ΕΙΣΑΓΩΓΗ ΣΤΟΝ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟ (2016-17)

Εργασία 3

Η εργασία αυτή εντάσσεται στις περιοχές της Επιστήμης των Υπολογιστών που ονομάζονται *μηχανική μάθηση* (machine learning) και *εξόρυξη δεδομένων* (data mining), για τις οποίες θα συζητήσετε εκτενέστερα σε μαθήματα επιλογής επόμενων εξαμήνων. Αντικείμενο της εργασίας είναι η *κατηγοριοποίηση χρονοσειρών* (time-series classification), εφαρμόζοντας τον αλγόριθμο DTW της *δυναμικής στρέβλωσης του χρόνου* (dynamic time warping) για την εύρεση του βαθμού ομοιότητας δύο χρονοσειρών.

Μία χρονοσειρά είναι μία ακολουθία τιμών ενός μεγέθους σε διαδοχικές ισαπέχουσες χρονικές στιγμές. Για παράδειγμα, για δεδομένα χρονικά διαστήματα, οι μέγιστες ημερήσιες θερμοκρασίες σε μία περιοχή, οι ημερήσιες τιμές κλεισίματος μίας μετοχής, οι τιμές της στιγμιαίας ταχύτητας ενός κινητού μετρημένες ανά δευτερόλεπτο, κλπ. είναι χρονοσειρές. Επίσης, κάθε χρονοσειρά από ένα σύνολο χρονοσειρών συγκεκριμένου τύπου, π.χ. χρονοσειρές μετοχών, μπορεί να ανήκει σε κάποια κατηγορία/κλάση (class), όπως τραπεζικές μετοχές, μετοχές κλάδου ενέργειας, μετοχές εταιρειών τηλεπικοινωνιών, κλπ.

Έστω δύο χρονοσειρές A και B που έχουν ως στοιχεία τα a_i $(1 \leq i \leq n)$ και b_j $(1 \leq j \leq m)$, αντίστοιχα. Συμβολίζουμε με A_i $(1 \leq i \leq n)$ τη χρονοσειρά που περιλαμβάνει τα i πρώτα στοιχεία της A. Ομοίως, συμβολίζουμε με B_j $(1 \leq j \leq m)$ τη χρονοσειρά που περιλαμβάνει τα j πρώτα στοιχεία της B. Η *απόσταση DTW* των χρονοσειρών A_i και B_j συμβολίζεται με $dtw(A_i, B_j)$ και ορίζεται στη συνέχεια, θεωρώντας ότι $d(a_i, b_j) = (a_i - b_j)^2$ και ότι το $min\{x, y, z\}$ είναι το ελάχιστο των x, y και z.

$$dtw(A_i, B_j) = \begin{cases} d(a_i, b_j) + min\{dtw(A_{i-1}, B_j), dtw(A_i, B_{j-1}), dtw(A_{i-1}, B_{j-1})\} & \text{αν } 1 < i \leq n, \ 1 < j \leq m \\ d(a_1, b_j) + dtw(A_1, B_{j-1}) & \text{αν } i = 1, \ 1 < j \leq m \\ d(a_i, b_1) + dtw(A_{i-1}, B_1) & \text{αν } 1 < i \leq n, \ j = 1 \\ d(a_1, b_1) & \text{αν } i = 1, \ j = 1 \end{cases}$$

Η απόσταση DTW των χρονοσειρών A και B ισούται, προφανώς, με:

$$dtw(A, B) = dtw(A_n, B_m)$$

Αναδρομική υλοποίηση (10%)

Γράψτε ένα πρόγραμμα C το οποίο να διαβάζει από την είσοδο δύο χρονοσειρές και να υπολογίζει, με τη βοήθεια μίας αναδρομικής συνάρτησης `dtwrec`, που θα βασίζεται στο σκεπτικό που αναφέρθηκε προηγουμένως, την απόσταση DTW των δύο χρονοσειρών. Αρχικά, το πρόγραμμα να διαβάζει σε μία γραμμή τα μήκη των δύο χρονοσειρών και στις επόμενες δύο γραμμές τα στοιχεία των χρονοσειρών αυτών (ως `double`). Στο πρόγραμμά σας δεν επιτρέπεται να ορίσετε άλλους πίνακες εκτός από αυτούς που χρειάζονται για τη φύλαξη των στοιχείων των χρονοσειρών.

Αν το εκτελέσιμο πρόγραμμα που θα κατασκευάσετε τελικά ονομάζεται "`distdtwrec`", ενδεικτικές εκτελέσεις του φαίνονται στη συνέχεια.

```
$ hostname
linux29
$
$ time ./distdtwrec
3 5
1.0 2.0 0.0
```

```
-2.0 0.0 1.0 3.0 1.0
Distance of time series is 12.000000
0.000u 0.000s 0:10.97 0.0%      0+0k 0+0io 0pf+0w
$
$ time ./distdtwrec
6 10
1.0 2.0 0.0 1.0 2.0 0.0
-2.0 0.0 1.0 3.0 1.0 -2.0 0.0 1.0 3.0 1.0
Distance of time series is 18.000000
4.004u 0.000s 0:13.94 28.6%     0+0k 0+0io 0pf+0w
$
$ time ./distdtwrec
8 10
1.0 2.0 0.0 1.0 2.0 0.0 1.0 -1.0
-2.0 0.0 1.0 3.0 1.0 -2.0 0.0 1.0 3.0 1.0
Distance of time series is 22.000000
83.937u 0.004s 1:47.24 78.2%    0+0k 0+0io 0pf+0w
$
```

Επαναληπτική υλοποίηση — Δυναμικός προγραμματισμός (15%)

Παρατηρήστε στις ενδεικτικές εκτελέσεις του προγράμματος "`distdtwrec`" την απότομη αύξηση του χρόνου εκτέλεσης όταν τα μήκη των χρονοσειρών δεν είναι πλέον πάρα πολύ μικρά. Έχετε κάποια εξήγηση γι΄ αυτό; Για την αντιμετώπιση αυτού του προβλήματος, δώστε μία εναλλακτική υλοποίηση του υπολογισμού της απόστασης DTW δύο χρονοσειρών μέσω μίας συνάρτησης `dtwdp` η οποία θα λειτουργεί επαναληπτικά με τον εξής τρόπο. Με τη βοήθεια ενός δισδιάστατου πίνακα θα υπολογίζει και θα αποθηκεύει όλες τις τιμές των $dtw(A_i, B_j)$, για κάθε i και j, μία φορά την κάθε μία. Η τεχνική αυτή, που ουσιαστικά υλοποιεί μία αναδρομική σχέση, αποφεύγοντας τη χρήση αναδρομής, ονομάζεται *δυναμικός προγραμματισμός* (dynamic programming).

Αν το εκτελέσιμο πρόγραμμα που θα κατασκευάσετε τελικά ονομάζεται "`distdtwdp`", ενδεικτικές εκτελέσεις του φαίνονται στη συνέχεια.

```
$ hostname
linux29
$
$ time ./distdtwdp
3 5
1.0 2.0 0.0
-2.0 0.0 1.0 3.0 1.0
Distance of time series is 12.000000
0.000u 0.000s 0:14.46 0.0%      0+0k 0+0io 0pf+0w
$
$ time ./distdtwdp
6 10
1.0 2.0 0.0 1.0 2.0 0.0
-2.0 0.0 1.0 3.0 1.0 -2.0 0.0 1.0 3.0 1.0
Distance of time series is 18.000000
0.000u 0.000s 0:02.38 0.0%      0+0k 0+0io 0pf+0w
$
```

```
$ time ./distdtwdp
8 10
1.0 2.0 0.0 1.0 2.0 0.0 1.0 -1.0
-2.0 0.0 1.0 3.0 1.0 -2.0 0.0 1.0 3.0 1.0
Distance of time series is 22.000000
0.000u 0.000s 0:23.34 0.0%      0+0k 0+0io 0pf+0w
$
$ time ./distdtwdp
20 20
2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0
1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
Distance of time series is 20.000000
0.000u 0.000s 0:04.75 0.0%      0+0k 0+0io 0pf+0w
$
```

Συγχώνευση των μεθόδων σε ενιαίο πρόγραμμα (5%)

Υλοποιήσατε δύο διαφορετικές μεθόδους εύρεσης της απόστασης DTW δύο χρονοσειρών, μέσω των συναρτήσεων `dtwrec` και `dtwdp`. Μπορείτε να οργανώσετε έτσι τον κώδικά σας σε ένα ενιαίο πηγαίο αρχείο, έστω με όνομα "`distdtw.c`", ώστε τελικά να καλείται για την επίλυση του προβλήματος η κατάλληλη συνάρτηση μεταξύ των δύο, ανάλογα με το αν έχει ορισθεί μία συμβολική σταθερά με όνομα `REC`. Δηλαδή, αν έχει ορισθεί η σταθερά αυτή, να καλείται η `dtwrec`, αλλιώς η `dtwdp`. Για το πώς μπορεί να γίνει αυτό, δείτε τα σχετικά με `#ifdef` και `#endif` στη σελίδα 159 των σημειώσεων/διαφανειών του μαθήματος. Άπαξ και γίνει η συγχώνευση, μπορείτε να δημιουργήσετε τα εκτελέσιμα που αντιστοιχούν σε κάθε μέθοδο ως εξής:

```
$ gcc -o distdtwrec distdtw.c -DREC
$ gcc -o distdtwdp distdtw.c
$
```

Απόσταση DTW υπό περιορισμό (20%)

Συνήθως, όταν μας ενδιαφέρει να βρούμε την απόσταση DTW δύο χρονοσειρών, αυτές έχουν το ίδιο μήκος. Στην περίπτωση αυτή, είναι πολύ πιθανό να θεωρούμε ότι δεν είναι λογικό να συνεισφέρουν στην απόσταση των δύο χρονοσειρών και οι αποστάσεις στοιχείων τους που είναι σε αρκετά απομακρυσμένες θέσεις. Τότε, βάζουμε ένα όριο (περιορισμό) c στην απόσταση των θέσεων δύο στοιχείων a_i και b_j, δηλαδή στο $|i-j|$, για να μπορεί να συνεισφέρει η απόστασή τους στην απόσταση DTW των χρονοσειρών. Για την παραλλαγή αυτή, ισχύει ακριβώς ο ίδιος ορισμός για το $dtw(A_i, B_j)$ που δόθηκε προηγουμένως, μόνο που τώρα έχουμε:

$$d(a_i, b_j) = \begin{cases} (a_i - b_j)^2 & \text{αν } |i-j| \leq c \\ +\infty & \text{αν } |i-j| > c \end{cases}$$

Τροποποιήστε το πρόγραμμα που έχετε ήδη γράψει, συντάσσοντας ένα καινούργιο πηγαίο, έστω με όνομα "`distcdtw.c`", το οποίο να έχει τη δυνατότητα να δεχθεί από τη γραμμή εντολής ένα μη αρνητικό ακέραιο ως όρισμα. Το όρισμα αυτό, όταν δίνεται, να αντιπροσωπεύει το όριο/περιορισμό c που θα χρησιμοποιηθεί στον υπολογισμό της απόστασης DTW υπό περιορισμό δύο χρονοσειρών. Αν δεν δοθεί όρισμα, να θεωρείται ότι το c ισούται με $+\infty$, το οποίο πρακτικά σημαίνει ότι η απόσταση των στοιχείων a_i και b_j θα είναι πάντοτε $(a_i - b_j)^2$. Το πρόγραμμα να διαβάζει από την είσοδο

το κοινό μήκος δύο χρονοσειρών και στη συνέχεια, σε δύο γραμμές, τα στοιχεία των χρονοσειρών αυτών. Επίσης, στο πρόγραμμα να επιλέγεται ποια από τις δύο μεθόδους (αναδρομική ή δυναμικού προγραμματισμού) θα είναι ενεργοποιημένη μέσω του ορισμού (ή όχι) της συμβολικής σταθεράς `REC`, έτσι ώστε να μπορεί να προκύψει κατά τη μεταγλώττιση και σύνδεση το κατάλληλο εκτελέσιμο αρχείο. Παραδείγματα εκτέλεσης για την παραλλαγή του υπολογισμού απόστασης DTW υπό περιορισμό είναι τα εξής:

```
$ hostname
linux29
$
$ gcc -o distcdtwrec distcdtw.c -DREC
$ gcc -o distcdtwdp distcdtw.c
$
$ ./distcdtwrec
5
1.0 2.0 0.0 -1.0 -2.0
-1.0 0.0 2.0 3.0 1.0
Distance of time series is 20.000000
$
$ ./distcdtwrec 1
5
1.0 2.0 0.0 -1.0 -2.0
-1.0 0.0 2.0 3.0 1.0
Distance of time series is 27.000000
$
$ ./distcdtwrec 0
5
1.0 2.0 0.0 -1.0 -2.0
-1.0 0.0 2.0 3.0 1.0
Distance of time series is 37.000000
$
$ ./distcdtwdp
5
1.0 2.0 0.0 -1.0 -2.0
-1.0 0.0 2.0 3.0 1.0
Distance of time series is 20.000000
$
$ ./distcdtwdp 1
5
1.0 2.0 0.0 -1.0 -2.0
-1.0 0.0 2.0 3.0 1.0
Distance of time series is 27.000000
$
$ ./distcdtwdp 0
5
1.0 2.0 0.0 -1.0 -2.0
-1.0 0.0 2.0 3.0 1.0
Distance of time series is 37.000000
$ time ./distcdtwrec 5
```

```
9
2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0
1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
Distance of time series is 9.000000
32.238u 0.000s 0:35.49 90.8%    0+0k 0+0io 0pf+0w
$
$ time ./distcdtwdp 5
9
2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0
1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
Distance of time series is 9.000000
0.000u 0.000s 0:14.14 0.0%      0+0k 16+0io 0pf+0w
$
```

Κατηγοριοποίηση χρονοσειρών (50% + 50% + 20%)

Το τελικό πρόγραμμα που καλείσθε να αναπτύξετε και να παραδώσετε σχετίζεται με την κατηγοριοποίηση χρονοσειρών. Δηλαδή, έχοντας διαθέσιμο ένα σύνολο από χρονοσειρές, όπου κάθε μία ανήκει σε μία εκ των προτέρων γνωστή κατηγορία/κλάση, επιθυμούμε να μπορούμε για μία χρονοσειρά που δεν γνωρίζουμε σε ποια κλάση ανήκει (ή, και αν γνωρίζουμε, το αγνοούμε), να την εντάξουμε σε κάποια κλάση. Η ένταξή της γίνεται στην κλάση στην οποία ανήκει εκείνη από τις χρονοσειρές για τις οποίες γνωρίζουμε την κλάση τους που είναι πλησιέστερη, με μέτρο την απόσταση DTW, στη χρονοσειρά με άγνωστη κλάση. Στη συνέχεια περιγράφονται τα χαρακτηριστικά του προγράμματος που καλείσθε να υλοποιήσετε.

- Το πρόγραμμα πρέπει να μπορεί να δέχεται στη γραμμή εντολής ένα μη αρνητικό ακέραιο που θα είναι το όριο/περιορισμός για τον υπολογισμό των αποστάσεων DTW, όπου αυτές χρειάζονται στη συνέχεια. Αν δεν δοθεί όρισμα, θεωρείται ότι δεν υπάρχει περιορισμός (ή, ισοδύναμα, αυτό είναι το $+\infty$).

- Το πρόγραμμα πρέπει μέσω του ορισμού ή όχι της συμβολικής σταθεράς `REC` να μπορεί να λειτουργεί είτε με την αναδρομική μέθοδο, είτε με αυτήν του δυναμικού προγραμματισμού.

- Η είσοδος του προγράμματος θα είναι η εξής: Αρχικά δίνονται σε μία γραμμή το πλήθος των χρονοσειρών με γνωστή κλάση, δηλαδή του *συνόλου εκπαίδευσης* (training set), και το μήκος των χρονοσειρών αυτών, που είναι το ίδιο για όλες τις χρονοσειρές. Στη συνέχεια, δίνονται κατά γραμμές τα δεδομένα για κάθε χρονοσειρά, που περιλαμβάνουν αρχικά την κλάση της χρονοσειράς (ένας ακέραιος) και τα στοιχεία της. Ακολουθούν τα δεδομένα για τις χρονοσειρές που αποτελούν το *σύνολο ελέγχου* (test set), που δίνονται στην ίδια μορφή με αυτή του συνόλου εκπαίδευσης. Δηλαδή, αρχικά δίνονται σε μία γραμμή το πλήθος των χρονοσειρών του συνόλου ελέγχου και το μήκος των χρονοσειρών αυτών (ίδιο για όλες, επίσης). Στη συνέχεια, δίνονται κατά γραμμές και τα δεδομένα για τις χρονοσειρές ελέγχου. Και γι΄ αυτές σαν πρώτο στοιχείο θα δίνεται η κλάση τους, παρότι υποτίθεται ότι δεν την γνωρίζουμε. Ο λόγος που δίνεται είναι για να μπορούμε να ελέγξουμε εκ των υστέρων τον βαθμό επιτυχίας της μεθόδου κατηγοριοποίησης χρονοσειρών που βασίζεται στην απόσταση DTW. Θα πρέπει να σημειωθεί ότι το μήκος των χρονοσειρών ελέγχου πρέπει να είναι το ίδιο με το μήκος των χρονοσειρών εκπαίδευσης, διαφορετικά το πρόγραμμα θα πρέπει να τερματίζει με κάποιο μήνυμα λάθους.

- Για κάθε χρονοσειρά από το σύνολο ελέγχου, το πρόγραμμα θα πρέπει να βρίσκει την πλησιέστερη με αυτήν από το σύνολο εκπαίδευσης, με βάση την απόσταση DTW μεταξύ τους (με ή

χωρίς περιορισμό, ανάλογα με το αν έχει δοθεί όρισμα στη γραμμή εντολής ή όχι), οπότε να κατατάσσει τη χρονοσειρά ελέγχου στην κλάση στην οποία ανήκει η πλησιέστερή της στο σύνολο εκπαίδευσης. Αν η κατάταξη αυτή είναι λάθος, με βάση την κλάση στην οποία γνωρίζουμε ότι ανήκει η χρονοσειρά ελέγχου (αλλά δεν χρησιμοποιήσαμε κατά την κατηγοριοποίηση), θεωρούμε ότι έχουμε μία αποτυχία. Αφού το πρόγραμμα κατηγοριοποιήσει με αυτόν τον τρόπο όλες τις χρονοσειρές ελέγχου, να εκτυπώνει και το σφάλμα της κατηγοριοποίησης ως το πηλίκο του πλήθος των αποτυχιών προς το πλήθος των χρονοσειρών ελέγχου.

- Είναι προφανές ότι όταν οι χρονοσειρές έχουν μεγάλο πλήθος στοιχείων, η αναδρομική μέθοδος δεν μπορεί να δώσει αποτέλεσμα, όπως έχετε ήδη παρατηρήσει. Δεν ζητείται να κάνετε κάτι γι΄ αυτό (δεν είναι δυνατόν, άλλωστε). Όμως, αν τα σύνολα εκπαίδευσης και ελέγχου περιέχουν μεγάλο αριθμό χρονοσειρών με πολύ μεγάλο πλήθος στοιχείων, ενδέχεται και η μέθοδος του δυναμικού προγραμματισμού να έχει πρόβλημα στην απόδοση. Σκεφτείτε ότι για την εύρεση της απόστασης δύο χρονοσειρών μήκους N, ο προφανής τρόπος υλοποίησης θα έχει πολυπλοκότητα χρόνου $O(N^2)$. Είναι δυνατόν όμως, αν έχει δοθεί όριο/περιορισμός c, ο αλγόριθμος να έχει πολυπλοκότητα χρόνου $O(cN)$. Είναι ζητούμενο της εργασίας να υλοποιήσετε τη "γρήγορη" εκδοχή της μεθόδου του δυναμικού προγραμματισμού (η υιοθέτηση της "αργής" εκδοχής θα επιφέρει μείωση στη βαθμολογία κατά 10%).

- Είναι εύκολο να δει κανείς ότι ο προφανής τρόπος υλοποίησης θα έχει πολυπλοκότητα μνήμης $O(N^2)$. Σε περιπτώσεις χρονοσειρών με εξαιρετικά μεγάλο πλήθος στοιχείων (π.χ. > 100000), θα υπάρχει πρόβλημα μνήμης. Είναι δυνατόν όμως να υλοποιηθεί η μέθοδος του δυναμικού προγραμματισμού για το πρόβλημα αυτό με πολυπλοκότητα μνήμης $O(N)$. Δεν απαιτείται στα ζητούμενα της εργασίας να υλοποιηθεί η "οικονομική" εκδοχή της μεθόδου του δυναμικού προγραμματισμού, αν όμως αυτό γίνει, μπορεί να οδηγήσει σε bonus στη βαθμολογία της άσκησης μέχρι και 20%.

- Θα πρέπει να δομήσετε το πρόγραμμά σας σε ένα σύνολο από **τουλάχιστον δύο πηγαία αρχεία** C (με κατάληξη `.c`) και **τουλάχιστον ένα αρχείο επικεφαλίδας** (με κατάληξη `.h`).

- Δεδομένου ότι το πρόγραμμα που καλείσθε να αναπτύξετε και να παραδώσετε υπερκαλύπτει τη λειτουργικότητα των προγραμμάτων που περιγράφηκαν σε προηγούμενες ενότητες και ζητούσαν απλώς την εύρεση της DTW απόστασης (με ή χωρίς περιορισμό) μεταξύ δύο χρονοσειρών, δεν χρειάζεται να παραδώσετε τα προηγούμενα προγράμματα. Αυτό θα το κάνετε μόνο αν δεν ολοκληρώσετε το πρόγραμμα για την κατηγοριοποίηση χρονοσειρών, ώστε να πάρετε το μέρος της βαθμολογίας της εργασίας που θα αντιστοιχεί σε ό,τι παραδώσατε. Η έννοια των ποσοστών στον τίτλο της ενότητας είναι ότι το πρώτο 50% αναφέρεται στα ζητούμενα των προηγούμενων ενοτήτων, το δεύτερο 50% στην υλοποίηση της κατηγοριοποίησης χρονοσειρών που ζητήθηκε σ΄ αυτή την ενότητα και το 20% αναφέρεται στην ενδεχόμενη υλοποίηση της οικονομικής εκδοχής σε μνήμη.

Παραδείγματα εκτέλεσης του προγράμματος, μόνο της εκδοχής του δυναμικού προγραμματισμού, αφού για την αναδρομική δεν υπάρχει ελπίδα τερματισμού σε πεπερασμένο χρόνο για χρονοσειρές μη τετριμμένου μεγέθους, δίνονται στη συνέχεια. Τα αρχεία δεδομένων μπορείτε να τα κατεβάσετε από τις διευθύνσεις

`http://www.di.uoa.gr/~ip/hwfiles/dtw/train_FaceFour.txt`
`http://www.di.uoa.gr/~ip/hwfiles/dtw/test_FaceFour.txt`
`http://www.di.uoa.gr/~ip/hwfiles/dtw/train_ECG5000.txt`
`http://www.di.uoa.gr/~ip/hwfiles/dtw/test_ECG5000.txt`
`http://www.di.uoa.gr/~ip/hwfiles/dtw/bigseries.txt`

Από την ιστοσελίδα

http://www.cs.ucr.edu/~eamonn/time_series_data/

μπορείτε να κατεβάσετε πολλά αρχεία δεδομένων με χρονοσειρές. Για να μετατρέψετε τα αρχεία αυτά στη μορφή που απαιτείται για την εργασία, μπορείτε να χρησιμοποιήσετε το πρόγραμμα κελύφους από τη διεύθυνση

http://www.di.uoa.gr/~ip/hwfiles/dtw/convform

```
$ hostname
linux29
$
$ cat train_FaceFour.txt test_FaceFour.txt | ./dtwdp
Series 1 (class 3) is nearest (distance 84.826912) to series 15 (class 3)
Series 2 (class 1) is nearest (distance 14.578062) to series 22 (class 1)
Series 3 (class 2) is nearest (distance 32.433863) to series 1 (class 2)
Series 4 (class 4) is nearest (distance 36.686934) to series 23 (class 2)
Series 5 (class 1) is nearest (distance 10.463726) to series 22 (class 1)
.....................................................................
Series 86 (class 4) is nearest (distance 11.613004) to series 6 (class 4)
Series 87 (class 2) is nearest (distance 26.171747) to series 14 (class 2)
Series 88 (class 4) is nearest (distance 16.116654) to series 11 (class 4)
Error rate: 0.170
CPU time: 5.49 secs
$
$ cat train_FaceFour.txt test_FaceFour.txt | ./dtwdp 0
Series 1 (class 3) is nearest (distance 428.324353) to series 17 (class 3)
Series 2 (class 1) is nearest (distance 90.726199) to series 22 (class 1)
Series 3 (class 2) is nearest (distance 271.997783) to series 3 (class 1)
Series 4 (class 4) is nearest (distance 202.823627) to series 11 (class 4)
Series 5 (class 1) is nearest (distance 79.581355) to series 24 (class 1)
.....................................................................
Series 86 (class 4) is nearest (distance 38.488262) to series 6 (class 4)
Series 87 (class 2) is nearest (distance 145.152577) to series 16 (class 2)
Series 88 (class 4) is nearest (distance 94.753967) to series 7 (class 4)
Error rate: 0.216
CPU time: 0.03 secs
$
$ cat train_FaceFour.txt test_FaceFour.txt | ./dtwdp 7
Series 1 (class 3) is nearest (distance 143.983927) to series 15 (class 3)
Series 2 (class 1) is nearest (distance 14.906337) to series 22 (class 1)
Series 3 (class 2) is nearest (distance 37.187481) to series 1 (class 2)
Series 4 (class 4) is nearest (distance 53.816056) to series 6 (class 4)
Series 5 (class 1) is nearest (distance 10.693162) to series 22 (class 1)
.....................................................................
Series 86 (class 4) is nearest (distance 11.613004) to series 6 (class 4)
Series 87 (class 2) is nearest (distance 48.565023) to series 14 (class 2)
Series 88 (class 4) is nearest (distance 16.670844) to series 7 (class 4)
Error rate: 0.114
```

```
CPU time: 0.26 secs
$
$ cat train_ECG5000.txt test_ECG5000.txt | ./dtwdp > /dev/null
Error rate: 0.076
CPU time: 934.48 secs
$
$ cat train_ECG5000.txt test_ECG5000.txt | ./dtwdp 0 > /dev/null
Error rate: 0.075
CPU time: 11.50 secs
$
$ cat train_ECG5000.txt test_ECG5000.txt | ./dtwdp 45 > /dev/null
Error rate: 0.076
CPU time: 521.41 secs
$
$ ./dtwdp 5000 < bigseries.txt
Series 1 (class 1) is nearest (distance 1155807.662826) to series 1 (class 1)
Error rate: 0.000
CPU time: 30.75 seconds
$
```

Παραδοτέο

Για να παραδώσετε το σύνολο των αρχείων που θα έχετε δημιουργήσει για την άσκηση αυτή, ακολουθήστε την εξής διαδικασία. Τοποθετήστε όλα τα αρχεία μέσα σ΄ ένα κατάλογο που θα δημιουργήσετε, έστω με όνομα `dtw`, στους σταθμούς εργασίας του Τμήματος. Χρησιμοποιώντας την εντολή `zip` ως εξής

```
zip -r dtw.zip dtw
```

δημιουργείτε ένα συμπιεσμένο (σε μορφή `zip`) αρχείο, με όνομα `dtw.zip`, στο οποίο περιέχεται ο κατάλογος `dtw` μαζί με όλα τα περιεχόμενά του.[1] Το αρχείο αυτό είναι που θα πρέπει να υποβάλετε μέσω του eclass.[2]

[1] Αρχεία `zip` μπορείτε να δημιουργήσετε και στα Windows, με διάφορα προγράμματα, όπως το WinZip.

[2] Μην υποβάλετε ασυμπίεστα αρχεία ή αρχεία που είναι συμπιεσμένα σε άλλη μορφή εκτός από `zip` (π.χ. `rar`, `7z`, `tar`, `gz`, κλπ.), γιατί δεν θα γίνουν δεκτά για αξιολόγηση.