΣΤΑΣΗ ΧΡΟΝΟΣΕΙΡΩΝ

Ο χρονικός χαρακτήρας των χρονοσειρών αποτελεί ένα πολύ σημαντικό χαρακτηριστικό τους το οποίο θα πρέπει να λαμβάνεται υπόψη κατά την εφαρμογή των διάφορων μεθόδων της διαδικασίας της εξόρυξης πληροφορίας μέσω ανάλυσης χρονοσειρών.

Σε κάθε περίπτωση, η αναζήτηση όμοιων χρονοσειρών προϋποθέτει την σύγκριση μίας χρονοσειράς με κάθε μία από τις χρονοσειρές μιας βάσης δεδομένων, η οποία θα πραγματοποιείται με τον υπολογισμό ενός μέτρου ομοιότητας. Στα πλαίσια της Εξόρυξης Πληροφορίας, μία τέτοια διαδικασία ενδέχεται να είναι έως και απαγορευτική σε σχέση με τον χρόνο που απαιτείται όταν έχουμε να κάνουμε με τεράστιες βάσεις δεδομένων. Μια χρονοσειρά μήκους n αποτελεί ένα αντικείμενο αντίστοιχου πλήθους διαστάσεων. Είναι φανερό ότι όσο το n παίρνει μεγαλύτερες τιμές τόσο η εφαρμογή τεχνικών εξόρυξης γνώσης τείνει να υλοποιείται λιγότερο αποδοτικά. Συνεπώς, είναι αναγκαία η αναπαράσταση της

χρονοσειράς σε μια μορφή η οποία θα ορίζεται από λιγότερες διαστάσεις και θα συμπιέζει τα αρχικά δεδομένα σε μικρότερο όγκο. Στη συνέχεια θα πρέπει πάνω σε τέτοιου είδους μορφές να εφαρμόζονται τεχνικές εξόρυξης γνώσης , δηλαδή σε μετασχηματισμένα δεδομένα. Η εφαρμογή μεθόδων εξόρυξης γνώσης στα μετασχηματισμένα δεδομένα θα πρέπει να εξασφαλίζει ότι θα εντοπίζει όλα τα «ενδιαφέροντα» πρότυπα, ενώ το πλήθος των ψευδώς αναγνωρισμένων πρότυπων ως «ενδιαφέροντα» θα πρέπει να ελαχιστοποιείται (Faloutsos et al., 1994) [8].

2.4.1 Η ΑΝΑΓΚΑΙΟΤΗΤΑ ΑΝΑΠΑΡΑΣΤΑΣΗΣ ΧΡΟΝΟΣΕΙΡΩΝ

Η διαδικασία ανάκτησης χρονοσειρών ή υπό-ακολουθιών των χρονοσειρών όμοιων με μια χρονοσειρά query-Q από μια βάση δεδομένων μεγάλου μεγέθους, εκτός από τη χρήση του κατάλληλου μέτρου ομοιότητας προϋποθέτει και την εφαρμογή αποδοτικών μεθόδων διαχείρισης των χρονοσειρών, οι οποίες θα συμβάλλουν στην ανάκτηση των χρονοσειρών σε πραγματικό χρόνο (Παπανίκου, 2008) [2]. Ο μεγάλος όγκος και το μέγεθος των διαθέσιμων χρονοσειρών καθιστά τη σειριακή αναζήτηση ακατάλληλη για την ικανοποίηση ερωτημάτων ανάκτησης σε πραγματικό χρόνο. Οι ερευνητές, προκειμένου να επιταχυνθεί η διαδικασία ανάκτησης όμοιων χρονοσειρών, πρότειναν την εφαρμογή κατάλληλων μεθόδων αποδοτικής αναπαράστασης και διαχείρισης αυτού του μεγάλου όγκου δεδομένων (ευρετηριοποίηση/indexing). Για να είναι αποτελεσματική μια τέτοια αναπαράσταση θα πρέπει να εγγυάται ότι: (α) θα διατηρεί τα σημαντικότερα χαρακτηριστικά / πληροφορίες της αρχικής χρονοσειράς και (β) θα συμπιέζει τα αρχικά δεδομένα σε ικανοποιητικό βαθμό (Agrawal, 1993) [3]. Η αποτελεσματικότητα των μεθόδων αναπαράστασης χρονοσειρών κρίνεται από τα εξής χαρακτηριστικά (Faloutsos et al) [8] :

- *Ταχύτητα*. Η ανάκτηση όμοιων χρονοσειρών από μια βάση δεδομένων θα πρέπει να πραγματοποιείται σε αρκετά λιγότερο χρόνο σε σχέση με τη σειριακή αναζήτηση.
- *Αξιοπιστία*. Τα αποτελέσματα των μεθόδων για την ανάκτηση όμοιων χρονοσειρών θα πρέπει να είναι αξιόπιστα. Θα πρέπει να επιστρέφονται

όλες οι όμοιες χρονοσειρές χωρίς παραλείψεις, φαινόμενο γνωστό και ως false dismissals.

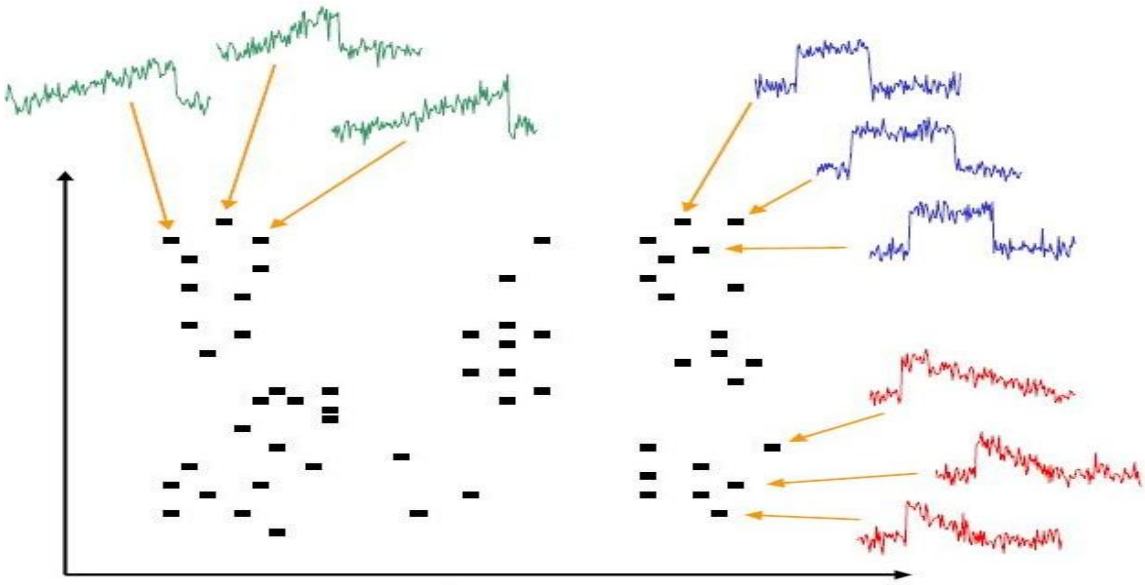
- *Δυναμικότητα*. Οι μέθοδοι ανάκτησης θα πρέπει να υποστηρίζουν τη δυναμικότητα που χαρακτηρίζει τις βάσεις δεδομένων, δηλαδή πρέπει να λαμβάνονται υπόψη διαγραφές ήδη υπαρχόντων χρονοσειρών, προσθήκες νέων και ενημερώσεις αυτών (update).
- *Ελάχιστη χρήση των πόρων του συστήματος*. Θα πρέπει να απασχολούνται όσοι το δυνατόν λιγότεροι πόροι του συστήματος από τις μεθόδους ανάκτησης.

Για την αποδοτική διαχείριση του μεγάλου όγκου δεδομένων και την επιτάχυνση της διαδικασίας ανάκτησης όμοιων χρονοσειρών οι ερευνητές Faloutsos και Lin (1995) [7] πρότειναν την αναπαράσταση μιας χρονοσειράς μεγέθους n, ως ένα σημείο στον N-διάστατο χώρο, όπου N <<n (Εικόνα 2.6) και την ευρετηριοποίηση (indexing) της με την εφαρμογή των R-Trees (Εικόνα 2.7). Η προσέγγιση αυτή έχει δύο βασικές προϋποθέσεις για να είναι αποτελεσματική:

- *Παραλείψεις όμοιων χρονοσειρών (false dismissals)*. Όπως αναφέρθηκε και προηγουμένως, οι μέθοδοι αναπαράστασης θα πρέπει να επιστρέφουν όλες τις όμοιες χρονοσειρές με μια χρονοσειρά query-Q χωρίς παραλείψεις. Συνεπώς, η επιλογή της μεθόδου αναπαράστασης θα πρέπει να εξασφαλίζει ότι το πρόβλημα αυτό δεν θα εμφανιστεί.
- «*Η κατάρα του πολυδιάστατου χώρου*» (*dimensionality curse*). Ο μετασχηματισμός / απεικόνιση πολύ μεγάλων χρονοσειρών μεγέθους n στον N-διάστατο χώρο και η ευρετηριοποίηση τους προϋποθέτει ότι N <<n. Όσο το N αυξάνεται τόσο η ευρετηριοποίηση τείνει να είναι ισοδύναμη (σε ταχύτητα) με την σειριακή αναζήτηση. Έχει αποδειχθεί ότι όσο αυξάνονται οι διαστάσεις τόσο αυξάνεται και ο χρόνος ανάκτησης όμοιων χρονοσειρών και μάλιστα με εκθετικό ρυθμό.

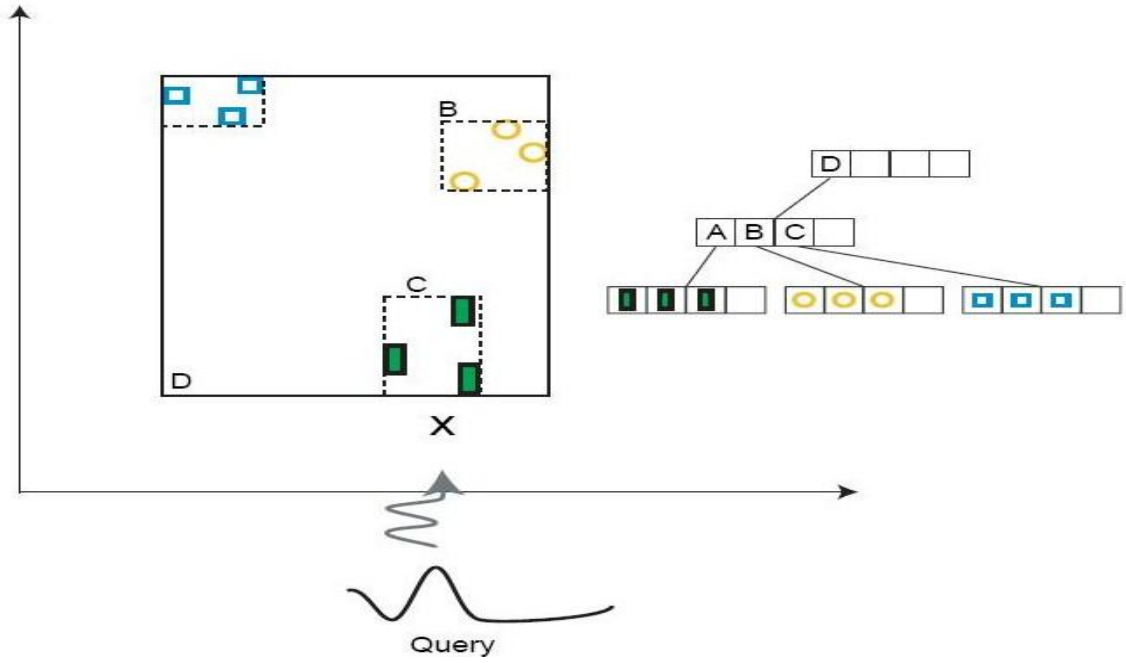
Οι ερευνητές ερχόμενοι αντιμέτωποι με τα παραπάνω εμπόδια οδηγήθηκαν στην επινόηση και εφαρμογή κατάλληλων μεθόδων οι οποίες να συμβάλλουν ουσιαστικά στη μείωση της πολυπλοκότητας των αλγορίθμων και κατά συνέπεια στην ταχύτερη εκτέλεση της διαδικασίας ανάκτησης. Αυτό επιτυγχάνεται με τη μείωση των διαστάσεων των χρονοσειρών έτσι ώστε να καταλαμβάνουν λιγότερο χώρο στη μνήμη και να δύναται να ευρετηριοποιηθούν. Η μείωση της διαστατότητας των χρονοσειρών αποδείχτηκε παράλληλα πως είναι ιδιαίτερα

αποδοτική αφού τα αποτελέσματα που προέκυπταν ύστερα από την εφαρμογή μεθόδων μείωσης της διάστασης των αρχικών δεδομένων ήταν στον ίδιο βαθμό ή και περισσότερο αξιόπιστα από τα αποτελέσματα της περίπτωσης ανάλυσης ακατέργαστων χρονοσειρών. Οι διάφορες τεχνικές μείωσης των διαστάσεων των χρονοσειρών παίζουν σημαντικό ρόλο στην ταχύτερη ανάκτηση όμοιων χρονοσειρών διότι προκειμένου να υπολογιστεί η απόσταση μιας χρονοσειράς από μια άλλη εξετάζονται πλέον πολύ λιγότερα σημεία και αυτό έχει ως αποτέλεσμα ο χρόνος υπολογισμού να είναι πολύ μικρότερος σε σχέση με το χρόνο που θα χρειαζόταν για την επεξεργασία ακατέργαστων δεδομένων.



Εικόνα 2.6 Αναπαράσταση χρονοσειρών ως σημεία στον Ν-διάστατο χώρο

Πηγή : Keogh, E. J. (2002). "Exact Indexing Of Dynamic Time Warping", In Proceedings of IEEE International Conference on Data Mining



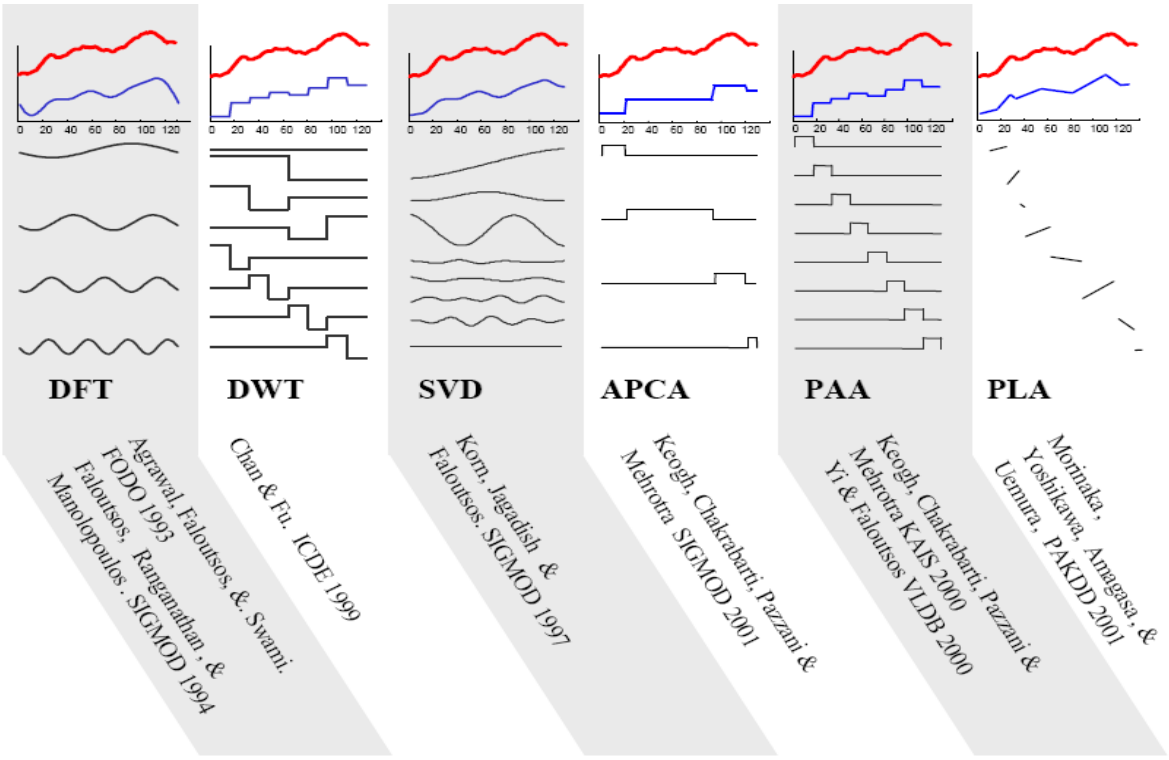
Εικόνα 2.7 Δημιουργία R-Trees για την ανάκτηση όμοιων χρονοσειρών

Πηγή: Vlachos M.(2004). "Similarity and Indexing in Multidimensional Spaces" California, September

2.4.2 ΤΕΧΝΙΚΕΣ ΑΝΑΠΑΡΑΣΤΑΣΗΣ ΧΡΟΝΟΣΕΙΡΩΝ

Το γεγονός ότι η μείωση των διαστάσεων των χρονοσειρών δεν επηρεάζει με αρνητικό τρόπο την αξιοπιστία των αποτελεσμάτων οφείλεται στο ότι οι διαδοχικές τιμές μιας χρονοσειράς δεν είναι απόλυτα ανεξάρτητες μεταξύ τους αλλά υψηλά συσχετιζόμενες (Παπανίκου, 2008)[2]. Κατά καιρούς παρουσιάστηκαν διάφορες μέθοδοι μείωσης των διαστάσεων των χρονοσειρών (Εικόνα 2.8). Οι πιο χαρακτηριστικές είναι οι μέθοδοι *Discrete Fourier Transform-DTF* (Agrawal et al.,1993, Keogh et al., 2000) [3,17,22], *Discrete Wavelet Transform-DWT* (Chan et Fu, 1999, Keogh et al., 2000) [6,17,22], *Singular Value Decomposition-SVD* (Keogh et al., 2000) [17,22], *Piecewise Aggregate Approximation-PAA* (Keogh & Smyth, 1997, Keogh et al., 2000) [23,17,22], *Piecewise Constant Approximation-PCA* (Keogh and Pazzani, 2000) [22], *Piecewise Linear Approximation-PLA* (Keogh & Pazzani, 1998, Keogh et al., 2001) [20,18], και η πιο πρόσφατη *Symbolic Aggregate Approximation-SAX* (Lin et al., 2002 & 2003) [24,25] η οποία αποτελεί την πιο αποτελεσματική μέθοδο αναπαράστασης καθώς επιτυγχάνει τη μείωση των διαστάσεων των χρονοσειρών μέσω της αντικατάστασης των

συνεχών τιμών με διακριτούς χαρακτήρες. Παρακάτω παρουσιάζεται η μέθοδος Piecewise Aggregate Approximation (PAA) πάνω στην οποία στηρίζεται και η SAX.



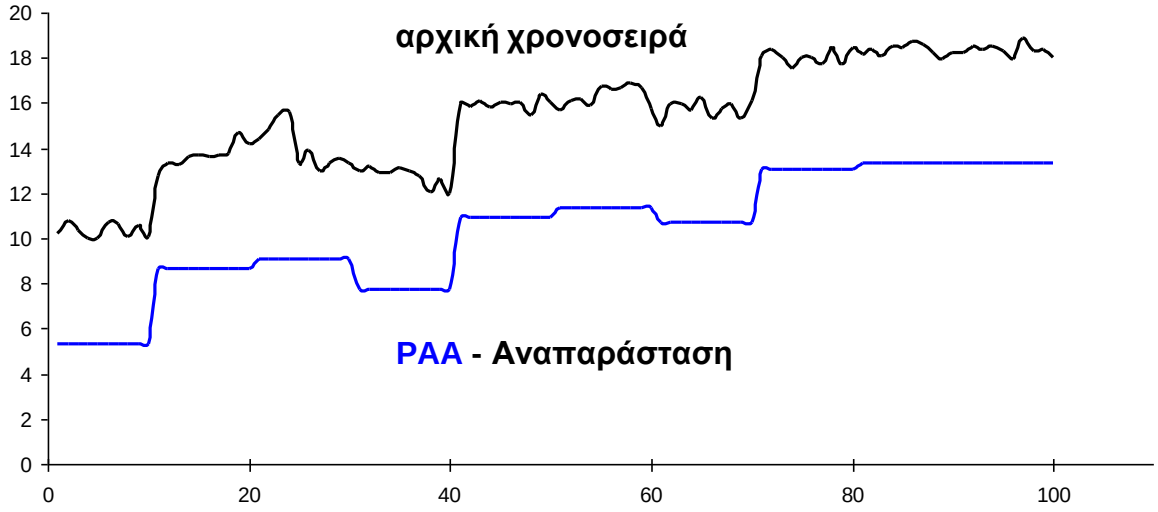
Εικόνα 2.8 Κυριότερες μέθοδοι μείωσης διαστάσεων χρονοσειρών

2.4.3 Η ΜΕΘΟΔΟΣ PIECEWISE AGGREGATE APPROXIMATION

Η μέθοδος PAA είναι μια πολύ απλή τεχνική μείωσης της διαστατότητας των χρονοσειρών η οποία προτάθηκε από τους Keogh et al 2001 [18] and Yi and Faloutsos 2000 [30]. Είναι γρήγορη στην εφαρμογή της, προσφέρει πολύ ικανοποιητικά αποτελέσματα και μπορεί να εφαρμοστεί σε χρονοσειρές διαφορετικού μεγέθους. Παράλληλα υποστηρίζει την εφαρμογή οποιασδήποτε τεχνικής απόστασης Minkowski. Εφαρμόζοντας την τεχνική PAA μια χρονοσειρά μήκους n χωρίζεται σε k συνεχόμενα ίσα τμήματα και υπολογίζεται η μέση τιμή για το κάθε ένα απ' αυτά, έτσι έχουμε μια νέα αναπαράσταση των αρχικών δεδομένων. Έστω μια χρονοσειρά $X = x_1, x_2, \cdots, x_n$ η οποία μπορεί να αναπαρασταθεί ως $\tilde{X} = (\bar{x}_1, s_1), (\bar{x}_2, s_2), \cdots, (\bar{x}_k, s_k)$, όπου :

$$\bar{x}_i = \sum_{j=(i-1)\frac{n}{k}+1}^{i\frac{n}{k}} x_j \Big/ \left(\frac{n}{k}\right)$$

Όπου n/k είναι ο αριθμός των ίσων τμημάτων ο οποίος θα πρέπει να είναι ακέραιος και να ισχύει k<<n. Αν k=n τότε η χρονοσειρά παραμένει όπως έχει, αν k=1 τότε η χρονοσειρά αντιστοιχίζεται με τη μέση τιμή. Θα πρέπει να δίνεται μεγάλη προσοχή στο ποιο είναι το κατάλληλο μέγεθος k στο οποίο θα διασπαστεί η χρονοσειρά, το αρχικό μέγεθος n απαιτείται να είναι ακέραιο πολλαπλάσιο του τελικού μεγέθους k.



Εικόνα 2.9 Τεχνική μείωσης διαστατικότητας χρονοσειρών PAA

2.4.4 Η ΤΕΧΝΙΚΗ ΤΟΥ ΚΑΤΩΤΕΡΟΥ ΦΡΑΓΜΑΤΟΣ

Στη διαδικασία ανάκτησης όμοιων χρονοσειρών από μεγάλες βάσεις δεδομένων αποτελεί πολύ σημαντικό θέμα τόσο η αξιοπιστία όσο και η ταχύτητα των αποτελεσμάτων. Για την επίτευξη αυτών των στόχων προτείνονται συνεχώς νέες αποδοτικότερες τεχνικές για την ανάκτηση όμοιων χρονοσειρών.

Στο πλαίσιο της εξόρυξης πληροφορίας μέσω ανάλυσης χρονοσειρών είναι επιθυμητό το μέτρο απόστασης που χρησιμοποιείται να ικανοποιεί την τριγωνική ανισότητα. Η χρησιμότητα αυτή παρουσιάζεται στην συνέχεια με ένα παράδειγμα. Ας υποθέσουμε ότι έχουμε 3 χρονοσειρές X, Y, Z και το query-Q για το οποίο

θέλουμε να εντοπίσουμε την ομοιότερη χρονοσειρά από τις X, Y, Z. Έστω ότι οι Ευκλείδειες αποστάσεις μεταξύ των X, Y, Z είναι γνωστές (Πίνακας 2.1).

Πίνακας 2.1

d	*X*	*Y*	*Z*
X		20	30
Y			10
Z			

Η απόσταση μεταξύ Q και X υπολογίζεται ότι είναι d(Q,X) = 80 και έτσι η X μέχρι στιγμής είναι η πιο κοντινή (πιο όμοια) στην Q. Κατόπιν υπολογίζεται η απόσταση Q και Y, d(Q,Y) =100 και αφού είναι μεγαλύτερη σημαίνει ότι η X παραμένει η πιο κοντινή στην Q. Σύμφωνα με την τριγωνική ανισότητα έχουμε:

$$d(Q,Y) \le d(Q,Z) + d(Z,Y) \Rightarrow d(Q,Z) \ge d(Q,Y) - d(Z,Y) = 100 - 10 = 90$$

Όπως βλέπουμε από την παραπάνω ανισότητα, η απόσταση της χρονοσειράς Q και της Z αν και μας είναι άγνωστη δεν μπορεί να είναι μικρότερη από 90. Η μέχρι τώρα μικρότερη μας απόσταση ήταν της Q με τη X, δηλαδή 80. Έτσι οδηγούμαστε στο συμπέρασμα πως δεν χρειάζεται να ανακτήσουμε τη χρονοσειρά Z και να υπολογίσουμε την απόσταση της από την Q. Σε πολύ μεγάλες βάσεις δεδομένων το συμπέρασμα αυτό μπορεί να οδηγήσει σε ουσιαστική βελτίωση της υπολογιστικής απόδοσης.

Για την επιτάχυνση των υπολογισμών των μετρικών απόστασης μπορεί να χρησιμοποιηθεί μία ακόμη τεχνική, η οποία ονομάζεται *lower bounding* και κάνει παρόμοια δουλειά με το προηγούμενο παράδειγμα. Το *lower bounding* αναφέρεται σε μια μέθοδο που μειώνει το όριο του αρχικού μέτρου που χρησιμοποιείται (original measure) και έτσι ο υπολογισμός γίνεται πολύ πιο γρήγορα, το όριο αυτό θα πρέπει να βρίσκεται όσο πιο κοντά γίνεται στο αρχικό μέτρο.

$$\forall\, X, Y:\ lower\_bound\_function(X, Y) \le original\_distance\_measure(X, Y)$$

Σύμφωνα μ' αυτή την προσέγγιση η *lower bounding* μέθοδος υπολογίζεται για δυο χρονοσειρές. Αν η μεταξύ τους απόσταση είναι μεγαλύτερη από την καλύτερη/μικρότερη μέχρι στιγμής απόσταση τότε δεν είναι αναγκαίο να υπολογιστεί η αρχική.

Το lower bounding είναι μια τεχνική η οποία απομακρύνει όλες εκείνες τις χρονοσειρές που απέχουν κατά πολύ από την query-Q ώστε να υπολογίζεται τελικά η πραγματική απόσταση μόνο μεταξύ εκείνων των χρονοσειρών που βρίσκονται κοντά στη χρονοσειρά query-Q. Μ' αυτόν τον τρόπο επιτυγχάνεται σε μεγαλύτερο βαθμό τόσο η αξιοπιστία των αποτελεσμάτων όσο και η ταχύτητα των υπολογισμών. Η τεχνική αυτή προϋποθέτει ότι το *lower _ bound _ function* υπολογίζεται σημαντικά ταχύτερα από το *original _ distance _measure*.

ΚΕΦΑΛΑΙΟ 3ο

ΜΕΤΡΑ ΑΠΟΣΤΑΣΗΣ

3.1 ΕΙΣΑΓΩΓΗ

Στο παρόν κεφάλαιο γίνεται αρχικά μια εισαγωγή στα χαρακτηριστικά των μέτρων και των μετρικών απόστασης. Σκοπός αυτού του κεφαλαίου είναι η παρουσίαση και σύγκριση των κυριότερων μεθόδων υπολογισμού της απόστασης/ομοιότητας μεταξύ δυο χρονοσειρών, όπως είναι η μετρική της ευκλείδειας απόστασης και η τεχνική Dynamic Time Warping. Επίσης αποδεικνύεται η αναγκαιότητα χρήσης αποτελεσματικών μεθόδων διαχείρισης χρονοσειρών με σκοπό την ταχύτερη και πιο αξιόπιστη ανάκτηση όμοιων χρονοσειρών από μεγάλες βάσεις δεδομένων.

3.2 ΜΕΤΡΑ ΚΑΙ ΜΕΤΡΙΚΕΣ ΑΠΟΣΤΑΣΗΣ

Λόγω του υποκειμενικού τρόπου διαχείρισης δεδομένων, κατά τη διάρκεια των προσπαθειών των επιστημόνων που ασχολούνται με τη διαδικασία της εξόρυξης πληροφορίας από χρονοσειρές να ανακτήσουν όμοιες χρονοσειρές από μεγάλες βάσεις δεδομένων διαπιστώθηκε αρκετά συχνά ότι τα αποτελέσματα διέφεραν αρκετά μεταξύ τους.

Για να αντιμετωπιστεί το πρόβλημα της υποκειμενικότητας αποφασίστηκε να οριστεί ένας τρόπος αξιολόγησης της ομοιότητας μεταξύ χρονοσειρών ο οποίος θα στηρίζεται σε αντικειμενικά κριτήρια. Ο τρόπος αυτός είναι η χρήση ενός μέτρου ομοιότητας / απόστασης (distance measure) το οποίο μετράει την ομοιότητα / απόσταση μεταξύ δύο χρονοσειρών. Συνήθως, τα μέτρα ομοιότητας τα οποία προτείνονται αποτελούν μέτρα αποστάσεων μεταξύ δύο χρονοσειρών, δηλαδή, στην πραγματικότητα αποτελούν μέτρα ανομοιότητας.

Μια τεχνική απόστασης ονομάζεται μετρική όταν διαθέτει τις ακόλουθες ιδιότητες:

- D(X,Y)=D(Y,X), η απόσταση (διαφορά) μιας χρονοσειράς X από μια χρονοσειρά Y ακριβώς ίδια με την απόσταση της χρονοσειράς Y από τη χρονοσειρά X.
- IF X=Y <=>D(X,Y)=0, η απόσταση μεταξύ δυο χρονοσειρών X και Y είναι μηδέν αν και μόνο αν πρόκειται για την ίδια χρονοσειρά.
- D(X,X)=0, η απόσταση μιας χρονοσειράς από την ίδια τη χρονοσειρά είναι μηδενική.
- D(X,Y)>=0, η απόσταση μιας χρονοσειράς X από μια χρονοσειρά Y είναι μεγαλύτερη ή ίση του μηδενός.
- D(X,Y)<=D(X,Z) + D(Y,Z), ισχύει η τριγωνική ανισότητα σύμφωνα με την οποία η απόσταση μιας χρονοσειράς X από μια χρονοσειρά Y είναι μικρότερη ή ίση με το άθροισμα των αποστάσεων της χρονοσειράς X από μια τρίτη χρονοσειρά Z και της χρονοσειράς Y από τη χρονοσειρά Z.

Αν κάποια από τις παραπάνω ιδιότητες δεν ικανοποιείται τότε το μέτρο απόστασης θεωρείται ότι δεν αποτελεί μετρική.

3.3 Lp - ΝΟΡΜΕΣ

Στον τομέα της εξόρυξης γνώσης μέσω χρονοσειρών χρησιμοποιούνται ευρέως οι μετρικές Lp-Νόρμες ως μέτρο για να ελέγξουν την ομοιότητα μεταξύ δυο χρονοσειρών. Συγκεκριμένα έστω ότι έχουμε δυο χρονοσειρές:

$$X = x_1, x_2, \cdots, x_n \quad \text{και} \quad Y = y_1, y_2, \cdots, y_n$$

Ο γενικός τύπος για την κατηγορία Lp-Νόρμες είναι ο εξής (Yi and Faloutsos, 2000) [30]:

$$L_p(X, Y) = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{\frac{1}{p}}$$

όπου n το μέγεθος των προς ανάλυση χρονοσειρών.

- Εάν p=1, τότε η απόσταση είναι γνωστή και ως *Manhattan απόσταση (L_1 - Norm)* και τότε ο τύπος παίρνει την εξής μορφή:

$$L_1(X, Y) = \sum_{i=1}^{n} |x_i - y_i|$$

Πλεονέκτημα αυτής είναι το γεγονός ότι είναι απλή, υπολογίζεται εύκολα. Δεν ενδύκνειται όμως ως η καλύτερη επιλογή.

- Εάν p=2, τότε ορίζεται ως την *Ευκλείδεια Απόσταση (L_2 - Norm)* και ο τύπος παίρνει την εξής μορφή:

$$L_2(X,Y) = \left(\sum_{i=1}^{n} (x_i - y_i)^2 \right)^{\frac{1}{2}}$$

Οι περισσότερες μέθοδοι ανάκτησης όμοιων χρονοσειρών βασίζονται στην Ευκλείδεια απόσταση διότι είναι αρκετά αποτελεσματική και γρήγορη. Δεν είναι βέβαια αποτελεσματική σε όλες τις περιπτώσεις. Η Ευκλείδεια απόσταση μπορεί να υπολογιστεί εύκολα, απλά και γρήγορα. Υπολογίζει την απόσταση σημείο προς σημείο και συνεπώς προϋποθέτει ότι όλες οι χρονοσειρές που εξετάζονται έχουν το ίδιο μέγεθος, πράγμα που δεν συμβαίνει σε όλες τις εφαρμογές. Αυτό συνήθως οφείλεται στον διαφορετικό ρυθμό δειγματοληψίας ή στο διαφορετικό χρόνο καταμέτρησης των τιμών του χαρακτηριστικού που μελετάται κάθε φορά.

- Εάν το p = ∞, τότε έχουμε την *άπειρη Νόρμα (L_{inf} - Norm)* και ο τύπος παίρνει την εξής μορφή:

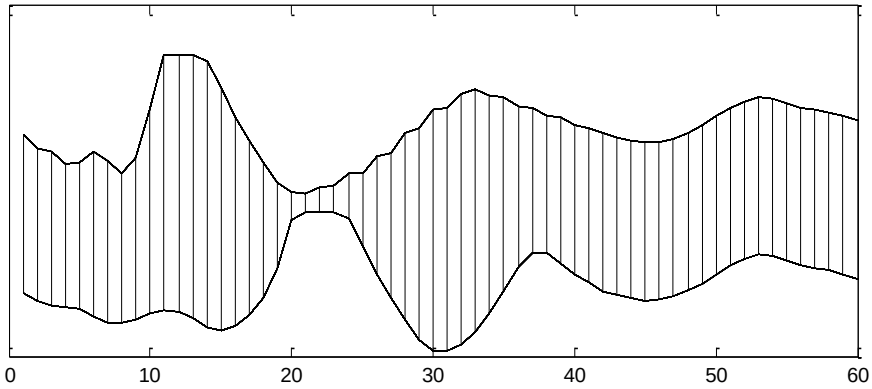
$$L_\infty(X,Y) = max|x_i - y_i|$$

Το μειονέκτημα των Lp – Norms, δηλαδή το ότι εφαρμόζονται μόνο στην περίπτωση χρονοσειρών ίδιου μεγέθους έρχεται να επιλύσει η μέθοδος *Dynamic Time Warping - DTW.*

3.4 Η ΤΕΧΝΙΚΗ ΑΠΟΣΤΑΣΗΣ DYNAMIC TIME WARPING

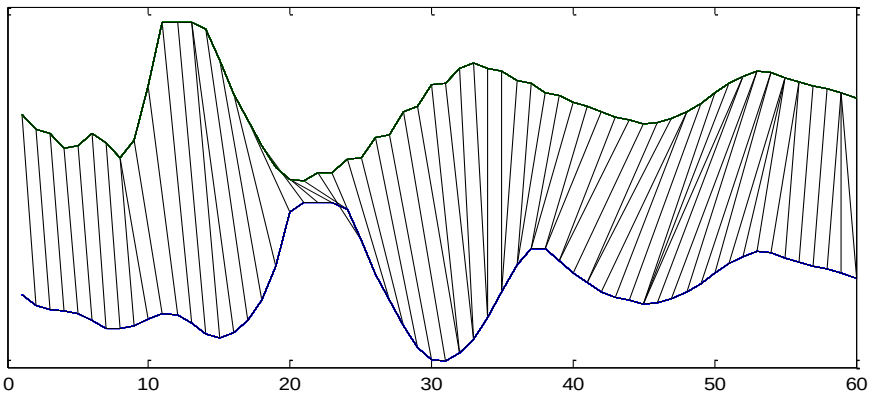
Όπως αναφέρθηκε και παραπάνω οι μετρικές Lp – Norms υπολογίζουν την απόσταση (ομοιότητα) των χρονοσειρών σημείο προς σημείο έτσι προϋποθέτουν όλες οι χρονοσειρές που εξετάζονται να έχουν το ίδιο μέγεθος, κάτι που δεν συμβαίνει πάντα (Εικόνα 3.1). Οι ερευνητές της διαδικασίας εξόρυξης γνώσης μέσω χρονοσειρών αποφάσισαν να εφαρμόσουν μια άλλη τεχνική η οποία να παρουσιάζει ευελιξία στο πεδίο του χρόνου, πράγμα το οποίο είναι απαραίτητο καθώς οι περισσότερες διεργασίες στον πραγματικό κόσμο εξελίσσονται με

διαφορετικούς ρυθμούς μέσα στο χρόνο, και έτσι να μπορεί να υπολόγιζει την απόσταση χρονοσειρών με διαφορετικό μέγεθος.



Εικόνα 3.1 Απόσταση σημείου προς σημείο (Ευκλείδεια Απόσταση) μεταξύ δυο χρονοσειρών

Η DTW (dynamic time warping) μπορεί να στηρίξει τα παραπάνω. Η ιδιαιτερότητα αυτής της τεχνικής είναι ότι σε σχέση με τις μετρικές απόστασης σημείο προς σημείο, όπου κάθε τιμή της μιας χρονοσειράς συγκρίνεται μόνο με την αντίστοιχη τιμή της άλλης χρονοσειράς, είναι ότι επιτρέπει τη σύγκριση ενός σημείου μίας χρονοσειράς με περισσότερα του ενός σημεία της άλλης χρονοσειράς (Εικόνα 3.2). Η τεχνική DTW χρησιμοποιείται σήμερα σε πολλές εφαρμογές.



Εικόνα 3.2 Αποστάσεις DTW μεταξύ δυο χρονοσειρών

Για παράδειγμα έστω ότι έχουμε κάποιες χρονοσειρές οι οποίες αναπαριστούν τη σεισμική δραστηριότητα μιας περιοχής σε διαφορετικά χρονικά διαστήματα. Οι επιστήμονες χρησιμοποιούν την DTW για να συγκρίνουν τις χρονοσειρές, η οποία λαμβάνει υπόψην της τις διαφορετικές χρονικές στιγμές που

εξελίσσονται τα φαινόμενα. Έτσι οδηγούνται σε σημαντικά συμπεράσματα χωρίς να εμποδίζονται από χρονικούς περιορισμούς.

Έστω ότι έχουμε δυο χρονοσειρές X και Y με μέγεθος m και n αντίστοιχα:

$$X = x_1 ,x_2 ,\cdots,x_m \qquad \text{και} \qquad Y = y_1 , y_2 ,\cdots, y_n$$

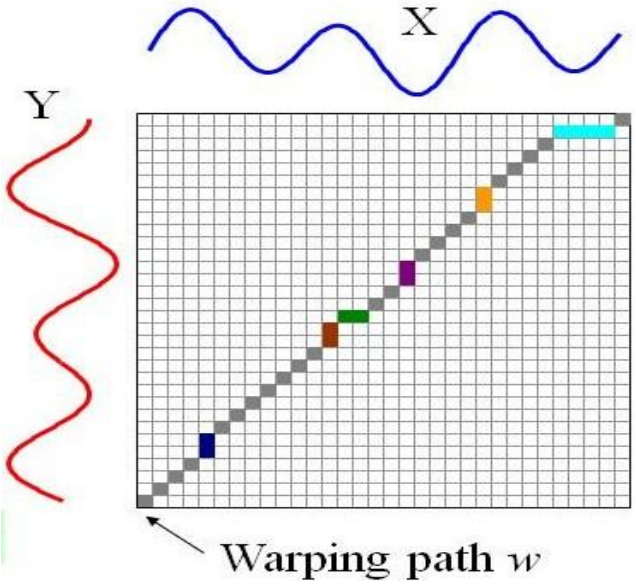
Το πρώτο βήμα της τεχνικής DTW είναι η δημιουργία ενός πίνακα διαστάσεων ίσων με τις διαστάσεις των χρονοσειρών, δηλαδή m * n, όπου αναπαριστάται κάθε πιθανός συνδυασμός των σημείων της X και της Y, των οποίων η απόσταση πρόκειται να υπολογιστεί (Yi et al, 1998) [31].

Στην συνέχεια, ξεκινώντας από το σημείο $\gamma_{1,1}$ του πίνακα υπολογίζεται η αθροιστική απόσταση μεταξύ του στοιχείου $\gamma_{1,1}$ και όλων των γειτονικών στοιχείων και επιλέγεται εκείνο το μικρότερο γειτονικό στοιχείο. Αυτό επιτυγχάνεται με την εφαρμογή της ακόλουθης αναδρομικής συνάρτησης :

$$\gamma(i, j) = d(i, j) + \min[\gamma(i -1, j -1),\gamma(i -1, j),\gamma(i, j -1)]$$

όπου i με τιμές από 1 έως m και όπου j με τιμές από 1 έως n. Η διαδικασία επαναλαμβάνεται με το στοιχείο που έχει ως αρχικό στοιχείο σύγκρισης και ολοκληρώνεται όταν το στοιχείο σύγκρισης ισοδυναμεί με το στοιχείο $\gamma_{m,n}$.

Αποτέλεσμα αυτής της διαδικασίας είναι η εύρεση της ελάχιστης διαδρομής από το στοιχείο $\gamma_{1,1}$ έως το στοιχείο $\gamma_{m,n}$, η οποία είναι ονομάζεται *warping path* (Εικόνα 3.3).



Πηγή : Keogh, E. J. (2001). "A Tutorial on Indexing and Mining Time Series Data", In Proceedings of IEEE International Conference on Data Mining, San Jose, November 29

Εικόνα 3.3 Υπολογισμός απόστασης μεταξύ δυο χρονοσειρών βάση της DTW

Η συνολική απόσταση μεταξύ των στοιχείων της ελάχιστης διαδρομής ισοδυναμεί με την απόσταση μεταξύ των χρονοσειρών Χ και Υ και προκύπτει από τον τύπο:

$$DTW(Q,C) = \min\left\{\sqrt{\sum_{k=1}^{K} w_k} \Big/ K\right\}$$

όπου w_k το k-οστό στοιχείο της ελάχιστης διαδρομής και Κ το συνολικό πλήθος των στοιχείων της ελάχιστης αυτής διαδρομής.

Στην συνέχεια παρατίθεται ένα παράδειγμα της όλης διαδικασίας. Ας υποθέσουμε ότι οι τιμές των χαρακτηριστικών των χρονοσειρών Χ, Υ είναι αυτές του Πίνακα 3.1:

Πίνακας 3.1 Οι Τιμές Δύο Χρονοσειρών

X	20	40	60	70	80	60	40	20
Y	10	30	40	50	40	30	10	

Χάριν ευκολίας, υπολογίζουμε την απόσταση μεταξύ δυο τιμών των χρονοσειρών με την απόλυτη τιμή της διαφοράς τους $d(x_i, y_j) = |x_i - y_j|$ και τα αποτελέσματα φαίνονται στον Πίνακα 3.2, δηλαδή στον πίνακα β:

Πίνακας 3.2 Απόλυτες αποστάσεις μεταξύ των σημείων των χρονοσειρών

Y	10	30	50	60	70	50	30	10
	10	10	30	40	50	30	10	10
	20	0	20	30	40	20	0	20
	30	10	10	20	30	10	10	30
	20	0	20	30	40	20	0	20
	10	10	30	40	50	30	10	10
	10	30	50	60	70	50	30	10
	X							

Έπειτα, με βάση την αναδρομική συνάρτηση που ακολουθεί υπολογίζουμε τον πίνακα των αθροιστικών αποστάσεων (Πίνακας 3.3), δηλαδή τον πίνακα γ, άρα έχουμε:

$\gamma(i, j) = d(i, j) + \min[\gamma(i-1, j-1), \gamma(i-1, j), \gamma(i, j-1)]$

γ(1,1) = β(1,1) + min[γ(0,1), γ(1,0), γ(0,0)] = 10

γ(1,2) = β(1,2) + min[γ(0,2), γ(1,1), γ(0,1)] = 10 + 10 = 20

γ(1,3) = β(1,3) + min[γ(0,2), γ(0,3), γ(1,2)] = 20 + 20 = 40

…….

γ(3,4) = β(3,4) + min[γ(2,3), γ(2,4), γ(3,3)] = 10 + 20 = 30

……

γ(8,7) = β(8,7) + min[γ(7,6), γ(7,7), γ(8,6)] = 10 + 100 = 110

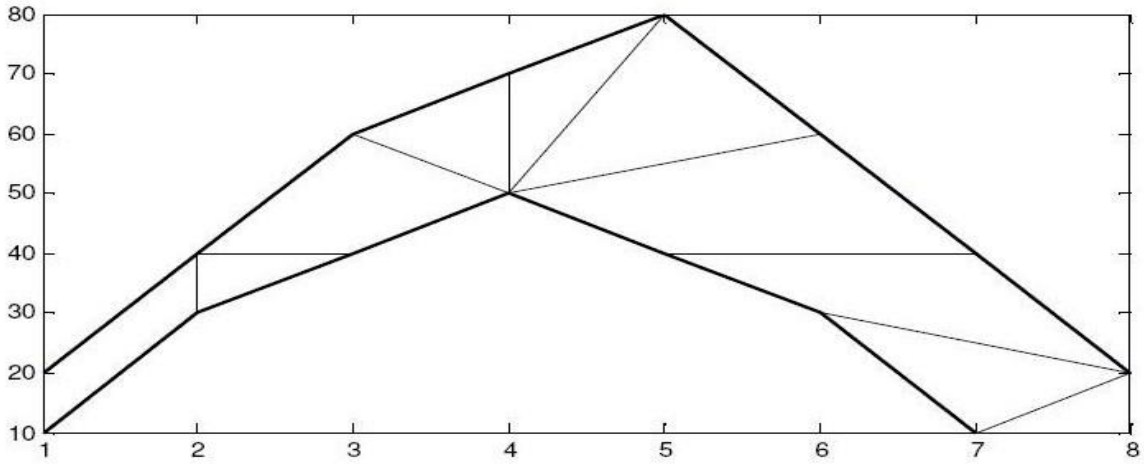
Πίνακας 3.3 Αθροιστικές αποστάσεις μεταξύ των χρονοσειρών

Y	110	70	90	120	160	160	130	110
	100	40	60	90	110	120	100	100
	90	30	50	60	90	100	90	110
	70	30	30	50	80	90	100	130
	40	20	40	70	110	130	130	150
	20	20	50	90	140	170	180	190
	10	40	90	150	220	270	300	310
	X							

Το τελευταίο άθροισμα στην πάνω δεξιά γωνία του πίνακα είναι και η συνολική *απόσταση* DTW μεταξύ των χρονοσειρών. Η ελάχιστη διαδρομή ή αλλιώς ***warping path*** υπολογίζεται πηγαίνοντας προς τα πίσω στον πίνακα γ και επιλέγοντας τη μικρότερη αθροιστική απόσταση από τις γειτονικές τιμές. Οι αποστάσεις μεταξύ των χρονοσειρών μετά την ευθυγράμμιση παρουσιάζονται στον Πίνακα 3.4, ενώ οι αντιστοιχίσεις των τιμών των δύο χρονοσειρών εμφανίζονται στο διάγραμμα 3.4:

Πίνακας 3.4 Αποστάσεις των χρονοσειρών μετά την ευθυγράμμιση

t_X	1	2	2	3	4	5	6	7	7	8	
X	20	40	40	60	70	80	60	40	40	20	
t_Y	1	2	3	4	4	4	4	5	6	7	
Y	10	30	40	50	50	50	50	40	30	10	
Αποστάσεις	10	10	0	10	20	30	10	0	10	10	110



Εικόνα 3.4 Γραφική αναπαράσταση της απόστασης DTW των χρονοσειρών

Τα σημεία του ***warping path*** θα πρέπει να υπακούουν τους τρεις ακόλουθους περιορισμούς.

- Στο warping path $W = w_1, w_2, \ldots w_p$ τα σημεία να είναι ταξινομημένα ως προς το χρόνο, δηλαδή για δυο διαδοχικά σημεία $w_k = (x_{ik} , y_{jk})$ και $w_{k-1} = (x_{ik-1} , y_{jk-1})$ να ισχύει :

$$i_k - i_{k-1} \geq 0 \quad \text{και} \quad j_k - j_{k-1} \geq 0$$

- Στο warping path $W = w1, w2, \ldots w_p$ τα επιτρεπόμενα βήματα περιορίζονται σε γειτονικά σημεία, δηλαδή για δυο διαδοχικά σημεία $w_k = (x_{ik} , y_{jk})$ και $w_{k-1} = (x_{ik-1} , y_{jk-1})$ να ισχύει :

$$i_k - i_{k-1} \geq 1 \quad \text{και} \quad j_k - j_{k-1} \geq 1$$

- Στο warping path $W = w_1, w_2, \ldots w_p$ το πρώτο και το τελευταίο σημείο πρέπει να είναι :

$$w_1 = (x_1 , y_1) \text{ και } w_p = (x_m , y_n)$$

Η τεχνική DTW στοχεύει στην εύρεση της καλύτερης δυνατής ευθυγράμμισης μεταξύ των δυο χρονοσειρών X και Y η οποία ελαχιστοποιεί την μεταξύ τους απόσταση. Η εύρεση της ελάχιστης απόστασης είναι το *warping path*. Ένα σημείο της χρονοσειράς X μπορεί να αντιστοιχηθεί με περισσότερα από ένα σημεία της χρονοσειράς Y (σύγκριση πολλά προς ένα), επιτρέπεται δηλαδή η στρέβλωση των χρονοσειρών (επέκταση ή συρρίκνωση κατά μήκος του άξονα χρόνου). Κάθε πιθανή στρέβλωση μεταξύ των χρονοσειρών είναι μια διαδρομή και

το ζητούμενο είναι η καλύτερη/ελάχιστη μεταξύ τους διαδρομή. Εδώ πρέπει να τονιστεί ότι η τεχνική DTW δεν είναι μετρική καθώς δεν ικανοποιεί την τριγωνική ανισότητα.

3.5 ΕΥΚΛΕΙΔΕΙΑ ΑΠΟΣΤΑΣΗ και DTW

Η τεχνική DTW αποδεικνύεται περισσότερο αξιόπιστη σε σχέση με την Ευκλείδεια απόσταση όσο αναφορά τα ποσοστά λάθους που σημειώνονται κατά τη διαδικασία εξόρυξης γνώσης, καθώς η DTW εμφανίζει αρκετά μικρότερα ποσοστά (Keogh, E. J. & Ratanamahatana, 2004) [16]. Επίσης η εφαρμογή της Ευκλείδειας απόστασης δεν παρουσιάζει μεγάλη ανθεκτικότητα στους θορύβους, δεν λαμβάνει υπόψην το διαφορετικό ρυθμό με τον οποίο εξελίσσονται δυο χρονοσειρές μέσα στο χρόνο και δεν συγκρίνει χρονοσειρές με διαφορετικό μέγεθος σε σχέση πάντα με την DTW για την οποία ισχύουν τα αντίθετα.

Από την άλλη μεριά όμως, η μέθοδος DTW μειονεκτεί σε εξίσου σημαντικά σημεία. Τα σημεία αυτά είναι ο χρόνος και το κόστος υπολογισμού καθώς πρόκειται για έναν αρκετα χρονοβόρο αλγόριθμο με μεγάλη πολυπλοκότητα $O(n^2)$, ο οποίος απαιτεί μεγάλο κόστος CPU.

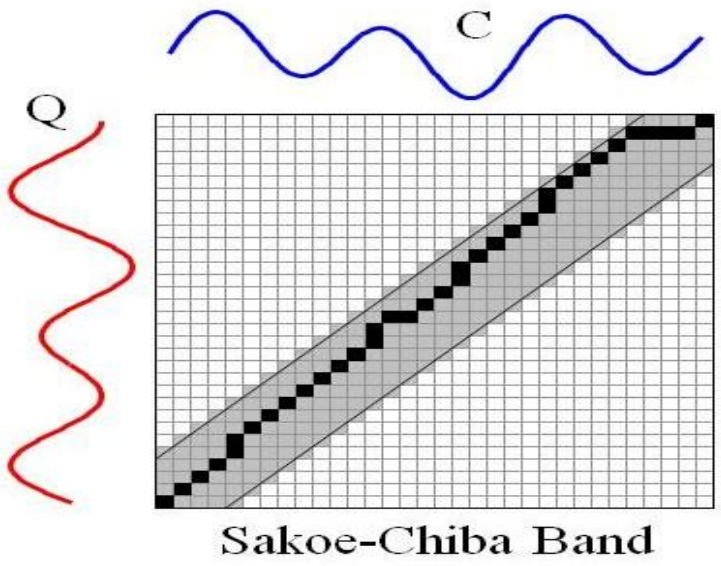
3.6 ΟΛΙΚΟΙ ΠΕΡΙΟΡΙΣΜΟΙ ΣΤΟΝ DTW

Θέλοντας οι ερευνητές της εξόρυξης πληροφορίας να μειώσουν το χρόνο αλλά και την πολυπλοκότητα υπολογισμού του DTW έθεσαν κάποιους περιορισμούς, οι οποίοι καθορίζουν την περιοχή των κελιών του πίνακα αποστάσεων που θα υπολογιστεί. Τα κελιά του πίνακα, τα οποία είναι εκτός περιοχής δεν μας ενδιαφέρουν και δεν υπολογίζονται. Δυο από τις πιο συχνά χρησιμοποιούμενες μεθόδους που θέτουν τέτοιου είδους περιορισμούς είναι οι Sakoe-Chiba (Sakoe and Chiba 1978) και Itakura (Itakura 1975). Οι μέθοδοι αυτοί επιβάλλουν ένα ολικό όριο στο πλήθος των κελιών του πίνακα αποστάσεων του DTW που θα πρέπει να υπολογιστούν, έτσι ώστε να μειωθεί ο συνολικός χρόνος υπολογισμού της DTW απόστασης. Τις περισσότερες φορές ένα σχετικά μικρό τμήμα του πίνακα αποστάσεων του DTW μπορεί να είναι αντιπροσωπευτικό ενός μεγαλύτερου τμήματος και το γεγονός αυτό το εκμεταλλεύονται αυτού του είδους οι

αλγόριθμοι περιορίζοντας το warping path. Η περιοχή του πίνακα όπου επιτρέπεται να «βρίσκεται» το warping path ονομάζεται warping window. Επίσης, οι περιορισμοί αυτοί αποτρέπουν ακραίες / στρεβλές περιπτώσεις ευθυγράμμισης δύο χρονοσειρών (ένα σημείο της μίας χρονοσειράς να αντιστοιχεί σε όλα τα σημεία της άλλης χρονοσειράς πλην ενός).

3.6.1 ΠΕΡΙΟΡΙΣΜΟΣ SAKOE-CHIBA

Ο περιορισμός Sakoe-Chiba δεν επιτρέπει στον αλγόριθμο του DTW να επεκταθεί πέρα από μια σταθερή απόσταση r από την κύρια διαγώνιο του πίνακα των αποστάσεων. Υπολογίζεται δηλαδή μόνο ένα μέρος του πίνακα. Η παράμετρος r είναι ένα ποσοστό το οποίο παίρνει τις τιμές στο διάστημα $0 \leq r \leq 100$. Αυτό το ποσοστό υπολογίζει πόσο μακριά μπορούν να είναι οι συντεταγμένες (δείκτες) των οποίων η απόσταση υπολογίζεται $j-r \leq i \leq j+r$. Στο παρακάτω σχήμα (Εικόνα 3.5) φαίνεται ο περιορισμός του φίλτρου Sakoe-Chiba διατηρώντας μια σταθερή απόσταση κατά μήκος της κύριας διαγωνίου.

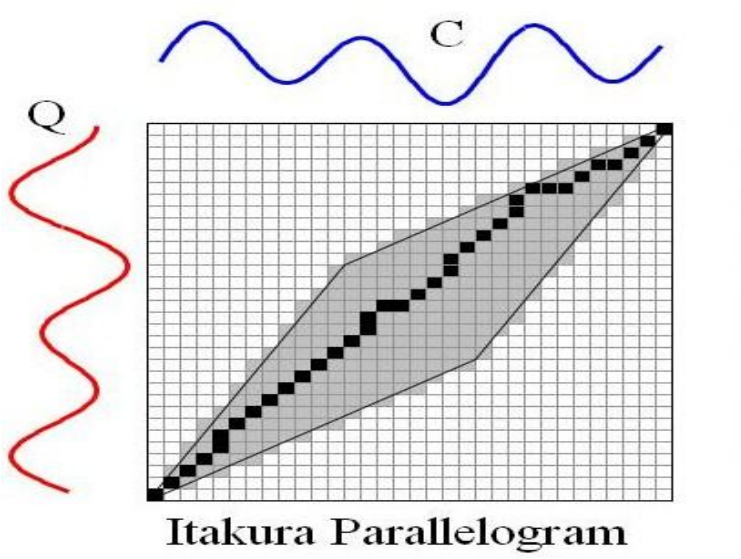


Εικόνα 3.5 Φίλτρο Sakoe-Chiba band

Πηγή : Keogh, E. J. (2001). "A Tutorial on Indexing and Mining Time Series Data", In Proceedings of IEEE International Conference on Data Mining, San Jose, November29

3.6.2 ΦΙΛΤΡΟ ITAKURA

Στο φίλτρο Itakura όπως και στο Sakoe-Chiba υπάρχει η παράμετρος r η οποία δεν επιτρέπει στον αλγόριθμο DTW να επεκταθεί πέρα από αυτή την απόσταση r . Βασική διαφορά είναι πως εδώ το r δεν είναι σταθερό αλλά είναι συνάρτηση των i, j. Το r στην αρχή του πίνακα των αποστάσεων αυξάνεται και έπειτα προς το τέλος μειώνεται με αποτέλεσμα να έχουμε ένα παραλληλόγραμμο με τα κελιά του πίνακα που χρειαζόμαστε. Το r εδώ ρυθμίζει την κλίση των πλευρών του παραλληλογράμμου. Όπως φαίνεται και στο παρακάτω σχήμα (Εικόνα 3.6) στο φίλτρο Itakura η απόσταση κατά μήκος της κύριας διαγωνίου αυξομειώνεται.



Εικόνα 3.6 Φίλτρο Itakura

Πηγή : Keogh, E. J. (2001). "A Tutorial on Indexing and Mining Time Series Data", In Proceedings of IEEE International Conference on Data Mining, San Jose, November 29

3.7 ΠΡΟΣΘΕΤΕΣ ΜΕΤΡΙΚΕΣ ΟΜΟΙΟΤΗΤΑΣ

Η Ευκλείδεια απόσταση και η DTW είναι οι μέθοδοι οι οποίες αποδεικνύονται ως οι περισσότερο αποτελεσματικές στην διεθνη βιβλιογραφία. Άλλες προτεινόμενες τεχνικές είναι οι εξής:

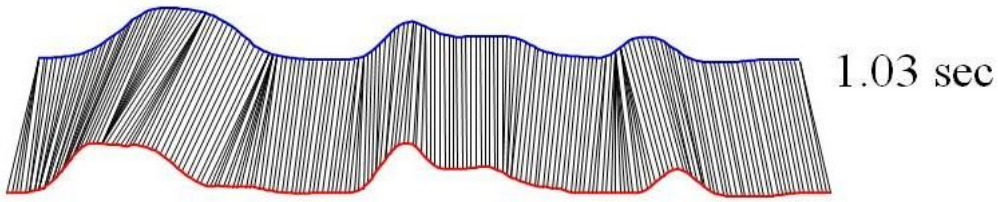
- *LCSS (Longest Common Subsequence).* Μια άλλη οικογένεια μέτρων απόστασης είναι η Μεγαλύτερη Κοινή Ακολουθία (LCSS), η οποία

χρησιμοποιείται συχνά για την αναγνώριση ομιλίας και την ταύτιση κειμένων. Γενική ιδέα είναι ότι δυο χρονοσειρές είναι παρόμοιες όταν παρουσιάζουν παρόμοια συμπεριφορά για ένα μεγάλο μέρος του μήκους τους (Agrawal et al, 1995) [4]. Η τεχνική αυτή μοιάζει με την DTW καθώς παρέχει και αυτή ευελιξία στο θέμα του χρόνου και επιτρέπει τη σύγκριση διαφορετικού μεγέθους χρονοσειρών, επιπλέον παρουσιάζει λιγότερη ευαισθησία σε θορύβους. Μειονέκτημα της είναι ότι απαιτεί τον προσδιορισμό μιας ακόμα παραμέτρου, γεγονός που σημαίνει επιπλέον υπολόγισμούς άρα επιπλέον χρόνος εκτέλεσης.

- *Hausdorff distance*. Ο γενικός ορισμός της μεθόδου αυτής είναι: «Το πιο απομακρυσμένο σημείο ενός συνόλου A μπορεί να είναι το πλησιέστερο σημείο σε ένα άλλο σύνολο B». Η μέθοδος αυτή εκφράζει τη χωρική ομοιότητα μεταξύ δύο χρονοσειρών. Είναι πολύ ευαίσθητη στους θορύβους και δεν είναι μετρική καθώς δεν στηρίζει την τριγωνική ανισότητα. (http://en.wikipedia.org/wiki/Hausdorff_distance).
- *HMM-based distance (Hidden Markov Models).* Είναι ένα μοντέλο ομοιότητας ακολουθιών το οποίο βασίζεται στην πιθανή αντιστοίχιση (Ge and Smyth, 2000) [10]. Εδώ η ομοιότητα μεταξύ δύο ακολουθιών S και S΄ μετράται με τον υπολογισμό της πιθανότητας ότι S΄ παράγεται από ένα μοντέλο, το οποίο είναι κατασκευασμένο από ένα μοντέλο Markov S το οποίο έχει ήδη χρησιμοποιηθεί.

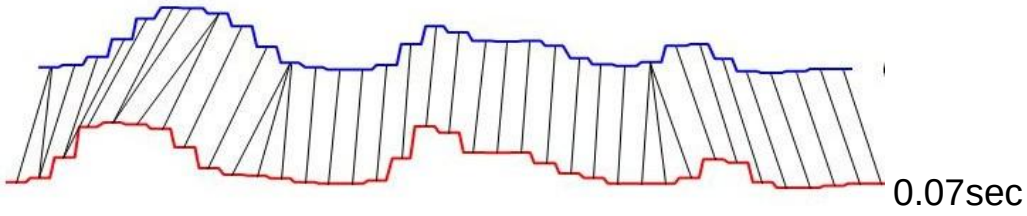
3.8 ΑΝΑΚΤΗΣΗ ΟΜΟΙΩΝ ΧΡΟΝΟΣΕΙΡΩΝ ΜΕ ΤΗ ΒΟΗΘΕΙΑ ΤΟΥ ΚΑΤΩΤΕΡΟΥ ΦΡΑΓΜΑΤΟΣ (LOWER BOUNDING)

Οι διάφορες τεχνικές μείωσης των διαστάσεων των χρονοσειρών που μελετήσαμε στο κεφάλαιο δυο, παίζουν σημαντικό ρόλο στην ταχύτερη ανάκτηση όμοιων χρονοσειρών διότι προκειμένου να υπολογιστεί η απόσταση μιας χρονοσειράς από μια άλλη εξετάζονται πλέον πολύ λιγότερα σημεία και αυτό έχει ως αποτέλεσμα ο χρόνος υπολογισμού να είναι πολύ μικρότερος σε σχέση με το χρόνο που θα χρειαζόταν για την επεξεργασία ακατέργαστων δεδομένων.



Εικόνα 3.7 Υπολογισμός απόστασης ακατέργαστων χρονοσειρών με βάση την DTW

Πηγή : Keogh, E. J. (2002). “Exact Indexing Of Dynamic Time Warping”, *In Proceedings of IEEE International Conference on Data Mining*



Εικόνα 3.8 Υπολογισμός απόστασης μειωμένης διάστασης χρονοσειρών με βάση τη DTW

Πηγή : Keogh, E. J. (2002). “Exact Indexing Of Dynamic Time Warping”, *In Proceedings of IEEE International Conference on Data Mining*

Παρόλα αυτά, ο αριθμός των προσβάσεων σε μια μεγάλη βάση δεδομένων για την ανάκτηση όμοιων χρονοσειρών παραμένει ο ίδιος, γεγονός που καθιστά τους αλγόριθμους ανάκτησης μη αποδοτικούς. Αυτό συμβαίνει καθώς είναι αναγκαία η πρόσβαση σε όλες τις χρονοσειρές της βάσης για την ανάκτηση των όμοιων χρονοσειρών.

Το πρόβλημα παρουσιάζεται κυρίως όταν έχουμε να κάνουμε με περισσότερες από δυο χρονοσειρές. Οι μετρικές απόστασης που είδαμε ως τώρα η Ευκλείδια απόσταση και η Dynamic Time Warping, υποστηρίζουν τον υπολογισμό της απόστασης μεταξύ δυο μόνο χρονοσειρών σε κάθε εφαρμογή τους. Στην περίπτωση που θέλουμε να ανακτήσουμε από μια βάση δεδομένων τις k περισσότερο όμοιες χρονοσειρές με τη χρονοσειρά query-Q (kNN Search) είναι απαραίτητη η ανάκτηση όλων των χρονοσειρών της βάσης ώστε να υπολογιστεί η απόσταση της κάθε μιας με την query-Q. Αφού υπολογιστεί η κάθε μια απόσταση

τότε γίνονται οι κατάλληλες συγκρίσεις έτσι ώστε να ανακτηθούν μόνο οι k χρονοσειρές που είναι πιο κοντά με τη χρονοσειρά query-Q. Ακρίβως το ίδιο συμβαίνει και με την περίπτωση της Range Query, όπου ζητείται να ανακτηθούν όλες οι χρονοσειρές οι οποίες διαφέρουν από την query-Q κατά ένα ποσοστό p. Και πάλι γίνεται ανάκτηση και υπολογισμός της απόστασης της όλων των χρονοσειρών της βάσης από την query-Q ώστε να ανακτηθούν όλες εκείνες που διαφέρουν από την Q κατά το ποσοστό p. Το γεγονός μάλιστα πως οι βάσεις δεδομένων όπου βρίσκονται καταγεγραμμένες οι χρονοσειρές είναι πάρα πολύ μεγάλες στο μέγεθος δυσκολέυει ακόμα πιο πολύ τη διαδικασία ανάκτησης όμοιων χρονοσειρών αφού για να συλλεχθούν όλες οι χρονοσειρές της βάσης, να υπολογιστούν όλες οι αποστάσεις και να γίνουν οι απαραίτητες συγκρίσεις χρειάζεται πολύ περισσότερος χρόνος. Εύκολα μπορεί να καταλάβει κανείς πως η προσέγγιση αυτή είναι σχεδόν ίδια με τη σειριακή αναζήτηση γ' αυτό και δεν είναι συμφέρουσα να χρησιμοποιηθεί στη διαδικασία ανάκτησης όμοιων χρονοσειρών από μεγάλες βάσεις δεδομένων (Παπανίκου, 2008) [2].

Όλα αυτά που αναφέρθηκαν παραπάνω οδήγησαν τους επιστήμονες στη διαπίστωση ότι η μείωση των διαστάσεων των χρονοσειρών δεν επαρκεί για την γρήγορη και αξιόπιστη ανάκτηση αποτελεσμάτων(Yi et al, 1998) [30]. Έτσι προτάθηκε μια τεχνική η οποία θα επέτρεπε λιγότερες προσβάσεις στη βάση δεδομένων, θα απομακρύνει όλες εκείνες τις χρονοσειρές που απέχουν κατά πολύ από την query-Q ώστε να υπολογίζεται τελικά η πραγματική απόσταση μόνο μεταξύ εκείνων των χρονοσειρών που βρίσκονται κοντά στη χρονοσειρά query-Q και όλα αυτά αφού φυσικά πρώτα εφαρμοστούν τεχνικές μείωσης της διάστασης των χρονοσειρών. Την τεχνική αυτή τη συναντήσαμε σε προηγούμενη ενότητα και είναι το *lower bounding*.

Η τεχνική lower bounding είναι απλή στην εφαρμογή της, υπολογίζει μια προσεγγιστική απόσταση μεταξύ των χρονοσειρών η οποία πρέπει να είναι μικρότερη ή το πολύ ίση με την πραγματική απόσταση. Επιτρέπει την απομάκρυνση ανόμοιων χρονοσειρών, κρατώντας μόνο τις όμοιες χωρίς παραλείψεις (false dismissals). Με αυτό τον τρόπο συμβάλλει ουσιαστικά στην αύξηση της ταχύτητας των υπολογισμών καθώς και στην εξαγωγή αξιόπιστων αποτελεσμάτων κατά τη διάρκεια της ανάκτησης όμοιων χρονοσειρών. Πολλές φορές υπάρχει περίπτωση να προκύψουν χρονοσειρές ανόμοιες με την query-Q (false alarms) από τον υπολογισμό της προσεγγιστικής απόστασης. Αυτό όμως

μπορεί να διορθωθεί γρήγορα με τον υπολογισμό της πραγματικής απόστασης μεταξύ των χρονοσειρών κατά το επόμενο στάδιο επεξεργασίας και να απομακρυνθούν όλες οι ανόμοιες χρονοσειρές, να μείνουν μόνο οι πραγματικά όμοιες με την query-Q ώστε τα αποτελέσματα να παραμείνουν αξιόπιστα.

ΚΕΦΑΛΑΙΟ 4ο

ΠΑΡΟΥΣΙΑΣΗ ΤΗΣ JAVA-ML(MACHINE LEARNING)

4.1 ΕΙΣΑΓΩΓΗ

Στο κεφάλαιο αυτό γίνεται μια περιληπτική / συνοπτική παρουσίαση της βιβλιοθήκης java-ml, την οποία χρησιμοποιήσαμε στην εργασία αυτή για να πραγματοποιήσουμε τα πειράματα του επόμενου κεφαλαίου. Επίσης, παρουσιάζεται περιληπτικά ο τρόπος με τον οποίο εισάγουμε και επεξεργαζόμαστε τους αλγόριθμους της βιβλιοθήκης στο περιβάλλον ανάπτυξης λογισμικού του Eclipse.

4.2 ΕΙΣΑΓΩΓΗ ΣΤΗ JAVA-ML

Η java-ml είναι μια βιβλιοθήκη η οποία περιλαμβάνει μια συλλογή αλγορίθμων που αφορούν τη διαδικασία εξόρυξης πληροφορίας. Το όνομα της προκύπτει από το ότι είναι γραμμένη στην αντικειμενοστραφή γλώσσα προγραμματισμού java και από τη μηχανική μάθηση (machine learning). Αποτελεί ένα πακέτο λογισμικού ανοιχτού κώδικα (open source), μπορεί κανείς να το αποκτήσει από την εξής πηγή: http://java-ml.sourceforge.net/ και υποστηρίζεται από όλες τις γνωστές πλατφόρμες (Unix/Linux, Windows, Macintosh). Η βιβλιοθήκη δεν διαθέτει γραφικό περιβάλλον και για αυτό προορίζεται κυρίως για προγραμματιστές όσο και ερευνητές.

Κατά καιρούς έχουν αναπτυχθεί αρκετές βιβλιοθήκες με βασικό αντικείμενο την εξόρυξη πληροφορίας όπως το WEKA (Witten and Frank, 2005) και το Yale/RapidMiner (Mierswa et al., 2006). Οι βιβλιοθήκες αυτές παρέχουν ένα φιλικό interface και είναι προσανατολισμένες προς τη διαδραστική επικοινωνία με τον χρήστη. Σε αντίθεση, η java-ml είναι προσανατολισμένη προς τους χρήστες που θέλουν να χρησιμοποιήσουν τη μηχανική μάθηση στα δικά τους προγράμματα παρέχοντας καλή τεκμηρίωση (documentation) και παραδείγματα πάνω στις

λειτουργίες και στο σύνολο των αλγορίθμων που υλοποιεί. Όσο αναφορά το περιεχόμενο της java-ml έχει επίσης μια διαφορετική εστίαση από τις άλλες βιβλιοθήκες. Ο μεγάλος αριθμός αλγορίθμων ομοιότητας διευκολύνει την ομαδοποίηση (clustering)και την κατηγοριοποίηση (classification) ενώ οι τεχνικές επιλογής καθιστούν τους αλγόριθμους της java-ml κατάλληλους για την εφαρμογή τους σε μεγάλες βάσεις δεδομένων.

4.3 ΔΟΜΗ - ΠΑΚΕΤΑ ΤΗΣ JAVA-ML

Ο κώδικας της java-ml αποτελείται από πολλά, μικρού μεγέθους κυρίως προγράμματα γραμμένα σε γλώσσα προγραμματισμού java, καθένα από τα οποία υλοποιείται σε μια κλάση. Κάθε κλάση περιέχει ιδιότητες και μεθόδους. Οι κλάσεις οργανώνονται σε packages καθένα από τα οποία περιλαμβάνει μια συλλογή από συσχετιζόμενες κλάσεις. Στο documentation της βιβλιοθήκης διατίθενται λεπτομερείς πληροφορίες σχετικά με τη δομή του κώδικα. Με τον όρο documentation εννοούμε το τεχνικό εγχειρίδιο σχετικά με τη λειτουργία της βιβλιοθήκης.

Το βασικότερο πακέτο γύρω από το οποίο είναι χτισμένη η java-ml είναι το net.sf.javaml.core το οποίο περιέχει τις δυο βασικές διεπαφές της βιβλιοθήκης : Dataset και Instance. Αυτές οι δυο διεπαφές έχουν αρκετές εφαρμογές για διαφορετικού τύπου παραδείγματα. Το Dataset αποτελεί τη διεπαφή για ένα σύνολο δεδομένων (data-set) ενώ το Instance αποτελεί τη διεπαφή ώστε να ορίζονται ‘δείγματα’ των δεδομένων του πραγματικού κόσμου όπως π.χ. ορισμός μιας χρονοσειράς. Τα DenseInstance και SparseInstance υλοποιούν το Instance. Στο DenseInstance απαιτείται μια τιμή για κάθε χαρακτηριστικό ενώ στο SparseInstance μπορούν να παραλειφθούν τιμές.

Ένα ακόμα σημαντικό πακέτο της java-ml είναι το net.sf.javaml.classification το οποίο περιέχει ένα σύνολο αλγορίθμων κατηγοριοποίησης. Το interface όλων αυτών είναι το Classifier. Στην παρούσα εργασία εμείς θα χρησιμοποιήσουμε τον αλγόριθμο KNearestNeighbors ο οποίος ζητά από μια βάση δεδομένων να ανακτηθούν οι k περισσότερο όμοιες χρονοσειρές με μια δοθείσα χρονοσειρά query Q, το k δηλαδή είναι παραμετροποιήσιμο.

Το πακέτο net.sf.javaml.distance περιλαμβάνει αλγόριθμους που μετρούν την απόσταση/ομοιότητα μεταξύ δυο δειγμάτων (Instances). Η κλάση AbstractSimilarity αποτελεί την υπέρ-κλάση για όλους τους αλγόριθμους ομοιότητας. Εδώ συναντούμε αλγόριθμους όπως την Ευκλείδεια Απόσταση (EuclideanDistance) και την απόσταση Manhattan (ManhattanDistance).

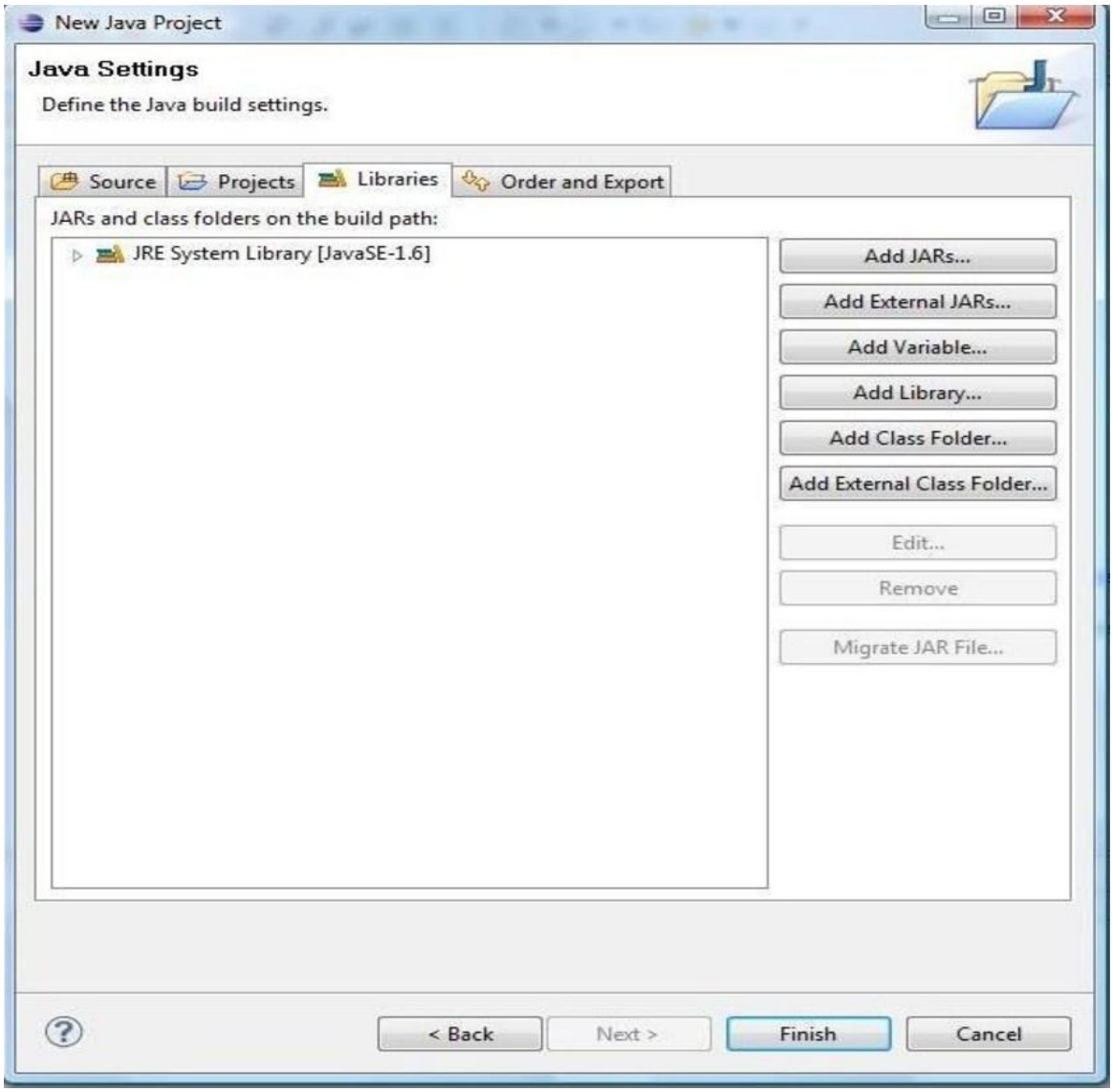
Το πακέτο net.sf.javaml.distance.dtw είναι αυτό το οποίο περιέχει τον βασικό αλγόριθμο του DTW (DTWSimilarity) πάνω στον οποίο θα στηριχτούμε για να εκτελέσουμε έναν αριθμό πειραμάτων στην επόμενη ενότητα αυτής της εργασίας. Ο DTW έχει ως υπέρ-κλάση την AbstractSimilarity που αναφέρθηκε προηγουμένως.

4.4 JAVA-ML ΚΑΙ ECLIPSE

Το περιβάλλον ανάπτυξης λογισμικού που χρησιμοποιήσαμε για να ενσωματώσουμε τη βιβλιοθήκη java-ml αλλά και για την ανάπτυξη των αλγόριθμων είναι το **Eclipse** και η έκδοση **Eclipse IDE for Java Developers** την οποία μπορεί να ανακτήσει κανείς από την εξής ηλεκτρονική διεύθυνση : http://www.eclipse.org/downloads/. Αρχικά «κατεβάζουμε» το εργαλείο του Eclipse καθώς και τη βιβλιοθήκη της java-ml και τα εγκαθιστούμε στον υπολογιστή μας. Αποσυμπιέζουμε το αρχείο της java-ml, μέσα σ΄ αυτό υπάρχουν δυο αρχεία που μας ενδιαφέρουν το javaml-0.1.6.jar και το javaml-0.1.6-src.zip. Το πρώτο είναι η βιβλιοθήκη την οποία θα εισάγουμε στο project μας και το δεύτερο περιέχει τον κώδικα της βιβλιοθήκης.

Για να δημιουργήσουμε ένα νέο project, αφού εισέλθουμε στο κυρίως περιβάλλον του Eclipse ακολουθούμε την εξής διαδρομή: *File -> New -> Java Project* όπου και δίνουμε ένα όνομα στο αρχείο που θέλουμε να δημιουργήσουμε και έπειτα επιλέγουμε next. Στο επόμενο μενού (*java settings*) επιλέγουμε την καρτέλα *Libraries* και την επιλογή *Add External JARs* (Εικόνα 4.1). Εκεί εισάγουμε το αρχείο javaml-0.1.6.jar από αυτά που αποσυμπιέσαμε προηγουμένως και πατάμε finish. Με αυτόν τον τρόπο στο project μας θα είναι πλέον διαθέσιμες όλες οι τάξεις της βιβλιοθήκης και μπορούμε μέσα από το API της (http://java-ml.sourceforge.net/api/0.1.6/) να περιηγηθούμε στο σύνολο των λειτουργιών της java-ml. Με τον όρο API (Application Programming Interface) εννοούμε τη διεπαφή των προγραμματιστικών διαδικασιών που παρέχει μια βιβλιοθήκη, ένα

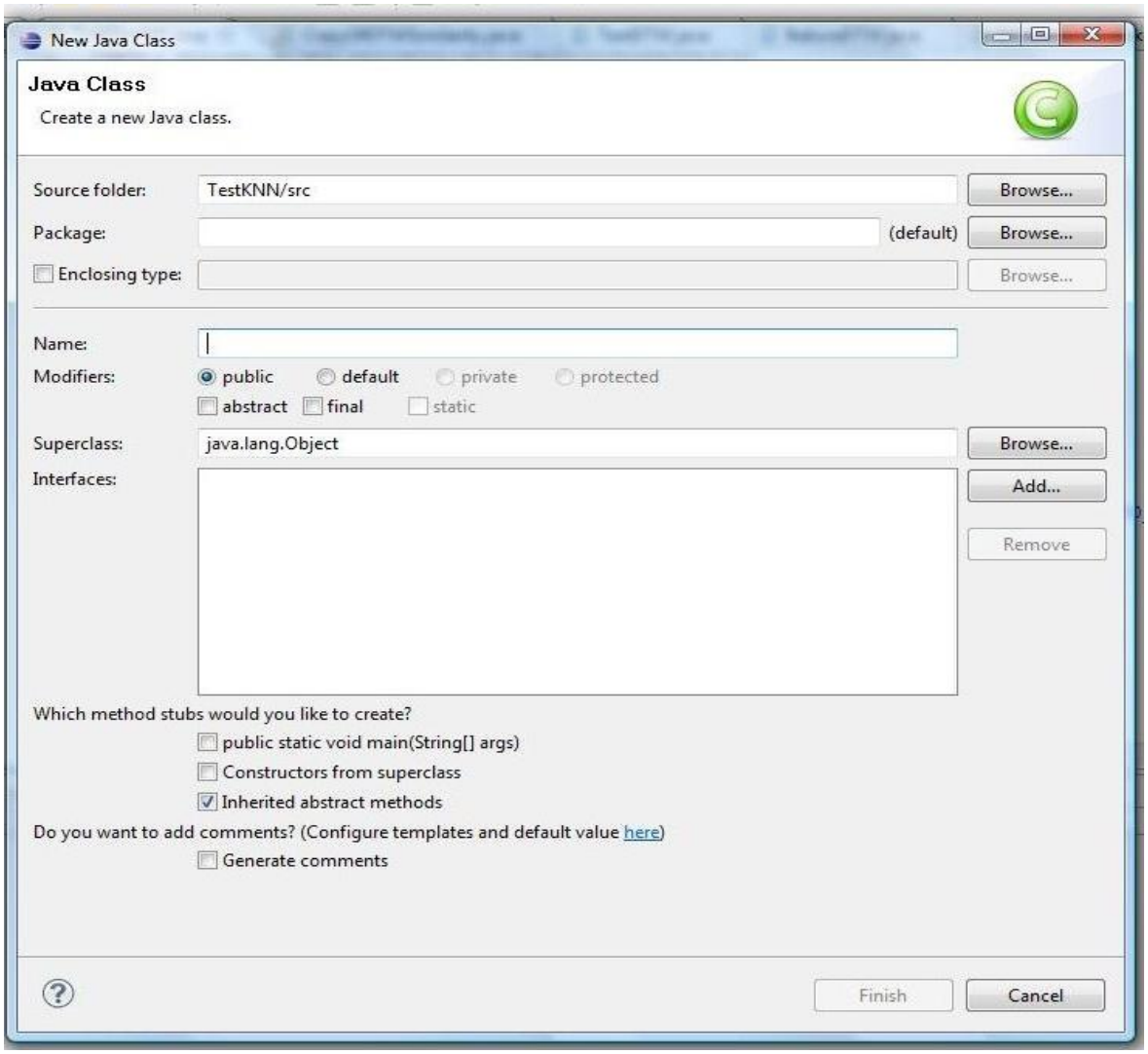
λειτουργικό σύστημα, μια εφαρμογή προκειμένου να επιτρέπει να γίνονται προς αυτήν αιτήσεις από άλλα προγράμματα ή και ανταλλαγή δεδομένων. Βασικός σκοπός του API εδώ, είναι να ορίζει και να διατυπώνει το σύνολο των λειτουργιών/υπηρεσιών που μπορεί να παρέχει μια βιβλιοθήκη χωρίς να επιτρέπει πρόσβαση στον κώδικα που υλοποιεί αυτές τις υπηρεσίες.



Εικόνα 4.1 Καρτέλα «java settings»

Για να μπορέσουμε να έχουμε πρόσβαση στον κώδικα της βιβλιοθήκης, να κάνουμε τροποποιήσεις, να εισάγουμε κλάσεις ακολουθούμε τα εξής βήματα: Πάνω αριστερά στην οθόνη μας βλέπουμε ένα φάκελο με το όνομα που δώσαμε στο project μας .Στη ρίζα (src) του φακέλου όπου είναι η ρίζα που θα περιέχει τον

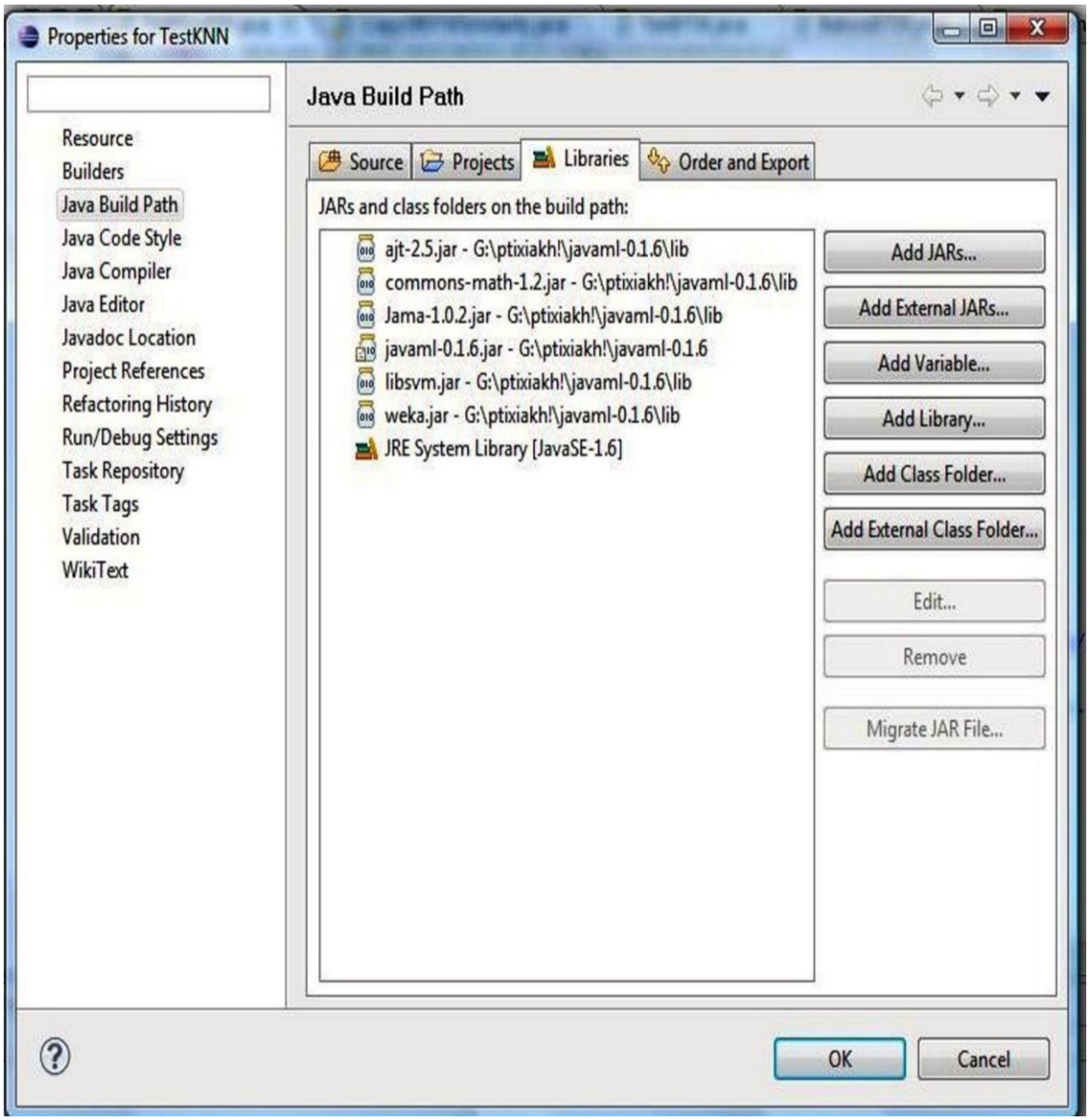
κώδικα μας, κάνουμε δεξί κλικ και επιλέγουμε *new -> Class* (Εικόνα 4.2)ώστε να δώσουμε ένα όνομα στο αρχείο μας, εδώ είναι καλύτερα να δημιουργήσουμε ένα νέο πακέτο και να τοποθετήσουμε εκεί το αρχείο μας.



Εικόνα 4.2 Δημιουργία νέας κλάσης

Στο σημείο αυτό για να μπορέσουμε να επεξεργαστούμε τον κώδικα της java-ml στο eclipse θα πρέπει να εισάγουμε στο project μας (import) τα sources. Για να γίνει αυτό αποσυμπιέζουμε το javaml-0.1.6-src.zip και αντιγράφουμε τα στοιχεία του στη ρίζα του φακέλου του project στο eclipse. Κατόπιν κάνουμε δεξί κλικ στο project μας *properties -> java build path -> libraries -> add external JARs* (Σχήμα 4.3)και εκεί προσθέτουμε από το φάκελο lib του αρχικού javaml-0.1.6 τις

πέντε διαθέσιμες βιβλιοθήκες τις οποίες εμπεριέχει η java-ml και βοηθούν στη μεταγλώττιση. Τώρα είμαστε σε θέση να «εκμεταλλευτούμε» τη δυνατότητα που μας παρέχει το γεγονός πως η java-ml βασίζεται σε κώδικα ανοιχτού λογισμικού, δηλαδή να επεξεργαστούμε τους αλγόριθμους της βιβλιοθήκης είτε τροποποιώντας τους ήδη υπάρχοντες είτε ενσωματώνοντας νέους.



Σχήμα 4.3 Java Build Path

ΚΕΦΑΛΑΙΟ 5ο

ΠΕΙΡΑΜΑΤΙΚΗ ΑΞΙΟΛΟΓΗΣΗ ΑΛΓΟΡΙΘΜΩΝ

5.1 ΕΙΣΑΓΩΓΗ

Στο κεφάλαιο αυτό παρουσιάζονται τα αποτελέσματα των πειραμάτων που εκτελέσαμε με βάση τη βιβλιοθήκη της java-ml και τους βασικούς αλγόριθμους αναζήτησης ομοιότητας της διαδικασίας εξόρυξης πληροφορίας μέσω ανάλυσης χρονοσειρών και πραγματοποιούνται οι απαραίτητες συγκρίσεις.

5.2 ΠΕΡΙΓΡΑΦΗ ΤΩΝ ΣΥΝΟΛΩΝ ΔΕΔΟΜΕΝΩΝ

Θέλοντας να ελέγξουμε την αποδοτικότητα των βασικών μεθόδων ανάκτησης όμοιων χρονοσειρών, πραγματοποιήσαμε μια σειρά πειραμάτων χρησιμοποιώντας κάποια σύνολα δεδομένων (datasets). Τα σύνολα δεδομένων αυτά αποτελούν σημείο αναφοράς για τη δοκιμή αλγορίθμων κατηγοριοποίησης και γ΄ αυτό το λόγο είναι διαχωρισμένα σε σύνολα εκπαίδευσης (train set) και ελέγχου (test set). Τα σύνολα αυτά περιλαμβάνουν ένα πλήθος χρονοσειρών για τις οποίες γνωρίζουμε σε ποια κλάση ανήκει η κάθε μια. Το train set χρησιμοποιείται για την κατασκευή / εκπαίδευση ενός μοντέλου κατηγοριοποίησηςς, ενώ το test set χρησιμοποιείται για την αξιολόγησή του. Όσο αναφορά τη δομή ενός dataset, κάθε μια γραμμή αντιστοιχεί σε μια χρονοσειρά. Στην πρώτη στήλη/πεδίο υπάρχει η κλάση στην οποία ανήκει, ενώ στις υπόλοιπες είναι καταχωρημένες οι τιμές της χρονοσειράς. Στον Πίνακα 5.1 παρουσιάζονται τα βασικά χαρακτηριστικά των συνόλων δεδομένων που χρησιμοποιήθηκαν.

5.3 ΜΕΘΟΔΟΣ ΑΞΙΟΛΟΓΗΣΗΣ

Η πραγματοποίηση των πειραμάτων με βάση το σύνολα δεδομένων (datasets) έγινε χρησιμοποιώντας τη γνωστή μέθοδο αναζήτησης όμοιων

χρονοσειρών kNN, όπου k=1, την οποία έχουμε αναφέρει και σε προηγούμενα κεφάλαια στην εργασία αυτή. Με βάση τη μέθοδο αυτή ζητείται από μια βάση δεδομένων να ανακτηθούν οι k περισσότερο όμοιες χρονοσειρές με μια δοθείσα χρονοσειρά query Q. Στην περίπτωση μας αναζητούμε μόνο μια χρονοσειρά, η οποία είναι η περισσότερο όμοια με την query Q.

Πίνακας 5.1-Περιγραφή των datasets

Α/Α	ΣΥΝΟΛΟ	ΠΛΗΘΟΣ ΚΛΑΣΕΩΝ	ΜΕΓΕΘΟΣ ΣΥΝΟΛΟΥ ΕΚΠΑΙΔΕΥΣΗΣ	ΜΕΓΕΘΟΣ ΣΥΝΟΛΟΥ ΕΛΕΓΧΟΥ	ΜΗΚΟΣ ΧΡΟΝΟ ΣΕΙΡΑΣ
1	50words	50	450	455	270
2	CBF	3	30	900	128
3	ECG200	2	100	100	96
4	FaceFour	4	24	88	350
5	GunPoint	2	50	50	150
6	Lingthing2	2	60	61	637
7	OSULeaf	6	200	242	627
8	SwedishLeaf	15	500	625	128
9	Synthetic_control	6	300	300	60
10	Trace	4	100	100	275

Σε μια ρεαλιστική εφαρμογή έχουμε ένα σύνολο χρονοσειρών (train set) για τις οποίες γνωρίζουμε σε ποια κλάση ανήκει η κάθε μια. Όταν έρχεται μια νέα χρονοσειρά Q της οποίας την κλάση δεν γνωρίζουμε, υπολογίζουμε την απόσταση της από κάθε μια χρονοσειρά της βάσης δεδομένων και επιλέγουμε αυτή με τη μικρότερη απόσταση, έστω x. Κατηγοριοποιούμε την Q στην κλάση όπου ανήκει η x. Στην δική μας περίπτωση, κατηγοριοποιούμε τις χρονοσειρές που ανήκουν στο test set εφαρμόζοντας τον αλγόριθμο 1-NN και μετράμε πόσες επιτυχίες ή αποτυχίες έχουμε.

Με βάση τη μέθοδο κατηγοριοποίησης 1-NN για ένα δοθέν σύνολο δεδομένων, πιο συγκεκριμένα η διαδικασία είναι η εξής:

1. Ανακτάται μια χρονοσειρά Q από το test set.
2. Υπολογίζεται η απόσταση της από όλες τις χρονοσειρές του train set.
3. Επιλέγεται η χρονοσειρά με τη μικρότερη απόσταση και ανακτάται η κλάση στην οποία ανήκει.

4. Αν η κλάση αυτή είναι ίδια με την κλάση της Q τότε η κατηγοριοποίηση είναι σωστή.
5. Η διαδικασία αυτή επαναλαμβάνεται για κάθε χρονοσειρά του test set.

   Το ποσοστό λάθους της κατηγοριοποίησης είναι ο αριθμός των λάθος κατατάξεων δια του συνολικού μεγέθους του test set.

Ο χρόνος απόκρισης ενός αλγόριθμου για ένα σύνολο δεδομένων υπολογίζεται με την εκτέλεση του και καταγραφή του χρόνου απόκρισης και με την επανάληψη της διαδικασίας αυτής 10 φορές. Ο μέσος όρος των χρόνων αυτών αποτελεί το χρόνο απόκρισης του συγκεκριμένου αλγόριθμου για το συγκεκριμένο σύνολο δεδομένων.

5.4 ΑΛΓΟΡΙΘΜΟΙ – ΠΑΡΑΜΕΤΡΟΙ

Με βάση τη μέθοδο 1-NN την οποία αναφέραμε στην προηγούμενη ενότητα υλοποιήσαμε τα πειράματα μας με τη βοήθεια ενός συνόλου αλγορίθμων υπολογισμού της απόστασης μεταξύ χρονοσειρών.

Στην πρώτη φάση των πειραμάτων χρησιμοποιήσαμε τους δυο πιο συχνά συναντόμενους αλγόριθμους της διαδικασίας εξόρυξης πληροφορίας μέσω ανάλυσης χρονοσειρών, την Ευκλείδεια απόσταση και την τεχνική απόστασης DTW. Ας σημειωθεί ότι ο 1-NN εφαρμόζεται εξ΄ ορισμού με τον υπολογισμό της Ευκλείδειας απόστασης. Κατόπιν, συγκρίναμε τα αποτελέσματα αυτών τον δυο αλγορίθμων με ένα αλγόριθμο της java-ml βιβλιοθήκης τον FastDTW, ο οποίος παρουσιάζεται ως μια καλύτερη εκδοχή του απλού DTW. Ο FastDTW είναι ένας αλγόριθμος τον οποίο πρώτη φορά συναντούμε στην παρούσα εργασία και ο οποίος ανήκει στη βιβλιοθήκη της java-ml στο πακέτο net.sf.javaml.distance.fastdtw. Ο FastDTW αποτελεί μια προσπάθεια των δημιουργών της java-ml για τη βελτίωση του απλού DTW και έχει ως στόχο να μειώσει το εύρος των τιμών που αναζητείται ώστε να προκύπτουν καλύτερα και γρηγορότερα αποτελέσματα σε σχέση πάντα με τον απλό DTW. Γ’ αυτό και εδώ υπάρχει η παράμετρος r , η οποία μειώνει το εύρος των τιμών που αναζητούμε και στα πειράματά μας δίνουμε σ΄αυτή την τιμή r = 5 όπου είναι η τιμή που προτείνεται στην πλειοψηφία της σχετικής βιβλιογραφίας. Περισσότερες πληροφορίες για τη λειτουργία του FastDTW μπορεί να βρει κανείς στο αντίστοιχο paper της βιβλιοθήκης http://cs.fit.edu/~pkc/papers/tdm04.pdf .

Κατά τη διάρκεια της δεύτερης φάσης των πειραμάτων χρησιμοποιήσαμε τους δυο αλγόριθμους οι οποίοι θέτουν περιορισμούς στον «κλασσικό» DTW, τους Sakoe-Chiba και Itakura. Στον Sakoe-Chiba οι παράμετροι με τους οποίους υλοποιήσαμε τα πειράματα είναι r = 1, 5, 10 ενώ στον Itakura οι παράμετροι είναι r = 1.5, 2, 2.5. Οι παράμετροι που χρησιμοποιήθηκαν στους αλγορίθμους αυτούς γενικότερα επιλέγονται στην σχετική βιβλιογραφία.

5.5 ΑΠΟΤΕΛΕΣΜΑΤΑ

Στις επόμενες ενότητες θα περιγράψουμε τα αποτελέσματα των πειραμάτων που εκτελέσαμε όσο αναφορά τα σφάλματα που προέκυψαν από την κατηγοριοποίηση των χρονοσειρών με τον εκάστοτε αλγόριθμο που χρησιμοποιήθηκε, καθώς επίσης και τους χρόνους απόκρισης αυτών.

5.5.1 ΠΕΙΡΑΜΑΤΙΚΑ ΑΠΟΤΕΛΕΣΜΑΤΑ : ΕΥΚΛΕΙΔΙΑ ΑΠΟΣΤΑΣΗ, DTW ΚΑΙ FAST_DTW

Στον πίνακα 5.2 παρουσιάζονται τα αποτελέσματα των σφαλμάτων κατηγοριοποίησης χρησιμοποιώντας τη μέθοδο 1-NN και τους αλγόριθμους μέτρησης απόστασης μεταξύ χρονοσειρών, Ευκλείδεια απόσταση, DTW (Dynamic Time Warping) και FastDTW.

Τα ποσοστά λάθους της κατηγοριοποίησης προκύπτουν από τη διαίρεση του αριθμού των χρονοσειρών που κατηγοριοποιήθηκαν εσφαλμένα δια (/) του συνολικού μεγέθους του test set της εκάστοτε χρονοσειράς. Στη στήλη ποσοστό σφάλματος % καταγράφεται το ποσοστό λάθους επί της 100 (%). Το μικρότερο ποσοστό σφάλματος που προκύπτει για το κάθε dataset εμφανίζεται σε σκιασμένο κελί.

Σύμφωνα λοιπόν με τον Πίνακα 5.2, συγκρίνοντας αρχικά τα ποσοστά σφάλματος της Ευκλείδειας απόστασης και του DTW παρατηρούμε πως τα ποσοστά λάθους του DTW αλγορίθμου στην πλειοψηφία τους είναι μικρότερα από αυτά της Ευκλείδειας απόστασης, για την ακρίβεια αυτό συμβαίνει στα οχτώ από τα δέκα datasets. Η διαφορά κυμαίνεται από 0,3 έως 24 ποσοστιαίες μονάδες.. Στη συνέχεια όμως εκτελώντας τα πειράματα και με τον αλγόριθμο FastDTW

παρατηρούμε πως έχοντας περιορίσει το εύρος των τιμών αναζήτησης (r=5) τα ποσοστά σφάλματος μειώνονται αρκετά σε σχέση με τον κλασικό DTW αλγόριθμο, δηλαδή σε επτά από τα δέκα datasets και σε ένα dataset όπου το ποσοστό παραμένει ίδιο. Η μείωση αυτή κυμαίνεται από 0,2 έως 11,1 ποσοστιαίες μονάδες. Έτσι παρατηρούμε από τη γενική εικόνα του πίνακα 5.2 πως τα αποτελέσματα χρησιμοποιώντας τον αλγόριθμο του FastDTW είναι στις περισσότερες των περιπτώσεων πιο αξιόπιστα σε σχέση με τα αντίστοιχα των άλλων δυο αλγορίθμων.

Πίνακας 5.2 – 1-NN Ευκλείδεια - DTW - FastDTW – ποσοστά σφάλματος (%)

Α/Α	ΣΥΝΟΛΟ	ΕΥΚΛΕΙΔΕΙΑ ΑΠΟΣΤΑΣΗ	DTW	FastDTW (r=5)
1	50words	36.9	31.0	29.0
2	CBF	14.8	0.3	0.1
3	ECG200	12.0	23.0	19.0
4	FaceFour	21.6	17.0	15.0
5	GunPoint	8.7	9.3	12.0
6	Lingthing2	24.6	13.1	2.0
7	OSULeaf	48.3	40.9	37.0
8	SwedishLeaf	21.1	20.8	20.8
9	Synthetic_control	12.0	7.0	1.3
10	Trace	24.0	0.0	1.0

Αφού παρουσιάσαμε τα αποτελέσματα όσο αναφορά τα ποσοστά σφάλματος κατά την κατηγοριοποίηση των dataset χρησιμοποιώντας τη μέθοδο 1-NN και τους αλγόριθμους της Ευκλείδειας απόστασης, του DTW και του FastDTW, τώρα θα γίνει μια παρουσίαση των αποτελεσμάτων όσο αναφορά τους χρόνους απόκρισης των αλγορίθμων αυτών.

Όπως φαίνεται στον Πίνακα 5.3, οι μικρότεροι σε διάρκεια χρόνοι εμφανίζονται στον αλγόριθμο της Ευκλείδειας απόστασης καθώς είναι ο αλγόριθμος αυτός με τη μικρότερη πολυπλοκότητα σε σχέση με τους άλλους δυο.

Από τα αποτελέσματα του χρόνου για τον DTW καταλαβαίνει κανείς πόσο αργός είναι ο αλγόριθμος αυτός σε σχέση με την Ευκλείδεια απόσταση, καθώς ο χρόνος που προκύπτει σ' αυτόν είναι από 33 έως 640 φορές μεγαλύτερος από τον αντίστοιχο της Ευκλείδειας απόστασης στα συγκεκριμένα dataset. Ο FastDTW κάνει μια προσπάθεια να μειώσει το χρόνο εκτέλεσης του απλού DTW μειώνοντας το εύρος του πίνακα (r=5) και σύμφωνα με τα αποτελέσματα του πίνακα το καταφέρνει στα πέντε από τα δέκα datasets (ο DTW απαιτεί από 1,5 έως 4,6 φορές περισσότερο χρόνο).

Πίνακας 5.3 – 1-NN Ευκλείδεια - DTW - FastDTW – χρόνοι απόκρισης (sec)

Α/Α	ΣΥΝΟΛΟ	ΕΥΚΛΕΙΔΕΙΑ ΑΠΟΣΤΑΣΗ	DTW	FastDTW (r=5)
1	50words	4.9	877	570
2	CBF	0.5	27	33
3	ECG200	0.13	5.6	8.6
4	FaceFour	0.13	22	7.7
5	GunPoint	0.15	9.6	10.6
6	Lingthing2	0.2	128	28
7	OSULeaf	2	706	226
8	SwedishLeaf	3.6	293	369
9	Synthetic_control	0.6	20	44
10	Trace	0.3	54	28

5.5.2 ΠΕΙΡΑΜΑΤΙΚΑ ΑΠΟΤΕΛΕΣΜΑΤΑ : SAKOE/CHIBA

Στη συνέχεια των πειραμάτων μας ασχοληθήκαμε με τον αλγόριθμο (φίλτρο) Sakoe-Chiba ο οποίος επιβάλλει έναν περιορισμό στον DTW ώστε να μην μπορεί να επεκταθεί πέρα από μια σταθερή απόσταση r από την κύρια διαγώνιο του πίνακα των αποστάσεων. Η παράμετρος r μας δείχνει το ποσοστό σύμφωνα με το οποίο θα καθοριστεί το εύρος αναζήτησης για τον DTW. Στα πειράματά μας εκτελέσαμε τον αλγόριθμο Sakoe-Chiba για r = 1, r = 5, r = 10. Στον πίνακα 5.4

φαίνεται το ποσοστό επί τις εκατό των σφαλμάτων κατηγοριοποίησης για τον Sakoe-Chiba, τα χρωματισμένα κελιά δείχνουν το μικρότερο ποσοστό για κάθε dataset.

Παρατηρούμε ότι από τα δέκα datasets που χρησιμοποιήσαμε για τα πειράματά μας τα δυο από αυτά παρουσιάζουν μικρότερο ποσοστό λάθους για r = 1, δυο για r = 5, τέσσερα για r = 10, ένα dataset όπου το ποσοστό σφάλματος παραμένει αμετάβλητο για r = 1 και για r = 5 και ένα dataset όπου το ποσοστό σφάλματος παραμένει αμετάβλητο για r = 5 και για r = 10. Οι διαφορές που παρατηρούνται κυμαίνονται από 0 έως 8,2 ποσοστιαίες μονάδες.

Πίνακας 5.4 – 1-NN Sakoe/Chiba – ποσοστά σφάλματος (%)

A/A	**ΣΥΝΟΛΟ**	**Sakoe/Chiba r = 1**	**Sakoe/Chiba r = 5**	**Sakoe/Chiba r = 10**
1	50words	31.2	23.0	24.0
2	CBF	7.0	2.0	0.3
3	ECG200	12.0	11.0	18.0
4	FaceFour	11.4	13.6	17.0
5	GunPoint	3.0	3.0	6.0
6	Lingthing2	15.0	15.0	9.8
7	OSULeaf	44.6	42.0	40.0
8	SwedishLeaf	16.3	18.2	19.8
9	Synthetic_control	1.7	1.3	0.7
10	Trace	5.0	0.0	0.0

Σύμφωνα με τον πίνακα 5.5, στον οποίο παρουσιάζονται οι χρόνοι εκτέλεσης παρατηρούμε τα εξής:

Πίνακας 5.5 – 1-NN – Sakoe/Chiba - χρόνοι απόκρισης (sec)

Α/Α	ΣΥΝΟΛΟ	Sakoe/Chiba r = 1	Sakoe/Chiba r = 5	Sakoe/Chiba r = 10
1	50words	139	217	305
2	CBF	4.5	6.8	10
3	ECG200	1	1.5	2
4	FaceFour	3	4	5.5
5	GunPoint	1.8	2.6	3.6
6	Lingthing2	22.6	30.5	41.5
7	OSULeaf	84	128	185
8	SwedishLeaf	50	76	111
9	Synthetic_control	4.3	6	8
10	Trace	7	11	16

Οι μικρότεροι σε διάρκεια χρόνοι εμφανίζονται στον Sakoe-Chiba για r = 1 σε ποσοστό δέκα στα δέκα datasets. Σε κάποιες περιπτώσεις μάλιστα ένα dataset εμφανίζει διαφορές στους χρόνους αρκετά μεγάλες. Επίσης παρατηρούμε πως όσο το r αυξάνεται δηλαδή όσο μεγαλώνει το εύρος αναζήτησης των τιμών τόσο αυξάνεται και ο χρόνος εκτέλεσης του εκάστοτε dataset (όπως ήταν αναμενόμενο). Προσεγγιστικά, ο απαιτούμενος χρόνος είναι 1,5 φορές μεγαλύτερος, κατά μέσο όρο, στις περιπτώσεις όπου αυξάνουμε το r από 1 σε 5 και από 5 σε 10.

5.5.3 ΠΕΙΡΑΜΑΤΙΚΑ ΑΠΟΤΕΛΕΣΜΑΤΑ : ITAKURA

Στην πορεία των πειραμάτων μας εξετάσαμε τη συμπεριφορά του αλγόριθμου (φίλτρου) Itakura. Στο φίλτρο Itakura όπως και στο Sakoe-Chiba υπάρχει η παράμετρος r η οποία δεν επιτρέπει στον αλγόριθμο του απλού DTW να επεκταθεί πέρα από αυτή την απόσταση r, με τη βασική διαφορά ότι το r εδώ δεν είναι σταθερό δηλαδή, η απόσταση κατά μήκος της κύριας διαγωνίου του πίνακα των αποστάσεων του DTW αυξομειώνεται. Στα πειράματά μας εκτελέσαμε τον αλγόριθμο Itakura για r = 1.5 ,r = 2, r = 2.5 και τα αποτελέσματα για τα

ποσοστά σφάλματος παρουσιάζονται στον ακόλουθο Πίνακα 5.6, τα χρωματισμένα κελιά δείχνουν το μικρότερο ποσοστό για κάθε dataset.

Πίνακας 5.6 – 1-NN Itakura – ποσοστά σφάλματος (%)

Α/Α	**ΣΥΝΟΛΟ**	**Itakura r = 1.5**	**Itakura r = 2**	**Itakura r = 2.5**
1	50words	25.0	26.0	28.0
2	CBF	0.1	0.4	0.3
3	ECG200	14.0	18.0	20.0
4	FaceFour	18.0	16.0	16.0
5	GunPoint	6.0	10.0	11.3
6	Lingthing2	11.5	13.1	16.4
7	OSULeaf	37.0	40.1	40.5
8	SwedishLeaf	18.0	18.6	19.0
9	Synthetic_control	0.7	1.0	1.6
10	Trace	0.0	0.0	0.0

Από τον πίνακα προκύπτει από τα δέκα datasets τα οχτώ έχουν μικρότερο ποσοστό σφάλματος για r = 1.5, ένα για r = 2.5 και r = 2, και ένα του οποίου το ποσοστό σφάλματος παραμένει αμετάβλητο για όλες τις τιμές του r. Γενικότερα οι διαφορές ανάμεσα στις τιμές του r δεν είναι πολύ μεγάλες. Οι διαφορές που παρατηρούνται κυμαίνονται από 0 έως 6 ποσοστιαίες μονάδες.

Σύμφωνα με τον πίνακα των χρόνων απόκρισης (Πίνακας 5.7) για τις διαφορετικές τιμές του αλγόριθμου Itakura οι μικρότεροι σε διάρκεια χρόνοι εμφανίζονται για r = 1.5 σε ποσοστό δέκα στα δέκα datasets. Επίσης είναι φανερό πως όσο το r αυξάνεται δηλαδή όσο μεγαλώνει το εύρος αναζήτησης των τιμών τόσο αυξάνεται και ο χρόνος εκτέλεσης του εκάστοτε dataset. Προσεγγιστικά, ο απαιτούμενος χρόνος είναι 1,5 φορές μεγαλύτερος, κατά μέσο όρο, στις περιπτώσεις όπου αυξάνουμε το r από 1,5 σε 2 και από 2 σε 2,5.

Πίνακας 5.7 – 1-NN Itakura- χρόνοι απόκρισης (sec)

Α/Α	ΣΥΝΟΛΟ	Itakura r = 1.5	Itakura r = 2	Itakura r = 2.5
1	50words	508	713	955
2	CBF	17	23	30
3	ECG200	3.7	5	6.3
4	FaceFour	10	13.6	18
5	GunPoint	6	8	11
6	Lingthing2	71	91	115
7	OSULeaf	324	450	587
8	SwedishLeaf	185	256	339
9	Synthetic_control	13.6	18	23.7
10	Trace	27	38	51

5.5.4 ΠΕΙΡΑΜΑΤΙΚΑ ΑΠΟΤΕΛΕΣΜΑΤΑ : ΣΥΓΚΡΙΣΗ

Ας προχωρήσουμε στο σημείο αυτό σε κάποιες πιο ουσιαστικές συγκρίσεις. Προηγουμένως εκτελέσαμε τα πειράματά μας για κάποιες τιμές του Sakoe/Chiba ας δούμε τώρα την κυρίως σύγκριση δηλαδή του απλού DTW με τον Sakoe/Chiba, για να ελέγξουμε κατά πόσο επηρεάζεται ο DTW από τον περιορισμό που του επιβάλλει ο αλγόριθμος Sakoe/Chiba. Επιλέξαμε να συγκρίνουμε με τον περιορισμό r = 5.

Παρατηρώντας τον συγκεντρωτικό πίνακα των DTW και Sakoe/Chiba (Πίνακας 5.8) συμπεραίνουμε πως τα αποτελέσματα και για τα ποσοστά σφάλματος αλλά και για τους χρόνους απόκρισης είναι σαφώς καλύτερα χρησιμοποιώντας τον αλγόριθμο που περιορίζει το εύρος αναζήτησης τιμών. Πιο συγκεκριμένα σε έξι από τα δέκα datasets ο απλός DTW παρουσιάζει μεγαλύτερα ποσοστά σφάλματος από τον Sakoe/Chiba και σε ένα dataset το ποσοστό παραμένει ίδιο. Η διαφορά αυτή κυμαίνεται από 2,6 έως 12 ποσοστιαίες μονάδες. Παρόμοια συμπεριφορά παρουσιάζει ο DTW και όσο αναφορά τους χρόνους απόκρισης. Σε ποσοστό εκατό τις εκατό (δέκα στα δέκα datasets) οι χρόνοι του

Sakoe/Chiba είναι σαφώς μικρότεροι/καλύτεροι από τους αντίστοιχους του DTW. Ο χρόνος που απαιτείται στον DTW είναι από 3,3 έως 5,5 φορές μεγαλύτερος από τον αντίστοιχο του Sakoe/Chiba στα συγκεκριμένα datasets.

Πίνακας 5.8– 1-NN DTW - Sakoe/Chiba – ποσοστά σφάλματος(%) / χρόνοι απόκρισης (sec)

A/A	**ΣΥΝΟΛΟ**	**DTW**		**Sakoe/Chiba r = 5**	
		Ποσοστό σφάλματος (%)	**Χρόνος (sec)**	**Ποσοστό σφάλματος (%)**	**Χρόνος (sec)**
1	50words	31.0	877	23.0	217
2	CBF	0.3	27	2.0	6.8
3	ECG200	23.0	5.6	11.0	1.5
4	FaceFour	17.0	22	13.6	4
5	GunPoint	9.3	9.6	3.0	2.6
6	Lingthing2	13.1	128	15.0	30.5
7	OSULeaf	40.9	706	42.0	128
8	SwedishLeaf	20.8	293	18.2	76
9	Synthetic_control	7.0	20	1.3	6
10	Trace	0.0	54	0.0	11

Αφού στην προηγούμενη ενότητα εκτελέσαμε τα πειράματα μας και είδαμε τα αποτελέσματα για κάποιες βασικές τιμές του αλγορίθμου Itakura, προχωρήσαμε σε μια ακόμα κυρίως σύγκριση δηλαδή του απλού DTW με τον Itakura, για να ελέγξουμε κατά πόσο επηρεάζεται ο DTW από τον περιορισμό που του επιβάλλει ο αλγόριθμος αυτός. Επιλέξαμε να συγκρίνουμε με τον περιορισμό r = 2.

Πίνακας 5.9 – 1-NN DTW - Itakura – ποσοστά σφάλματος/ χρόνοι απόκρισης

		DTW		Itakura r = 2	
Α/Α	**ΣΥΝΟΛΟ**	**Ποσοστό σφάλματος (%)**	**Χρόνος (sec)**	**Ποσοστό σφάλματος (%)**	**Χρόνος (sec)**
1	50words	31.0	877	26.0	713
2	CBF	0.3	27	0.4	23
3	ECG200	23.0	5.6	18.0	5
4	FaceFour	17.0	22	16.0	13.6
5	GunPoint	9.3	9.6	10.0	8
6	Lingthing2	13.1	128	13.1	91
7	OSULeaf	40.9	706	40.1	450
8	SwedishLeaf	20.8	293	18.6	256
9	Synthetic_control	7.0	20	1.0	18
10	Trace	0.0	54	0.0	38

Από τον συγκεντρωτικό πίνακα (Πίνακας 5.9) των DTW και Itakura αλγορίθμων συμπεραίνουμε πως τα αποτελέσματα που προέκυψαν όσο αναφορά τα ποσοστά σφάλματος κατηγοριοποίησης είναι μικρότερα/καλύτερα για τον Itakura σε έξι από τα 10 datasets και σε δυο datasets τα ποσοστά παραμένουν αμετάβλητα σε σχέση με τον αλγόριθμου του απλού DTW. Όσο αναφορά το χρόνο απόκρισης των δυο αλγορίθμων προκύπτει πως σε ποσοστό εκατό τις εκατό (δέκα στα δέκα datasets) οι χρόνοι του Itakura είναι σαφώς μικρότεροι/καλύτεροι από τους αντίστοιχους του DTW. Ο χρόνος που απαιτείται στον DTW είναι από 1,1 έως 1,6 φορές μεγαλύτερος από τον αντίστοιχο του Itakura στα συγκεκριμένα datasets.

Από τους δυο ως τώρα συγκεντρωτικούς πίνακες διαπιστώνουμε πως οι αλγόριθμοι οι οποίοι περιορίζουν το εύρος αναζήτησης τιμών (Sakoe/Chiba και Itakura) παρουσιάζονται ως πιο αποτελεσματικοί τόσο στα ποσοστά σφάλματος κατά τη διάρκεια της κατηγοριοποίησης όσο και στους χρόνους απόκρισης τους σε σχέση με τα αποτελέσματα του απλού DTW αλγορίθμου.

Ας εξετάσουμε όμως ανάμεσα σε Sakoe/Chiba και Itakura ποιος είναι πιο ικανοποιητικός σε σχέση με τον απλό αλγόριθμο του DTW.

Πίνακας 5.10– 1-NN DTW - Sakoe/Chiba - Itakura - ποσοστά σφάλματος (%)

A/A	ΣΥΝΟΛΟ	DTW	Sakoe/Chiba r = 5	Itakura r = 2
1	50words	31.0	23.0	26.0
2	CBF	0.3	2.0	0.4
3	ECG200	23.0	11.0	18.0
4	FaceFour	17.0	13.6	16.0
5	GunPoint	9.3	3.0	10.0
6	Lingthing2	13.1	15.0	13.1
7	OSULeaf	40.9	42.0	40.1
8	SwedishLeaf	20.8	18.2	18.6
9	Synthetic_control	7.0	1.3	1.0
10	Trace	0.0	0.0	0.0

Πίνακας 5.11– 1-NN DTW - Sakoe/Chiba - Itakura – χρόνοι απόκρισης (sec)

A/A	ΣΥΝΟΛΟ	DTW	Sakoe/Chiba r = 5	Itakura r = 2
1	50words	877	217	713
2	CBF	27	6.8	23
3	ECG200	5.6	1.5	5
4	FaceFour	22	4	13.6
5	GunPoint	9.6	2.6	8
6	Lingthing2	128	30.5	91
7	OSULeaf	706	128	450
8	SwedishLeaf	293	76	256
9	Synthetic_control	20	6	18
10	Trace	54	11	38

Από τη σύγκριση των ποσοστών σφάλματος ανάμεσα στους αλγόριθμους του DTW, Sakoe/Chiba, Itakura (Πίνακας 5.10)παρατηρούμε πως τα μικρότερα ποσοστά τα συναντούμε σε πέντε από τα δέκα datasets για τον Sakoe/Chiba, σε δύο από τα δέκα datasets για τον Itakura, σε ένα για τον DTW, σε ένα dataset του οποίου το ποσοστό είναι ίδιο για τον DTW και τον Itakura και ένα dataset του οποίου το ποσοστό παραμένει ίδιο και στις τρεις περιπτώσεις. Τα ποσοστά σφάλματος για τον Sakoe/Chiba, στα datasets στα οποία υπερέχει, είναι χαμηλότερα από 2,6 έως 12 ποσοστιαίες μονάδες, ενώ στα datasets στα οποία μειονεκτεί είναι υψηλότερα από 0,3 έως 1,9 ποσοστιαίες μονάδες.

Όσο αναφορά τους χρόνους απόκρισης (Πίνακας 5.11) εδώ τα πράγματα είναι πιο ξεκάθαρα. Ο αλγόριθμος του Sakoe/Chiba υπερέχει των άλλων δυο καθώς έχει τους μικρότερους χρόνους απόκρισης και μάλιστα και με αρκετή διαφορά. Στα συγκεκριμένα datasets, ο χρόνος που απαιτείται στον DTW είναι από 3,3 έως 5,5 φορές μεγαλύτερος από τον αντίστοιχο του Sakoe/Chiba, ενώ ο χρόνος που απαιτείται στον Itakura είναι από 3 έως 3,5 φορές μεγαλύτερος από τον αντίστοιχο του Sakoe/Chiba.

5.5.5 ΣΥΜΠΕΡΑΣΜΑΤΑ

Με βάση όλα τα παραπάνω πειράματα συμπεραίνουμε γενικότερα πως όταν χρησιμοποιούμε τον απλό αλγόριθμο του DTW χάνουμε τόσο σε ποσοστά σφάλματος όσο και σε χρόνο. Αντίθετα με τους αλγόριθμους που μειώνουν το εύρος αναζήτησης τιμών του DTW μας παρουσιάζονται καλύτερα ποσοστά σφάλματος αλλά και χρόνου. Όσο αναφορά για το ποιού είδους περιορισμό θα μας ήταν προτιμότερο να χρησιμοποιήσουμε, ίσως από την άποψη χρόνου αυτός θα ήταν ο Sakoe/Chiba καθώς οι μικρότεροι χρόνοι εκτέλεσης των πειραμάτων που παρουσιάζει μπορούν να αποτελέσουν σημαντικό κίνητρο για όταν έχουμε να δουλέψουμε σε μεγάλες βάσεις δεδομένων. Συν το γεγονός ότι τα ποσοστά του Sakoe/Chiba δεν διαφέρουν πολύ από τα αντίστοιχα του Itakura και μάλιστα είναι μικρότερα/καλύτερα στην πλειοψηφία των datasets.

ΕΠΙΛΟΓΟΣ

Οι εξελίξεις στον τομέα της εξόρυξης γνώσης μέσω ανάλυσης χρονοσειρών και οι ευρεία χρήση των τελευταίων σε διάφορους και σημαντικούς τομείς της σύγχρονης ζωής, απαιτεί την ανάπτυξη νέων τεχνικών και μεθόδων ώστε να αναλυθούν αποτελεσματικά και με ακρίβεια. Για την επίτευξη αυτού του στόχου πρέπει να λαμβάνονται κάθε φορά υπόψη τα ιδιαίτερα χαρακτηριστικά των χρονοσειρών.

Δυο σημαντικά θέματα που προκύπτουν κατά την εφαρμογή τεχνικών εξόρυξης γνώσης είναι η μετατροπή των αρχικών δεδομένων ώστε να πραγματοποιηθεί μείωση της διαστατικότητάς τους και ο ορισμός ενός μέτρου απόστασης για τον εντοπισμό όμοιων χρονοσειρών. Μ΄ αυτόν τον τρόπο έχουμε ταχύτερα και πιο αξιόπιστα αποτελέσματα. Με βάση τα όσα παρουσιάστηκαν σ΄ αυτή την εργασία αλλά και με τα πειράματα που πραγματοποιήθηκαν συμπεραίνουμε γενικότερα τη σημαντικότητα επιλογής μιας τεχνικής εύρεσης όμοιων χρονοσειρών αλλά και ειδικότερα για το ποια τεχνική θα χρησιμοποιηθεί ανάλογα με τα χαρακτηριστικά των δεδομένων μας. Ιδιαίτερο ρόλο στην επιλογή μας παίζουν τόσο τα ποσοστά σφάλματος όσο και οι χρόνοι απόκρισης της εκάστοτε τεχνικής.

Το γεγονός ότι τα τελευταία χρόνια έχει γίνει πολλή έρευνα γύρω από τον τομέα της διαδικασίας εξόρυξης γνώσης μέσω ανάλυσης χρονοσειρών από αρκετές επιστημονικές κοινότητες πέραν της πληροφορικής, μας δείχνει πως οδηγούμαστε σε μια στενότερη διεπιστημονική συνεργασία στον τομέα αυτό με απώτερο σκοπό την εύρεση πιο αποτελεσματικών και αξιόπιστων τεχνικών .

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Ζήσος Ι. (2006). «Στατικές και δυναμικές μηχανές νευρωνικών δικτύων για πρόβλεψη μετεωρολογικών παραμέτρων» Πτυχιακή Εργασία. Πολυτεχνείο Κρήτης. Σελ.1-2

[2] Παπανίκου Αναστασία (2008). «Time Series Data Mining» Πτυχιακή Εργασία. Πανεπιστήμιο Μακεδονίας. Σελ. 50-59,

[3] Agrawal, R., Faloutsos, C., Swami, A. (1993). “Efficient Similarity Search in Sequence Databases”, In Proceedings of the 4th International Conference on Foundations of Data Organization and Algorithms (FODO), Chicago, Illinois, October 1-15, pp 69-84.

[4] Agrawal, R., Lin K. I., Sawhney, H. S., Shim, K. (1995). “Fast Similarity Search in the Presence of Noise, Scaling and Translation in Times-Series Databases”, In Proceedings of 21th International Conference on Very Large Databases, Zurich, Switzerland, September, pp 490-500.

[5] Berndt D. J., Clifford J., “Using Dynamic Time Warping to Find Patterns in Time Series”, KDD Workshop 1994, pp.359-370, Seattle, WA (USA), July 2004.

[6] Chan, K. & Fu, W., (1999). “Efficient time series matching by wavelets”, *In the Proceedings of the 15th IEEE International Conference on Data Engineering*. Sydney, Australia, March 23-26, pp.126-133.

[7] Faloutsos, C., & Lin, K., (1995). “Fast map: A Fast Algorithm for Indexing, Data-Mining and Visualization of Traditional and Multimedia Datasets”, In Proceedings ACM SIGMOD Conference, May, pp 163-174.

[8] Faloutsos, C., Ranganathan, M., Manolopoulos, Y. (1994). “Fast Subsequence Matching in Time-Series Databases”, In Proceedings of the ACM SIGMOD Conference on Management of Data, Mineapolis, May, pp 419-429.

[9] Fayyad, U. M., Gregory Piatetsky-Shapiro, and Padhraic Smyth (1996). " From data mining to knowledge discovery in databases", October, pages 37-41.

[10] Ge X. and Smyth P.. Deformable markov model templates for time-series pattern matching. In Proc ACM SIGKDD, 2000.

[11] Gupta, A., Vadhavkar, S., Au, S.C. "Importance Of Data Mining for Electronic Commerce". Ανάκτηση *από* *http://www.neural.uom.gr/Documents/Data Mining/Importance of Data Mining for Electronic Commerce.pdf* στις 10/9/2010.

[12] Kamath, C. "An Introduction to Scientific Data Mining". Ανάκτηση από http://www.ipam.ucla.edu/publications/sdm2002/sdm2002_1240.pdf στις10/9/2010.

[13] Keogh, E. J. (1997). "Fast similarity search in the presence of longitudinal scaling in time series databases", In Proceedings of the 9th International Conference on Tools with Artificial Intelligence, IEEE Press, pp 578-584.

[14] Keogh, E. J. (2001). "A Tutorial on Indexing and Mining Time Series Data", In Proceedings of IEEE International Conference on Data Mining, San Jose, November 29.

[15] Keogh, E. J., (2002). "Exact Indexing Of Dynamic Time Warping", In Proceedings of IEEE International Conference on Data Mining.

[16] Keogh, E. J. & Ratanamahatana (2004). "Exact Indexing Of Dynamic Time Warping", In Proceedings of IEEE International Conference on Data Mining. Ανάκτηση από http://www.cs.ucr.edu/~eamonn/KAIS_2004_warping.pdf στις 20/09/2010 .

[17] Keogh, E,. Chakrabarti, K,. Pazzani, M., Mehrotra (2000). "Dimensionality Reduction For Fast Similarity Search in Large Time Series Databases", Journal of Knowledge and Information Systems, pp 263-286.

[18] Keogh, E., Chakrabarti, K., Pazzani, M., Mehrotra, S. (2001). “Locally Adaptive Dimensionality Reduction for Indexing Large Time Series Databases”, In Proceedings of ACM SIGMOD Conference on Management of Data, Santa Barbara, CA, May 21-24, pp 151-162.

[19] Keogh, E., Chu, S., Hart, D., Pazzani, M. (2001). “An Online Algorithm for Segmenting Time Series”, *In Proceedings of IEEE International Conference on Data Mining,* pp 289-296.

[20] Keogh, E., & Pazzani, M. (1998). “An Enhanced Representation of Time Series which Allows Fast and Accurate Classification, Clustering and Relevance Feedback”, *In Proceedings of the 4th International Conference of Knowledge Discovery and Data Mining*, AAAI Press, New York, NY, Aug 27-31, pp 239-243.

[21] Keogh, E., & Pazzani, M. (1999). “Relevance Feedback Retrieval of Time Series Data”, In Proceedings of the 22th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval.

[22] Keogh, E., & Pazzani, M. (2000). “A Simple Dimensionality Reduction Technique for Fast Similarity Search in Large Time Series Databases”, *In Proceedings of the 4th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, Kyoto, Japan.

[23] Keogh, E., & Smyth, P. (1997). “A Probabilistic Approach to Fast Pattern Matching in Time Series Databases”, *In Proceedings of the 3rd International Conference of Knowledge Discovery and Data Mining*, AAAI Press, pp 24-20.

[24] Lin, J., Keogh, E., Lonardi, S. Chiu, B. (2003). “A Symbolic Representation of Time Series, with Implications for Streaming Algorithms”, *In Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery,* San Diego, CA, June 13.

[25] Lin, J., Keogh, E., Lonardi, S. Patel, P. (2002). “Finding Motifs in Time Series”, In Proceedings of the 2nd Workshop on Temporal Data Mining, at

the 8th ACM SIGKDD Conference on Knowledge Discovery and Data Mining", Edmonton, Alberta, Canada, July 23-26, pp. 53-68.

[26] Lin J., Keogh E., Lonardi S., (2005). *"*Visualizing and Discovering Nontrivial Patterns In Large Time Series Databases*"*, Information Visualization, Vol 4, No. 2

[27] Negi, Σ. & Bansal, V. "Time Series: Similarity Search and its Applications". Ανάκτηση από http://www.iitk.ac.in/ime/veena/PAPERS/icsci05.pdf στις 15/9/2010.

[28] Margaret H. Dunham (2004)."Data Mining Introductory and Advanced Topics".1end, Μετάφραση από Βερύκιος Β. & Θεοδωρίδης Γ. Εκδόσεις Νέων Τεχνολογίων, σελ.6-9.

[29] Ramakrishnan, R. (2002). "Database Management Systems". 2nd Edition, Μετάφραση από τους Δέρβο Δ. & Ευαγγελίδη Γ., Εκδόσεις Τζιόλα, σελ.1065-1095.

[30] B. K. Yi and C. Faloutsos, "Fast time sequence indexing for arbitrary Lp Norms", In Proc. VLDB-2000, Cairo, Egypt, September 2000, pp.385-394.

[31] Yi, B.-K., Jagadish, H. V., Faloutsos, C. (1998). "Efficient Retrieval of Similar Time Sequences Under Time Warping", In 3rd IEEE Knowledge and Data Engineering Exchange Workshop (ICDE).

[32] Vlachos M.(2004). "Similarity and Indexing in Multidimensional Spaces". Univercity Of California, pp 9-38.

[33] Witten I. and Frank E. (2005). "Data Mining Practical Machine Learning Tools and Techniques". 2end, pp 3-10

ΗΛΕΚΤΡΟΝΙΚΕΣ ΠΗΓΕΣ

- http://en.wikipedia.org/wiki/Time_series
- http://www.stats.gla.ac.uk/steps/glossary/time_series.html
- http://www.abeel.be/taxonomy/term/1
- http://jmlr.csail.mit.edu/papers/volume10/abeel09a/abeel09a.pdf
- http://java-ml.sourceforge.net/

ΠΑΡΑΡΤΗΜΑ Α.

Α1. DTW Similarity Measure Code – Java-ML (Modified)

```
/**
 * This file is part of the Java Machine Learning Library
 * Copyright (c) 2006-2010, Thomas Abeel
 * Project: http://java-ml.sourceforge.net/
 */
package net.sf.javaml.distance.dtw;

import java.util.ArrayList;
import net.sf.javaml.core.Instance;
import net.sf.javaml.distance.AbstractSimilarity;

/**
 * A similarity measure based on "Dynamic Time Warping". The DTW
distance is
 * mapped to a similarity measure using f(x)= 1 - (x / (1 + x)).
Feature weights
 * are also supported.
 *
 * @author Piotr Kasprzak
 * @author Thomas Abeel
 *
 */
public class DTWSimilarity extends AbstractSimilarity {

    private static final long serialVersionUID =-8898553450277603746L;

    private double pointDistance(int i, int j, double[] ts1,
                                                    double[]ts2) {

        double diff = ts1[i] - ts2[j];
        return (diff * diff);

    }

    private double distance2Similarity(double x) {
            return (1.0 - (x / (1 + x)));
    }


@Override
    public double measure(Instance x, Instance y) {

        ArrayList<Double> l1 = new ArrayList<Double>();
        ArrayList<Double> l2 = new ArrayList<Double>();
        int i, j;

/** Filter NaNs */
        for (i = 0; i < x.noAttributes(); i++) {
            double value = x.value(i);

            if (!Double.isNaN(value)) {
                l1.add(value);
            }
        }
```

```
      for (i = 0; i < y.noAttributes(); i++) {
            double value = y.value(i);
            if (!Double.isNaN(value)) {
                l2.add(new Double(value));
            }
        }

/** Transform the examples to vectors */
        double[] ts1 = new double[l1.size()];
        double[] ts2 = new double[l2.size()];

        for (i = 0; i < ts1.length; i++) {
            ts1[i] = l1.get(i);
        }

        for (i = 0; i < ts2.length; i++) {
            ts2[i] = l2.get(i);
        }

/** Build a point-to-point distance matrix */
        double[][] dP2P = new double[ts1.length][ts2.length];
        for (i = 0; i < ts1.length; i++) {

            for (j = 0; j < ts2.length; j++) {
                dP2P[i][j] = pointDistance(i, j, ts1, ts2);
            }
        }

/** Check for some special cases due to ultra short time series */
        if (ts1.length == 0 || ts2.length == 0) {
             return Double.NaN;
        }

        if (ts1.length == 1 && ts2.length == 1) {
             return distance2Similarity(Math.sqrt(dP2P[0][0]));
        }

/** Build the optimal distance matrix using a dynamic programming
approach*/

        double[][] D = new double[ts1.length][ts2.length];

        D[0][0] = dP2P[0][0]; // Starting point

        for (i = 1; i < ts1.length; i++) { // Fill the first column
                        //of our distance matrix with optimal values

                 D[i][0] = dP2P[i][0] + D[i - 1][0];
   }
        if (ts2.length == 1) { // TS2 is a point
            double sum = 0;

            for (i = 0; i < ts1.length; i++) {
                sum += D[i][0];
            }
            return distance2Similarity(Math.sqrt(sum) / ts1.length);
```

```
        for (j = 1; j < ts2.length; j++) { // Fill the first row of our
                                           // distance matrix with optimal values
            D[0][j] = dP2P[0][j] + D[0][j - 1];
        }

        if (ts1.length == 1) { // TS1 is a point
            double sum = 0;
            for (j = 0; j < ts2.length; j++) {
                sum += D[0][j];
            }
            return distance2Similarity(Math.sqrt(sum) / ts2.length);
        }

        for (i = 1; i < ts1.length; i++) { // Fill the rest

            for (j = 1; j < ts2.length; j++) {
                double[] steps = { D[i - 1][j - 1], D[i - 1][j],
                                                    D[i][j - 1] };
                double min = Math.min(steps[0], Math.min(steps[1],
                                                      steps[2]));
                D[i][j] = dP2P[i][j] + min;

            }
        }

        double d= D[ts1.length - 1][ts2.length - 1];
        return distance2Similarity(Math.sqrt(d) );
    }


}
```

A2. Sakoe / Chiba Similarity Measure Code

```
package net.sf.javaml.distance.dtw;

import net.sf.javaml.core.Instance;
import net.sf.javaml.distance.AbstractSimilarity;

public class Sakoe_ChibaDTW extends AbstractSimilarity {

    private static final long serialVersionUID =-8898553450277603746L;

    private double r, rTemp;

    public Sakoe_ChibaDTW (double radius){
      rTemp = radius;
    }

    private double distance2Similarity(double x) {
        return (1.0 - (x / (1 + x)));
    }


@Override
    public double measure(Instance x, Instance y) {

      int m = x.values().size();
      int n = y.values().size();

      r =
      Math.round(Math.min(rTemp*(double)n/100.0,rTemp*(double)m/100));

      double pin[][] = new double[m+1][n+1];

      for(int i=0; i<pin.length; i++)
       for(int j=0; j<pin[i].length; j++)
         pin[i][j] = Double.POSITIVE_INFINITY;

      for(int i=1; i<=m; i++)
        for(int j = (int)Math.max(1, i-r); j<=Math.min(n, i+r); j++)
            if( i == 1 && j == 1)
              pin[i][j] = (x.value(i-1) - y.value(j-1))*(x.value(i-1)
                                                    - y.value(j-1));
              else
               pin[i][j] = (x.value(i-1) - y.value(j-1))*(x.value(i-1)
                             - y.value(j-1)) + Math.min(pin[i][j-1],
                               Math.min(pin[i-1][j],pin[i-1][j-1]));


        return distance2Similarity(Math.sqrt(pin[m][n]));
    }

}
```

A3. Itakura Similarity Measure Code

```
package net.sf.javaml.distance.dtw;

import net.sf.javaml.core.Instance;
import net.sf.javaml.distance.AbstractSimilarity;

public class ItakuraDTW extends AbstractSimilarity {

    private static final long serialVersionUID =-8898553450277603746L;

    private double radius,r;
    public double[][] dP2P;

    public ItakuraDTW(int radius) {
            super();
            this.radius = radius;
            r = radius;
    }

    public ItakuraDTW(double radius) {
            super();
            this.radius = radius;
            r = radius;
    }

    private double distance2Similarity(double x) {
        return (1.0 - (x / (1 + x)));
    }

@Override

    public double measure(Instance x, Instance y) {
        ArrayList<Double> l1 = new ArrayList<Double>();
        ArrayList<Double> l2 = new ArrayList<Double>();
        int i, j;


        for (i = 0; i < x.noAttributes(); i++) {
            double value = x.value(i);
            if (!Double.isNaN(value)) {
                l1.add(value);
            }
        }

        for (i = 0; i < y.noAttributes(); i++) {
            double value = y.value(i);
            if (!Double.isNaN(value)) {
                l2.add(new Double(value));
            }
        }

        double[] ts1 = new double[l1.size()];
        double[] ts2 = new double[l2.size()];

        for (i = 0; i < ts1.length; i++) {
            ts1[i] = l1.get(i);
        }

        for (i = 0; i < ts2.length; i++) {
            ts2[i] = l2.get(i);
        }
```

```
dP2P = new double[ts1.length+2][ts2.length+2];

for(i=0; i<dP2P.length; i++)
      for(j=0; j<dP2P[i].length; j++)
            dP2P[i][j] = Double.POSITIVE_INFINITY;


  if(ts1.length > 3 && ts2.length > 3) {
      dP2P[2][2] = Math.pow((ts1[0] - ts2[0]), 2);
      dP2P[3][3] = Math.pow((ts1[1] - ts2[1]), 2);
  }

  double m = (double)ts1.length;
  double n = (double)ts2.length;

  for(i=4; i<=Math.round((r/(r*r-1)) * (n-m/r)); i++)
   for(j=(int)Math.round((1./r)*i); j<=Math.round(r*i); j++ )
            try{
             dP2P[i][j] = Math.pow(ts1[i-1]-ts2[j-1], 2) +
               Math.min(dP2P[i-1][j-1], Math.min( dP2P[i-1][j],
                                                  dP2P[i][j-1]));
            }
            catch (ArrayIndexOutOfBoundsException e) {
                        System.out.println("ArrayIndex 1 : i="
                                           + i + "  j=" + j);
            }



 for(i=(int)(Math.round((r/(r*r-1)) * (n-m/r))+1);
                  i<=Math.round((r/(1-r*r)) * (n-r*m)); i++)
  for(j=(int)Math.round(i/r); j<=Math.round(i/r+(n-m/r)); j++)
      try{
             dP2P[i][j] = Math.pow(ts1[i-1]-ts2[j-1], 2) +
                  Math.min(dP2P[i-1][j-1], Math.min( dP2P[i-
                                          1][j], dP2P[i][j-1]));
         }
       catch (ArrayIndexOutOfBoundsException e) {
                        System.out.println("ArrayIndex 2 : i="
                                             + i + "  j=" + j);
       }



 for(i=(int)(Math.round((r/(1-r*r)) * (n-r*m))+1); i<=m+1; i++)
   for(j=(int)Math.round(r*i+(n-r*m)); j<=Math.round(i/r+(n-
                                                  m/r)); j++)
      try{
            dP2P[i][j] = Math.pow(ts1[i-1]-ts2[j-1], 2) +
                  Math.min(dP2P[i-1][j-1], Math.min( dP2P[i-
                                          1][j], dP2P[i][j-1]));
        }
       catch (ArrayIndexOutOfBoundsException e) {
                        System.out.println("ArrayIndex 3 : i="
                                             + i + "  j=" + j);
       }
```

```
dP2P[(int)m+1][(int)n+1] = Math.pow(ts1[(int)(m-1)]-ts2[(int)(n-1)],
                        2) + Math.min(dP2P[(int)m][(int)n], Math.min(
                    dP2P[(int)m+1][(int)n], dP2P[(int)m][(int)n+1]));



 return distance2Similarity(Math.sqrt((dP2P[ts1.length-1][ts2.length-
                                                          1])));

 }

}
```

A4. OneNearestNeighbor Code – Java-ML

```
package test1;
import java.io.File;
import java.io.IOException;

import net.sf.javaml.classification.Classifier;
import net.sf.javaml.classification.KNearestNeighbors;
import net.sf.javaml.core.Dataset;
import net.sf.javaml.core.Instance;
import net.sf.javaml.distance.dtw.DTWSimilarity;
import net.sf.javaml.tools.data.FileHandler;
//import net.sf.javaml.distance.dtw.Sakoe_ChibaDTW;
//import net.sf.javaml.distance.dtw.ItakuraDTW;
//import net.sf.javaml.distance.fastdtw.FastDTW;

public class TestOneNN {

      public static void main(String[] args) throws IOException {
            /* Load a data set */

        for (int i = 0; i < 5; i++) {
             String filePreffix = "G:/ptixiakh!/datasets/ECG200";
             long startTime = System.currentTimeMillis();
             Dataset data = FileHandler.loadDataset(new File(
                                       filePreffix + "_TRAIN"),0, " ");


            Classifier knn = new KNearestNeighbors(1, new
                                                    DTWSimilarity());
            knn.buildClassifier(data);



            Dataset dataForClassification = FileHandler.loadDataset(
                              new File(filePreffix + "_TEST"),0, " ");


                  int correct = 0, wrong = 0;

                  /* Classify all instances and check with the correct
                                                     class values */
          for (Instance inst : dataForClassification) {
             Object predictedClassValue = knn.classify(inst);
             Object realClassValue = inst.classValue();
                  if (predictedClassValue.equals(realClassValue))
                        correct++;
                  else
                        wrong++;
          }
             System.out.println("Correct predictions  " +  correct);
             System.out.println("Wrong predictions " + wrong);
             System.out.println(System.currentTimeMillis()- startTime);
      }

    } //end of main

} //end of class
```