Πρόγνωση – Ανίχνευση Ανωμαλιών – Συμπίεση Χρονοσειρών με Νευρωνικά Δίκτυα

Μέλη Ομάδας

Κωνσταντίνος Μαϊδάτσης (AM: 1115201800102)
Διονύσιος Μανιατάκος (AM: 1115201800104)

Σύνδεσμος GitHub

https://github.com/Kosmai/LSH_kNN_Clustering

Για λόγους αποφυγής αντιγραφής το παραπάνω repository είναι ιδιωτικό προς το παρόν.

Ερώτημα Α

Γενική Περιγραφή
Στο συγκεκριμένο ερώτημα υλοποείται ένα recurrent νευρωνικό δίκτυο αρχιτεκτονικής LSTM (Long Short Term Memory) το οποίο χρησιμοποιείται για πρόβλεψη τιμών σε χρονοσειρές. Η πρόβλεψη αφορά μία τιμή δεδομένων των προηγούμενων k τιμών (τις τιμές υστέρησης – lookback). Γίνεται οπτικοποίηση των αποτελεσμάτων συγκρίνοντας τις πραγματικές τιμές με τις προβλεπόμενες, την εξέλιξη δηλαδή της χρονοσειράς. Υπάρχει δυνατότητα εκπαίδευσης του μοντέλου με το δοσμένο σύνολο αλλά και χρήσης προεκπαιδευμένου μοντέλου για την παραγωγή προβλέψεων στο δοσμένο σύνολο. Γίνεται χρήση της γλώσσας προγραμματισμού Python 3 με το framework Tensorflow μέσω του Keras.

Οδηγίες Εκτέλεσης/Χρήσης
Για την εκτέλεση του αρχείου “forecast.py” είναι απαραίτητο να είναι εγκατεστημένος ο διερμηνέας της Python 3 και τα modules “numpy”, “pandas”, “sklearn”, “matplotlib”, “tensorflow” και “keras”. Αυτά μπορούν να εγκατασταθούν μέσω της εντολής “pip install <module>”. Δε χρησιμοποιήθηκε CUDA καθώς δεν υπήρχε συμβατή κάρτα γραφικών στα τοπικά μας μηχανήματα. Η εντολή εκτέλεσης του προγράμματος έχει την εξής μορφή:

python3 forecast.py -d <dataset> -n <number of time series selected> (-individual_training) (-model <model_name>) (-save_model)

Ενδεικτικά: python3 forecast.py -d datasets/nasdaq2007_17.csv -n 15 -model models/forecast/batch_trained_model

Τα ορίσματα που βρίσκονται σε παρένθεση είναι προαιρετικά. Ακολουθεί η περιγραφή της λειτουργικότητας των ορισμάτων:

-d : Για τον προσδιορισμό του συνόλου δεδομένων
-n : Για την επιλογή του αριθμού των χρονοσειρών που θα χρησιμοποιηθούν (για εκπαίδευση/πρόβλεψη)
-individual_training : Για να επιλεχθεί αν η εκπαίδευση θα γίνει ανα χρονοσειρά ή ανα σύνολο
-save_model : Για να γίνει αποθήκευση του μοντέλου που έχει εκπαιδευτεί.
-model : Για να φορτωθεί κάποιο προεκπαιδευμένο αποθηκευμένο μοντέλο και να αποφευχθεί η εκ νέου εκπαίδευση. Στην περίπτωση αυτή οι προβλέψεις δεν γίνονται στο κομμάτι ελέγχου των χρονοσειρών (προεπιλογή το τελευταίο 30%) αλλά σε όλη. Αυτό χρησιμοποιείται όταν έχουμε ήδη κάποιο καλά εκπαιδευμένο μοντέλο και θέλουμε να το θέσουμε σε λειτουργία.

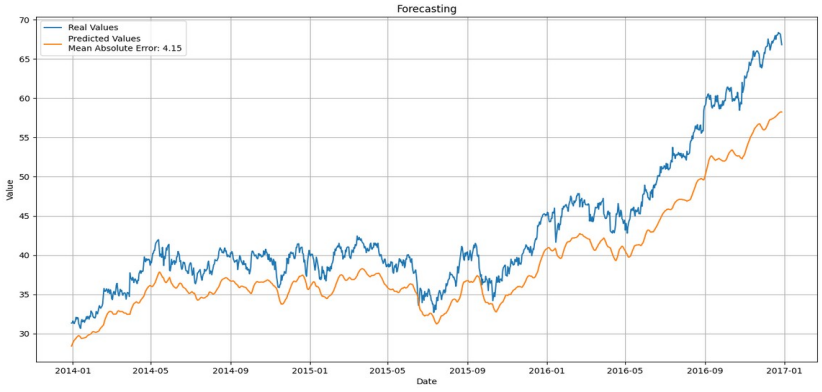
Το σύνολο δεδομένων που εισάγεται πρέπει να ακολουθεί τη μορφή των αρχείων που έχουν δοθεί.

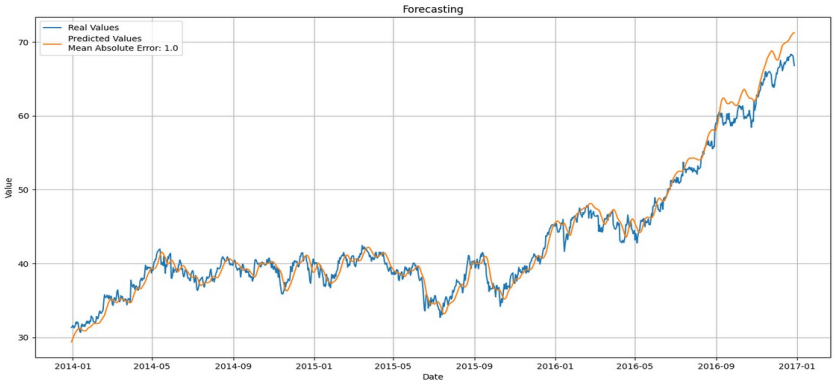
Στα παραδοτέα συμπεριλαμβάνεται ένα προεκπαιδευμένο μοντέλο που έχει εκπαιδευτεί με σύνολο χρονοσειρών(“models/forecast/batch_trained_model”) και, ενδεικτικά, ένα που έχει εκπαιδευτεί με την πρώτη χρονοσειρά (“models/forecast/ts1_trained_model”).

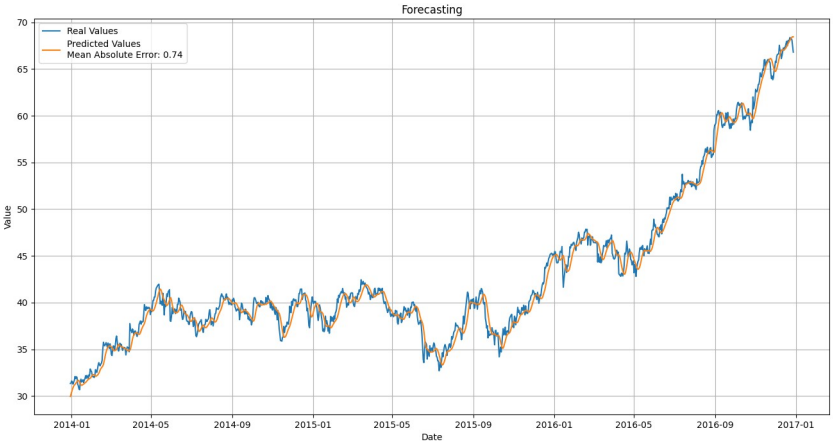
Πειραματισμός Εκπαίδευσης Μοντέλου
Γενικά, όπως και σε κάθε πρόβλημα μηχανικής μάθησης, η επίδοση του μοντέλου εξαρτάται από την επιλογή των υπερπαραμέτρων. Ακολουθούν πειραματισμοί που έγιναν για τον προσδιορισμό ενός κατάλληλου μοντέλου. Σε κάθε περίπτωση χρησιμοποιείται ο Adam optimizer, και ως συνάρτηση απώλειας η mean squared error, που είναι μια καλή επιλογή για προβλήματα παλινδρόμησης.

Αρχικά, θα πειραματιστούμε ως προς τη χρήση διαφορετικού αριθμού και μεγέθους LSTM στρωμάτων.

1 LSTM Στρώμα – 10 Κόμβοι/50 Κόμβοι/ 300 Κόμβοι

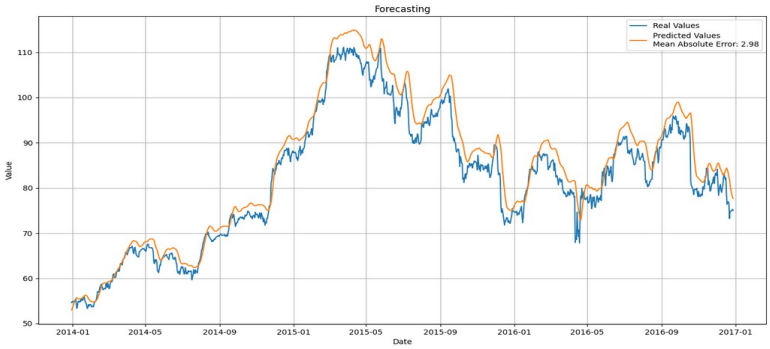


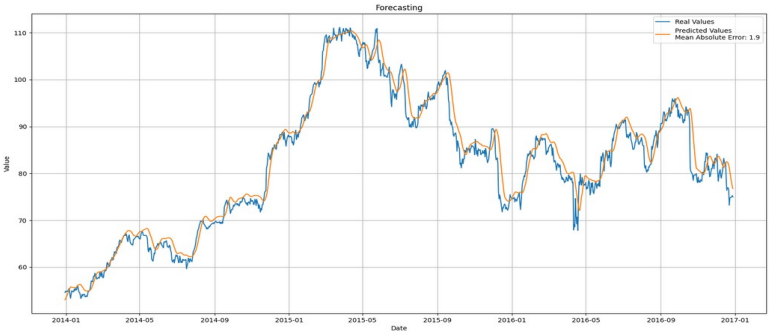


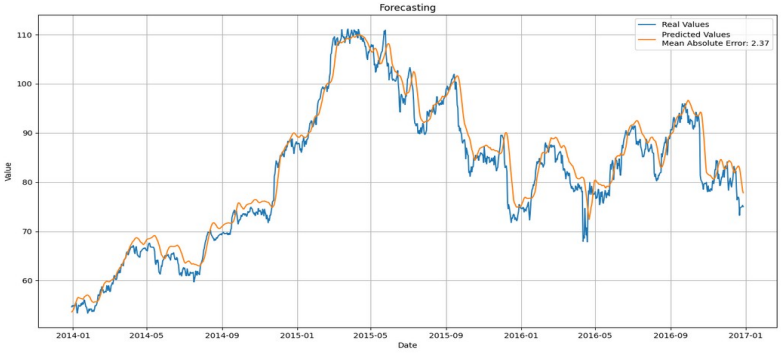


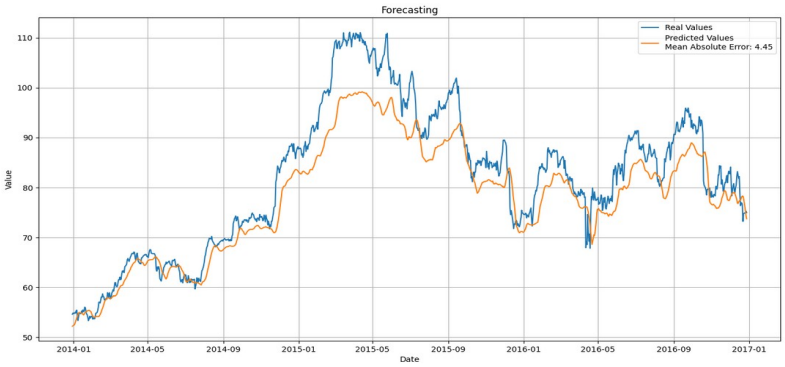
Παρατηρούμε ότι, χρησιμοποιώντας 10 μόνο units, ο χρόνος εκπαίδευσης είναι αρκετά χαμηλός αλλά οι προβλέψεις όχι αρκετά ακριβείς, πράγμα που μπορούμε να συμπεράνουμε από το σχετικά υψηλότερο mean absolute error. Χρησιμοποιώντας 50 units, τα αποτελέσματα στις προβλέψεις είναι αρκετά καλύτερα, με μικρή αύξηση στον χρόνο εκπαίδευσης. Στην τελευταία περίπτωση με τη χρήση 300 units, τα αποτελέσματα είναι λίγο καλύτερα αλλά ο χρόνος εκπαίδευσης υψηλός. Αν χρησιμοποιήσουμε πολύ μεγαλύτερο αριθμό units (π.χ 1000), απαιτείται υπερβολικά υψηλότερος χρόνος εκπαίδευσης χωρίς τα αποτελέσματα να βελτιώνονται αισθητά. Επομένως, μία καλή επιλογή βρίσκεται κάπου ενδιάμεσα. Θα δοκιμάσουμε τώρα να πειραματιστούμε με τον αριθμό των στρωμάτων.

1/2/3/4 LSTM Στρώματα με 50 Κόμβους









Παρατηρούμε ότι, για 2 LSTM στρώματα έχουμε καλύτερες προβλέψεις σε σχέση με τη χρήση 1 LSTM στρώματος. Ωστόστο, καθώς αυξάνουμε πολύ τον αριθμό των στρωμάτων, τα αποτελέσματα χειροτερεύουν. Επομένως, θα επιλέξουμε 2 με 3 στρώματα για το μοντέλο μας, αφού για τέτοιο αριθμό στρωμάτων παρατηρούμε καλή συμπεριφορά.

Πειραματισμός με Epochs

Παρατηρούμε ότι για πολύ μικρό αριθμό εποχών κατα την εκπαίδευση (π.χ για 1 εποχή) το μοντέλο μας υφίσταται υποπροσαρμογή (underfitting) αφού δεν προλαβαίνει να μάθει/εκφράσει καλά τα δεδομένα. Αυξάνοντας τον αριθμό των εποχών (π.χ 5-10 εποχές), τα αποτελέσματα στις προβλέψεις είναι αρκετά καλύτερα. Αν δοκιμάσουμε να αυξήσουμε υπερβολικά τον αριθμό των εποχών (π.χ για 50 εποχές) απαιτείται αρκετά μεγαλύτερος χρόνος εκπαίδευσης και υπάρχει ο κίνδυνος της υπερπροσαρμογής (overfitting), να μάθει δηλαδή το μοντέλο υπερβολικά καλά τα δεδομένα εκπαίδευσης αλλά να μη μπορεί να γενικευτεί. Επομένως, θα χρησιμοποιηθούν 5-10 εποχές.

Πειραματισμός με Τιμές Υστέρησης (Lookback)

Χρησιμοποιώντας λίγες τιμές υστέρησης (π.χ 5), οι προβλέψεις δεν είναι αρκετά ακριβείς, καθώς το νευρωνικό δίκτυο δε δέχεται ως είσοδο ακολουθία τιμών αρκετά μεγάλη ώστε να μπορεί να την ερμηνεύσει. Χρησιμοποιώντας μεγαλύτερη τιμή υστέρησης (π.χ 50) τα αποτελέσματα είναι αρκετά καλύτερα. Αν χρησιμοποιήσουμε υπερβολικά μεγάλη τιμή, η απόδοση του μοντέλου δε βελτιώνεται και σε κάποιες περιπτώσεις φθήνει.

Εκπαίδευση ανα Χρονοσειρά και Εκπαίδευση ανα Σύνολο

Στα παραπάνω πειράματα η εκπαίδευση γινόταν ανα χρονοσειρά στο 70% της χρονοσειράς και ο έλεγχος στο 30% της. Κάτι τέτοιο είναι συχνά χρήσιμο αφού θέλουμε το μοντέλο μας να χρησιμοποιηθεί αποκλειστικά για τη συγκεκριμένη χρονοσειρά και να εξάγει πιθανές προβλέψεις πάνω σε αυτή. Πολλές φορές όμως, θέλουμε ένα μοντέλο που μπορεί να γενικευτεί και σε άλλες χρονοσειρές μοντελοποιώντας μια πιο γενική συμπεριφορά (των μετοχών). Στην περίπτωση αυτή θα εκπαιδεύσουμε το μοντέλο με ένα σύνολο διαφορετικών χρονοσειρών (π.χ 10). Με τον τρόπο αυτό το μοντέλο μαθαίνει να μοντελοποιεί τη συμπεριφορά που έχουν γενικά οι μετοχές, χωρίς όμως να μπορεί να προσαρμοστεί, απαραίτητα, σε ιδιαιτερότητες. Ως εκ τούτου, αν θέλουμε να να εξάγουμε πιο ακριβείς προβλέψεις για μία χρονοσειρά, είναι προτιμότερο να εκπαιδεύσουμε το μοντέλο με βάση τα ιστορικά δεδομένα της συγκεκριμένης χρονοσειράς.

Παραδοχές

Είναι σημαντικό να σημειωθεί ότι το συγκεκριμένο πρόγραμμα όπως και το άρθρο στο οποίο αυτό βασίστηκε (και δόθηκε ως οδηγός) έχει τη δυνατότητα να προβλέπει μία μόνο τιμή χρησιμοποιώντας τις κ προηγούμενες τιμές μιας χρονοσειράς. Επομένως, για να δούμε προβλέψεις για πολλές τιμές στο μέλλον πρέπει είτε να χρησιμοποιήσουμε και τις προηγούμενες μελλοντικές τιμές (που σε ένα πραγματικό σενάριο δε γνωρίζουμε) ή να χρησιμοποιήσουμε τις προβλέψεις που έχουν εξαχθεί για αυτές. Φυσικά στη δεύτερη περίπτωση, οι προβλέψεις για το “μακρινό” μέλλον δε θα είναι τόσο ακριβείς και σε καμία περίπτωση δε θα προσεγγίζουν την ακρίβεια της εξέλιξης της τιμής που φαίνεται στα παραπάνω διαγράμματα και στο διάγραμμα του εν λόγω άρθρου.

Ως σύνολο εκπαίδευσης χρησιμοποιείται το πρώτο “κομμάτι” κάθε χρονοσειράς (το 70%) και ως σύνολο ελέγχου το δεύτερο κομμάτι (το 30%). Σε περίπτωση που χρησιμοποιηθεί προεκπαιδευμένο μοντέλο, οι προβλέψεις γίνονται για όλη τη χρονοσειρά. Φυσικά, δε μπορεί να προβλεφθεί το πρώτο τμήμα της χρονοσειράς που έχει μέγεθος όσες και οι τιμές υστέρησης, αφού δεν έχουμε τις προηγούμενες τιμές για να εξάγουμε πρόβλεψη.

Ερώτημα Β

Γενική Περιγραφή

Στο συγκεκριμένο ερώτημα υλοποείται ένα recurrent νευρωνικό δίκτυο αρχιτεκτονικής LSTM (Long Short Term Memory) το οποίο χρησιμοποιείται για αυτοκωδικοποίηση χρονοσειρών με απώτερο σκοπό την ανίχνευση ανωμαλιών σε αυτές, συγκρίνοντας τα αποκωδικοποιήμενα αποτελέσματα με τα αρχικά. Με βάση τη θεωρία, αναμένουμε οι αποκωδικοποιημένες χρονοσειρές να είναι πιο ομαλές, να έχει δηλαδή περιοριστεί ο θόρυβος. Και σε αυτή τη περίπτωση υπάρχει τόσο η δυνατότητα εκπαίδευσης ενός μοντέλου όσο και η δυνατότητα χρήσης ενός προεκπαιδευμένου μοντέλου. Γίνεται οπτικοποίηση των αποτελεσμάτων σε διαγράμματα όπου παρουσιάζονται τα σημεία της χρονοσειράς που αποτελούν ανωμαλίες. Χρησιμοποιούνται οι ίδιες τεχνολογίες με το προηγούμενο ερώτημα.

Οδηγίες Εκτέλεσης/Χρήσης

Η εκτέλεση του αρχείου “detect.py” έχει τα ίδια προαπαιτούμενα με το προηγούμενο ερώτημα. Η εντολή εκτέλεσης του προγράμματος έχει την εξής μορφή:

python3 detect.py -d <dataset> -n <number of time series selected> -mae <double> (-model <model_name>) (-save_model)

Ενδεικτικά: python3 detect.py -d datasets/nasdaq2007_17.csv -n 15 -mae 0.15 -model models/detect/model1

Τα ορίσματα που βρίσκονται σε παρένθεση είναι προαιρετικά. Ακολουθεί η περιγραφή της λειτουργικότητας των ορισμάτων:

-d : Για τον προσδιορισμό του συνόλου δεδομένων
-n : Για την επιλογή του αριθμού των χρονοσειρών που θα χρησιμοποιηθούν (για εκπαίδευση/πρόβλεψη)
-mae: Για την επιλογή του threshold που θα χρησιμοποιηθεί για την ανίχνευση των ανωμαλιών.
-save_model : Για να γίνει αποθήκευση του μοντέλου που έχει εκπαιδευτεί.
-model : Για να φορτωθεί κάποιο προεκπαιδευμένο αποθηκευμένο μοντέλο και να αποφευχθεί η εκ νέου εκπαίδευση. Στην περίπτωση αυτή οι προβλέψεις δεν γίνονται στο κομμάτι ελέγχου των χρονοσειρών (προεπιλογή το τελευταίο 30%) αλλά σε όλη. Αυτό χρησιμοποιείται όταν έχουμε ήδη κάποιο καλά εκπαιδευμένο μοντέλο και θέλουμε να το θέσουμε σε λειτουργία.

Το σύνολο δεδομένων που εισάγεται πρέπει να ακολουθεί τη μορφή των αρχείων που έχουν δοθεί.

Στα παραδοτέα συμπεριλαμβάνεται κάποια προεκπαιδευμένα μοντέλα (“models/detect/*”).

Πειραματισμός Εκπαίδευσης Μοντέλου

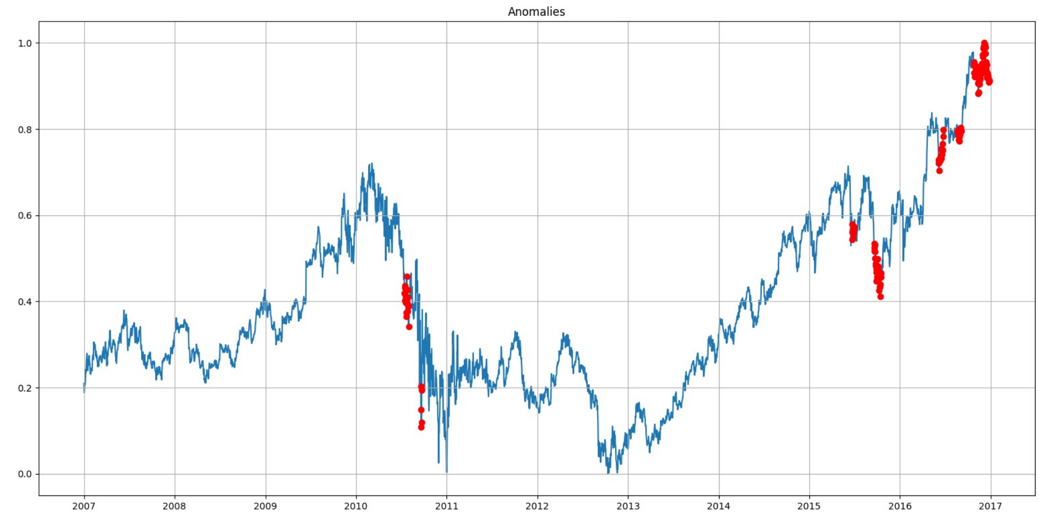
Στο συγκεκριμένο πρόβλημα, παρατηρήσαμε ότι η παραμετροποίηση παίζει αρκετά σημαντικό ρόλο στο πόσο θα απέχουν οι τιμές της πραγματικής χρονοσειράς από τις τιμές της αποκωδικοποιημένης. Χρησιμοποιώντας λίγες εποχές, οι αποκωδικοποιημένες χρονοσειρές που προκύπτουν είναι αρκετά διαφορετικές από τις πραγματικές, αφού το μοντέλο δεν προλαβαίνει να μάθει καλά, με αποτέλεσμα να προκύπτουν αρκετές ανωμαλίες αν επιλεχθεί σχετικά μεγάλο mae threshold. Από την άλλη, αν το μοντέλο μάθει να αναπαριστά αρκετά καλά τις χρονοσειρές, η διαφορά των πραγματικών από των αποκωδικοποιημένων θα είναι μικρότερη και θα χρειαστεί να χρησιμοποιηθεί ένα μικρότερο mae threshold για να εντοπιστούν οι ανωμαλίες.

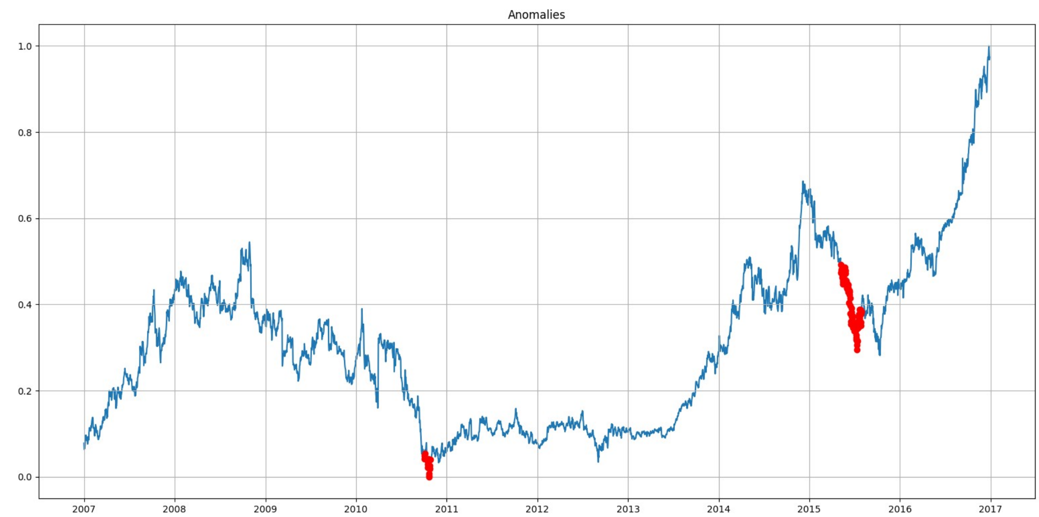
Για κάποιους συνδυασμούς υπερπαραμέτρων, το διάγραμμα διαφοράς μεταξύ πραγματικής και προβλεπόμενης χρονοσειράς είναι παρόμοιο με το διάγραμμα εξέλιξης της τιμής της χρονοσειράς. Υπάρχει δηλαδή η τάση οι υψηλές τιμές να ανιχνεύονται ως ανωμαλίες. Αυτό δεν ειναι επιθυμητό αφού οι υψηλές τιμές δεν είναι απαιραίτητα μη αναμενόμενες και υπάρχει περίπτωση χαμηλές τιμές να αποτελούν outliers. Ως εκ τούτου, επιλέξαμε τέτοιες υπερπαραμέτρους ώστε να μην προκύπτει το παραπάνω πρόβλημα.

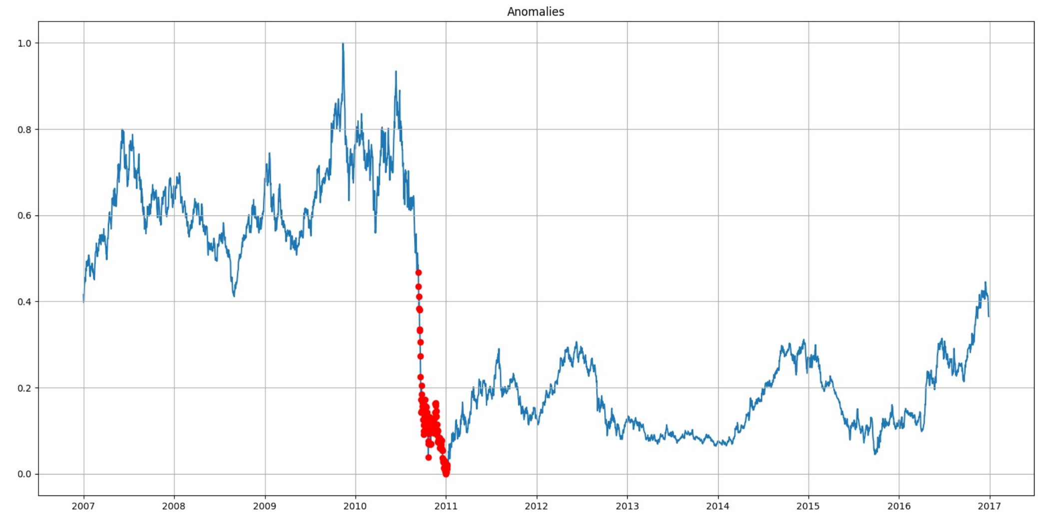
Όσον αφορά τα στρώματα, όπως και στο προηγούμενο ερώτημα έγινε η παρατήρηση οτι αυξάνοντας τον αριθμό τους κατα πολύ δεν έχουμε ιδιαίτερη βελτίωση και αυξάνεται ο χρόνος εκπαίδευσης. Επομένως, έγινε επιλογή 2 LSTM στρωμάτων.

Ιδιαίτερη σημασία στην απόδοση του μοντέλου έχει και η επιλογή του μήκους των ακολουθιών (sequences) που θα χρησιμοποιηθούν ως είσοδοι στο νευρωνικό δίκτυο. Για πολύ μικρό μήκος (π.χ 5) το δίκτυο δεν καταφέρνει να εκφράσει καλά τις χρονοσειρές αφού δεν του δίνεται αρκετά μεγάλη ακολουθία εισόδου. Αντίθετα, χρησιμοποιώντας μεγάλο μήκος ακολουθίας (π.χ 400) τα αποτελέσματα φαίνεται να είναι αρκετά καλά, αλλά ο χρόνος εκπαίδευσης ιδιαίτερα υψηλός. Επομένως, είναι απαραίτητο να βρεθεί μία μέση λύση ανάλογα και με τις απαιτήσεις του χρήστη (ακρίβεια ενάντια χρόνου εκπαίδευσης). Στο συγκεκριμένο πρόγραμμα χρησιμοποιείται η τιμή 200.

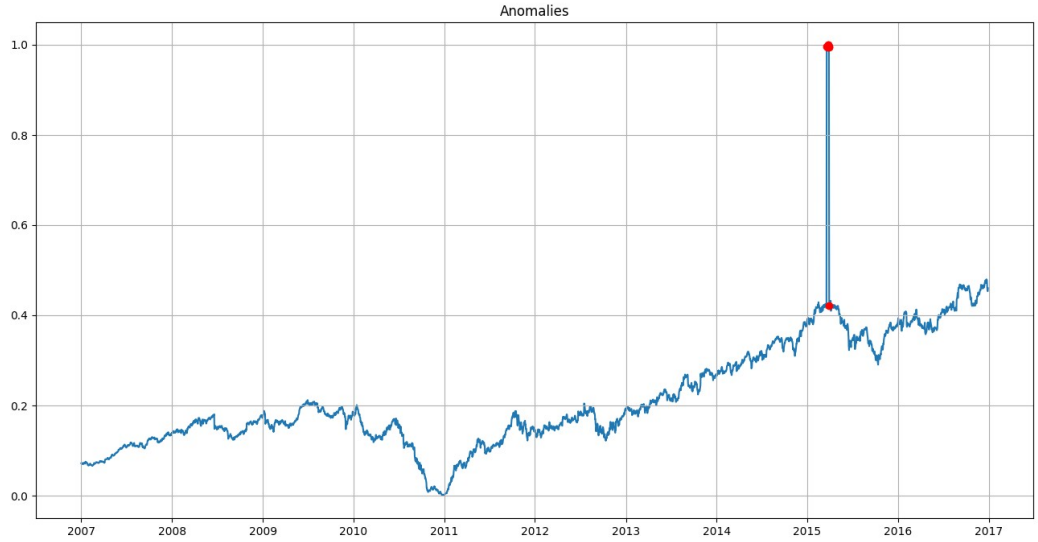
<u>Ενδεικτικά Αποτελέσματα</u>







Στο παρακάτω διάγραμμα, δοκιμάσαμε να αλλοιώσουμε τη συγκεκριμένη χρονοσειρά προστέθοντας μία τεχνητή ανωμαλία. Τα αποτελέσματα είναι τα αναμενόμενα, δηλαδή το συγκεκριμένο σημείο έχει σημειωθεί.



Παραδοχές

Κάθε χρονοσειρά έχει διαφορετική συμπεριφορά με αποτέλεσμα κάποιες να αυτοκωδικοποιούνται καλύτερα από το μοντέλο. Ως εκ τούτου, το μέσο σφάλμα μεταξύ πρωτότυπης χρονοσειράς και αυτοκωδικοποιημένης κινείται σε διαφορετική κλίμακα από χρονοσειρά σε χρονοσειρά. Επομένως, είναι ανάγκη ο χρήστης να δοκιμάσει διαφορετικές τιμές παραμέτρου mae, ώστε να δει λιγότερα ή περισσότερα σημεία που έχουν σημειωθεί ως ανωμαλίες. Σε κάθε περίπτωση ο χρήστης μπορεί να δει το διάγραμμα σφάλματος που εμφανίζεται (ασχέτως από την τιμή του mae threshold) ώστε να εντοπίσει τις τιμές που έχουν μεγαλύτερη απόκλιση.

Είναι σημαντικό να τονιστεί ότι το mae αναφέρεται στις κανονικοποιημένες χρονοσειρές, επομένως μπορεί να κυμένεται από 0 έως 1. Άλλωστε δε θα είχε νοημα να εξετάζαμε με τις μη-κανονικοποιημένες τιμές, αφού για μία χρονοσειρά μπορεί μία απόκλιση της τάξης του 20 να θεωρείται ανωμαλία ενώ για μιά άλλη να θεωρείται φυσιολογική μεταβολή (π.χ για μία μετοχή που παίρνει υψηλές τιμές).

Ερώτημα Γ

Γενική Περιγραφή
Στο συγκεκριμένο ερώτημα υλοποείται ένα μίας διάστασης συνελικτικό νευρωνικό δίκτυο για την αυτοκωδικοποίηση χρονοσειρών. Τελικός στόχος είναι η μείωση διαστάσεων στις χρονοσειρές και η παραγωγή ενός αρχείου δεδομένων που ακολουθεί την ίδια γραμμογράφηση με το αρχείο εισόδου. Για να γίνει αυτό χρησιμοποιείται η πληροφορία που προκύπτει ύστερα από την κωδικοποίηση/συμπίεση των χρονοσειρών. Χρησιμοποιείται και πάλι το Pytorch.

Οδηγίες Εκτέλεσης/Χρήσης
Η εκτέλεση του αρχείου “reduce.py” έχει τα ίδια προαπαιτούμενα με τα προηγούμενα ερωτήματα. Η εντολή εκτέλεσης του προγράμματος έχει την εξής μορφή:

python3 reduce.py -d <dataset> -od <output_dataset_file> -oq <output_query_file> (-out_n <number of time series selected>) (-model <model_name>) (-save_model) (-n <number of time series to train on>)

Ενδεικτικά: python3 reduce.py -d datasets/nasdaq2007_17.csv -od reduced_input.csv -oq reduced_output.csv

Η συγκεκριμένη εκτέλεση απαιτεί περίπου 4 λεπτά λόγω του όγκου των “κωδικοποιήσεων” που πρέπει να γίνουν (πολλά predicts).

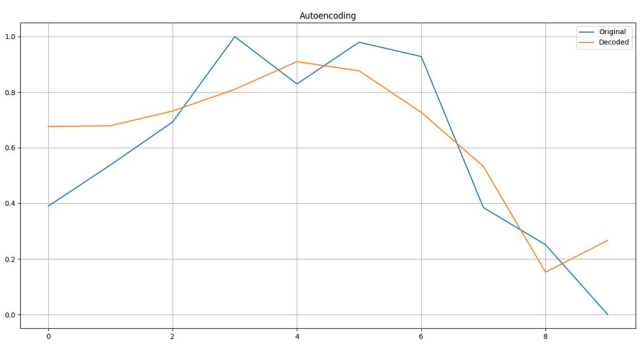
Τα ορίσματα που βρίσκονται σε παρένθεση είναι προαιρετικά. Ακολουθεί η περιγραφή της λειτουργικότητας των ορισμάτων:

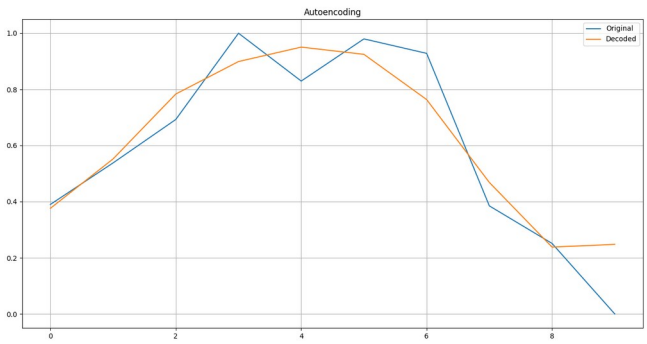
-d : Για τον προσδιορισμό του συνόλου δεδομένων
-od: Για τον προσδιορισμό του ονόματος του αρχείου εισόδου που παράγεται
-oq: Για τον προσδιορισμό του ονόματος του αρχείου ερωτημάτων που παράγεται
-save_model : Για να γίνει αποθήκευση του μοντέλου που έχει εκπαιδευτεί.
-model : Για να φορτωθεί κάποιο προεκπαιδευμένο αποθηκευμένο μοντέλο
-out_n : Για τον καθορισμό του πλήθους των χρονοσειρών που θα παραχθούν κωδικοποιήσεις.
-n : Για την επιλογή του αριθμού των χρονοσειρών που θα χρησιμοποιηθούν (για εκπαίδευση)

Πειραματισμός Εκπαίδευσης Μοντέλου

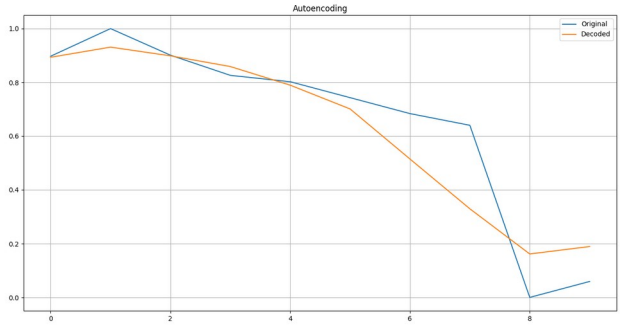
Πειραματιστήκαμε με τη συνάρτηση κόστους (loss function) χρησιμοποιώντας τόσο την binary cross entropy loss όσο και την mean square error (που αποτελεί και την σύνηθη επιλογή για προβλήματα regression). Δεν παρατηρήσαμε κάποια αξιοσημείωτη αλλαγή στα αποτελέσματα. Επίσης πειραματιστήκαμε με το μέγεθος των batches που δίνονται κατα την εκπαίδευση του μοντέλου. Και πάλι δεν παρατηρήθηκε καμία σημαντική αλλαγή ως προς την επίδοση (παρα μόνο στον αριθμό των εποχών που χρειάζονται για τη σύγκλιση).

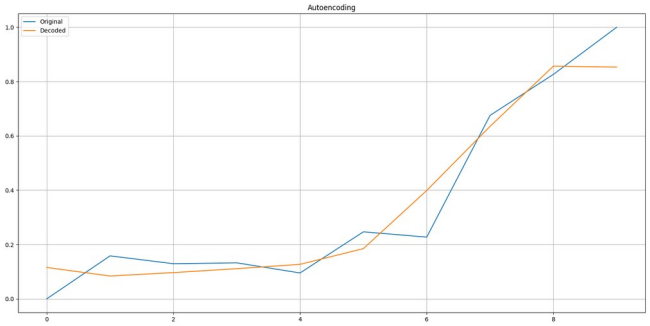
Πειραματιστήκαμε με τον αριθμό των διαστάσεων κωδικοποίησης. Επιλέγοντας έναν μεγάλο αριθμό διαστάσεων κωδικοποίησης, η συμπεριφορά της χρονοσειράς αποτυωπένεται με μεγαλύτερη λεπτομέρεια, αλλά οι τελικές διαστάσεις θα είναι σχετικά μεγάλες, πράγμα που δεν είναι το ζητούμενο. Αντίθετα, χρησιμοποιώντας σχετικά μικρές διαστάσεις κωδικοποίησης, η συμπεριφορά της χρονοσειράς δεν αποτυπώνεται με μεγάλη ακρίβεια, αλλά οι τελικές διαστάσεις που θα προκύψουν είναι πολύ λιγότερες. Ακολουθούν τα αποτελέσματα που προέκυψαν για ένα τμήμα μιας χρονοσειράς για διαστάσεις κωδικοποίησης 2 και 10 αντίστοιχα. Σε κάθε περίπτωση χρησιμοποιείται μήκος παραθύρου (συνέλιξης) ίσο με 10.

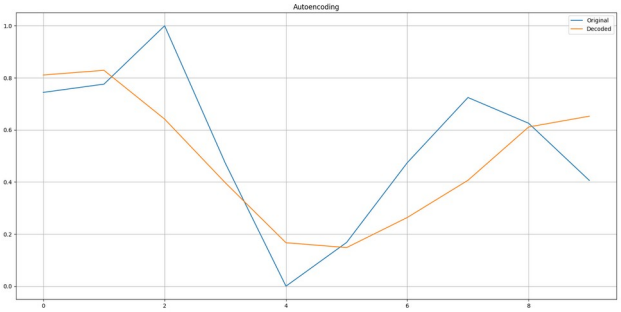


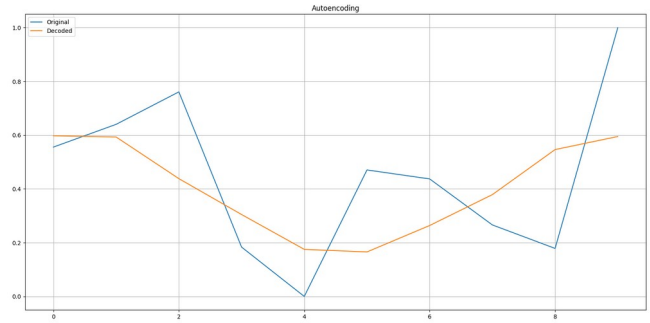


Παρατηρούμε ότι, στην αριστερά εικόνα (2 διαστάσεις κωδικοποίησης) η αποκωδικοποιημένη καμπύλη είναι περισσότερο διαφορετική από την αρχική, σε σχέση με τη δεξιά εικόνα (10 διαστάσεις). Θα επιλέξουμε μια μέση λύση ώστε και να έχουμε λίγες διαστάσεις (στις μειωμένες χρονοσειρές) και να είναι όσο καλύτερη γίνεται η αποτύπωση της συμπεριφοράς της αρχικής χρονοσειράς. Η μέση λύση θα είναι η χρήση 3 διαστάσεων κωδικοποίησης. Ακολουθούν κάποια διαγράμματα που δείχνουν πρωτότυπα και αποκωδικοποιημένα τμήματα χρονοσειρών.

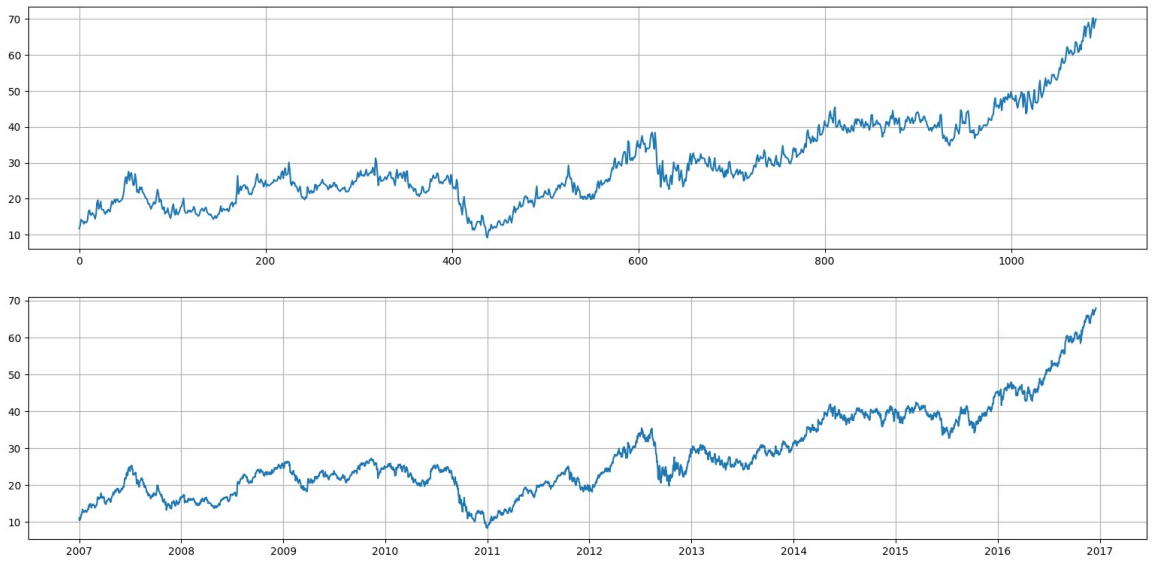








Ακολουθεί διάγραμμα που δείχνει τις μειωμένες χρονοσειρές (πάνω) και τις πρωτότυπες (κάτω). Προφανώς στην πάνω περίπτωση, το πλήθος των τιμών είναι πολύ μικρότερο από την κάτω περίπτωση.



Ερώτημα Δ

Γενική Περιγραφή

Στο συγκεκριμένο ερώτημα δοκιμάσαμε να χρησιμοποιήσουμε το δοσμένο αρχείο δεδομένων και να μειώσουμε τις διαστάσεις του μέσω του προγράμματος που αναπτύχθηκε για το προηγούμενο ερώτημα. Χρησιμοποιήσαμε τις 10 τελευταίες τιμές ως αρχείο αναζήτησης και τις υπόλοιπες ως αρχείο εισόδου. Χρησιμοποιήσαμε τα αρχεία αυτά στην προηγούμενη εργασία, τόσο στο μέρος της εύρεσης πλησιέστερων γειτόνων όσο και στο μέρος της συσταδοποίησης, και συγκρίναμε τα αποτελέσματα μεταξύ των αρχικών και των μειωμένων διαστάσεων. Τα αρχεία που χρησιμοποιήθηκαν βρίσκονται στον υποκατάλογο reduced_stock_ts κα είναι τα αρχεία “reduced_input.csv” και “reduced_query.csv”. Ακολουθούν τα αποτελέσματα της σύγκρισης.

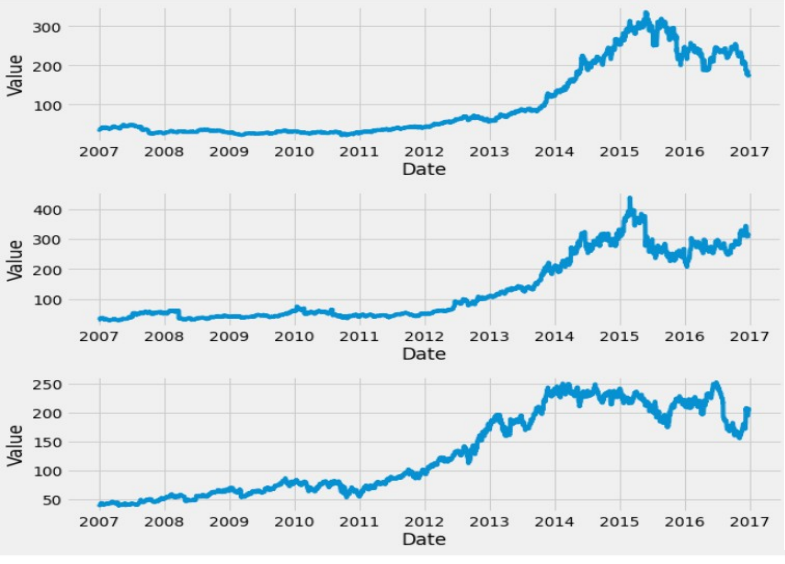
Αποτελέσματα

Στο πρόβλημα των πλησιέστερων γειτόνων, δοκιμάσαμε αρχικά να χρησιμοποιήσουμε την προσέγγιση με την διακριτή μετρική Frechet. Στο πρωτότυπο σύνολο δεδομένων (με τις πολλές διαστάσεις) η αναζήτηση ήταν σχετικά χρονοβόρα (κυρίως με τη χρήση της προσέγγισης ωμής βίας, όπου χρειάστηκαν κατα μέσο όρο 53.33 δευτερόλεπτα για κάθε query), πράγμα το οποίο οφείλεται στον μεγάλο αριθμό διαστάσεων αφού ο υπολογισμος της απόστασης είναι υπολογιστικά ακριβός. Στις μειωμένες διαστάσεις, η αναζήτηση χρειάστηκε πολύ λιγότερο χρόνο για κάθε σημείο αναζήτησης (μέσος όρος 4.89 δευτερόλεπτα για την ωμή βία) και η αναζήτηση LSH είχε μεγαλύτερη ακρίβεια αφού βρέθηκαν για περισσότερα σημεία κοντινοί γείτονες. Φυσικά, αφού οι διαστάσεις είναι μικρότερες, αυξάνεται η πιθανότητα τα σημεία να θεωρούνται κοντινά και περιορίζεται, σε μεγάλο βαθμό, η Κατάρα της Διαστατικότητας. Η χρονική διαφορά είναι ακόμα μεγαλύτερη στον υπολογισμό της συνεχούς απόστασης Frechet όπου γνωρίζουμε ότι είναι ιδιαίτερα κοστοβόρα καθώς αυξάνονται οι διαστάσεις. Όσον αφορά στο ποιοί είναι οι κοντινότεροι γείτονες, παρατηρήσαμε οτι στην πλειοψηφία τα αποτελέσματα είναι ίδια με το πρωτότυπο και το μειωμένο σύνολο δεδομένων που σημαίνει ότι το μειωμένων διαστάσεων σύνολο δεδομένων αποτυπώνει σε μεγάλο βαθμό το πρωτότυπο. Συμπερασματικά, στο συγκεκριμένο πρόβλημα η μείωση των διαστάσεων μπορεί να βοηθήσει σημαντικά τόσο στην επίτευξη χαμηλότερων χρόνων αναζήτησης όσο και στον περιορισμό της ανομοιότητας των σημείων.

Ανάλογα ήταν και τα αποτελέσματα στο πρόβλημα της συσταδοποίησης των σημείων. Αρχικά, χρησιμοποιώντας ως μέθοδο επαναπροσδιορισμού των κεντροειδών την ευκλείδια απόσταση, παρατηρήσαμε μικρή αλλαγή στον χρόνο εκτέλεσης (καθώς η μέθοδος από την φύση της δεν είναι αρκετά χρονοβόρα). Τα αποτελέσματα της μετρικής της σιλουέτας ήταν παρόμοια σε σχέση με το πρωτότυπο σύνολο δεδομένων. Στην περίπτωση όμως που για τον επαναπροσδιορισμό των κεντροειδών χρησιμοποιείται ως μετρική η απόσταση Frechet, η διαφορά στον χρόνο ήταν σημαντική. Οι λόγοι είναι και πάλι οι ίδιοι, ότι δηλαδή με πολλές διαστάσεις ο υπολογισμός της Frechet (που γίνεται μέσω δυναμικού προγραμματισμού) είναι αρκετά πιο χρονοβόρος. Ο δυναμικός προγραμματισμός έχει τετραγωνική πολυπλοκοτητα και η άυξηση της διάστασης της εισόδου μπορεί να προκαλέσει σημαντική αύξηση του χρόνου εκτέλεσης. Γενικά στον τρόπο που σχηματίζονται οι συστάδες, τα αποτελέσματα είναι σε μεγάλο βαθμό παρόμοια όποιο σύνολο εισόδου και αν χρησιμοποιήσουμε.

Δοκιμάσαμε να οπτικοποιήσουμε χρονοσειρές που ανήκουν στην ίδια συστάδα και να τις αντιπαραβάλλουμε με χρονοσειρές μίας διαφορετικής συστάδας. Πράγματι, οι συγκεκριμένες χρονοσειρές φαίνονται σωστά ομαδοποιημένες.

Πρώτη Συστάδα



Δεύτερη Συστάδα