Report Εργασίας 3 Ανάπτυξης Λογισμικού Αλγοριθμικών Προβλημάτων

Κωνσταντίνος Τσικούρης sdi1800201, Βασίλειος Βασιλάκης sdi1800018

Δεκέμβριος 2021- Ιανουάριος 2022

Περιεχόμενα

1 (A) **2**

2 (B) **9**

3 (Γ) **12**

4 (Δ) **17**

1 (A)

Αρχιτεκτονική-Υπερπαράμετροι:

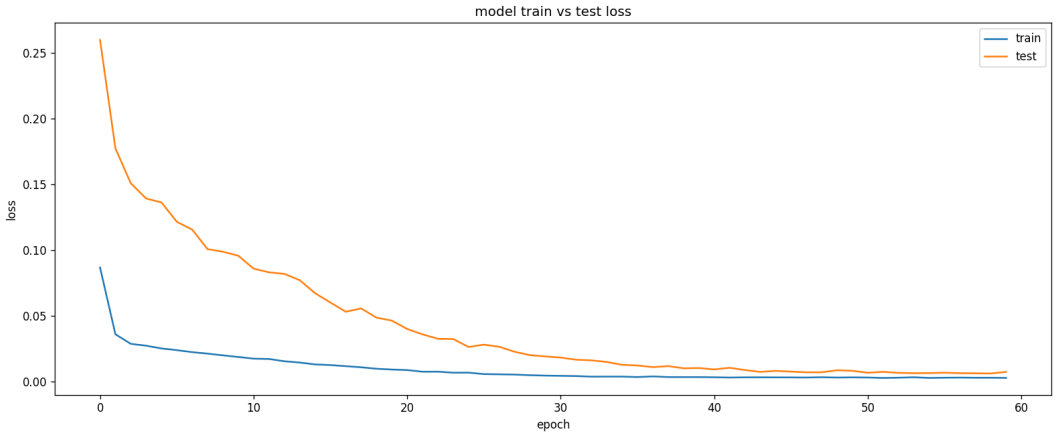
Για το πρόβλημα του forecasting, ξεκινήσαμε με ένα πολύ απλό μοντέλο, ένα stacked LSTM μοντέλο δύο επιπέδων με number of hidden units 64 και με πιθανότητα dropout 0.3 σε κάθε επίπεδο, και output layer, ένα dense layer με ένα μόνο unit, δεδομένου ότι το πρόβλημά μας, είναι πρόβλημα forecast της επόμενης τιμής μιας ακολουθίας, δεδομένου των προηγούμενων τιμών (άρα έχουμε έξοδο του μοντέλου μία μόνο αριθμητική τιμή). Για το window του sampling χρησιμοποιήσαμε την τιμή $w = 100$, δηλαδή χωρίζουμε την κάθε χρονοσειρά σε υπο-ακολουθίες 100 διαδοχικών τιμών, και θέλουμε να προβλέψουμε την επόμενη τιμή. Επίσης δεδομένου ότι έχουμε ένα regression πρόβλημα, χρησιμοποιήσαμε την mean squared error σαν loss function, και stochastic gradient descent σαν optimizer έναντι του adam, αφού έδινε πιο σταθερά αποτελέσματα (ως προς τη μείωση του loss σε κάθε εποχή). Επίσης εκπαιδεύσαμε για 60 εποχές με batch size 256

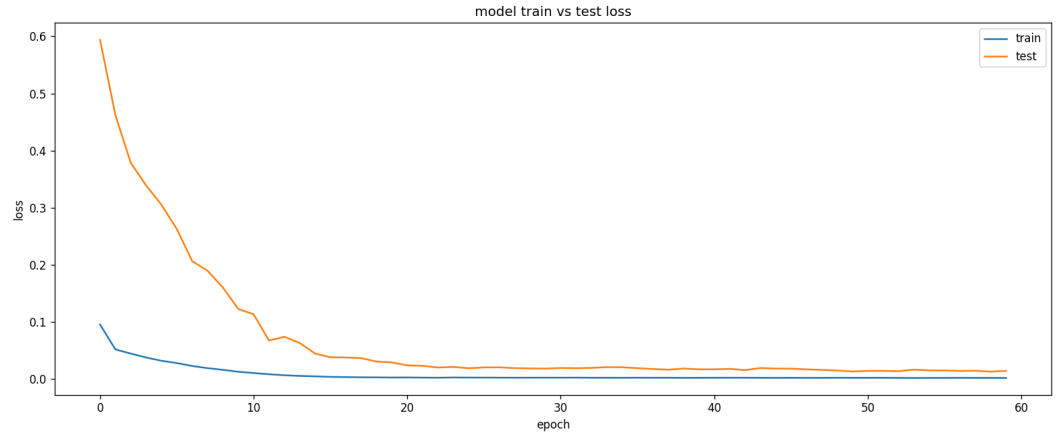
Αποτελέσματα-Σχολιασμός:

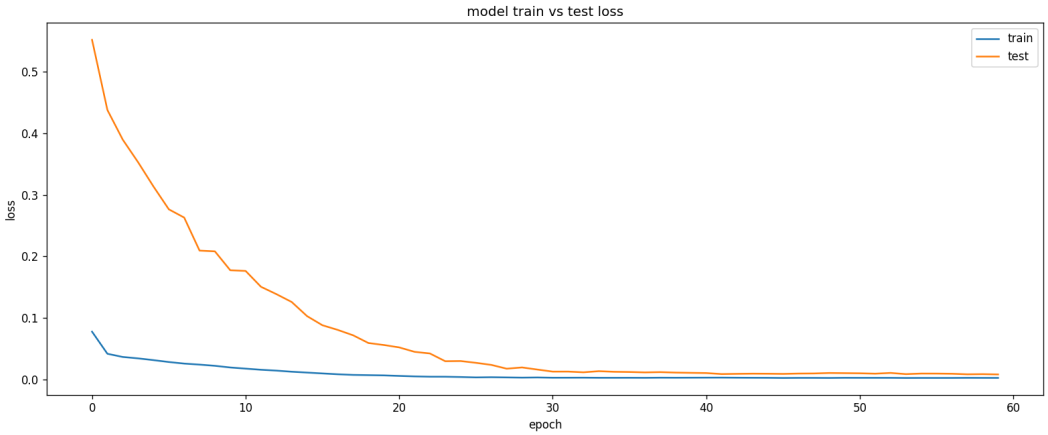
Εκπαιδεύσαμε το μοντέλο τόσο ανά χρονοσειρά όσο και ανά σύνολο χρονοσειρών, για τις πρώτες 20 χρονοσειρές (για μεγαλύτερο αριθμό χρονοσειρών θα ήταν overkill από άποψη χρόνου και πλήθους μοντέλων). Τα αποτελέσματα που παρατηρήθηκαν είναι τα ακόλουθα:

Ανά χρονοσειρά:

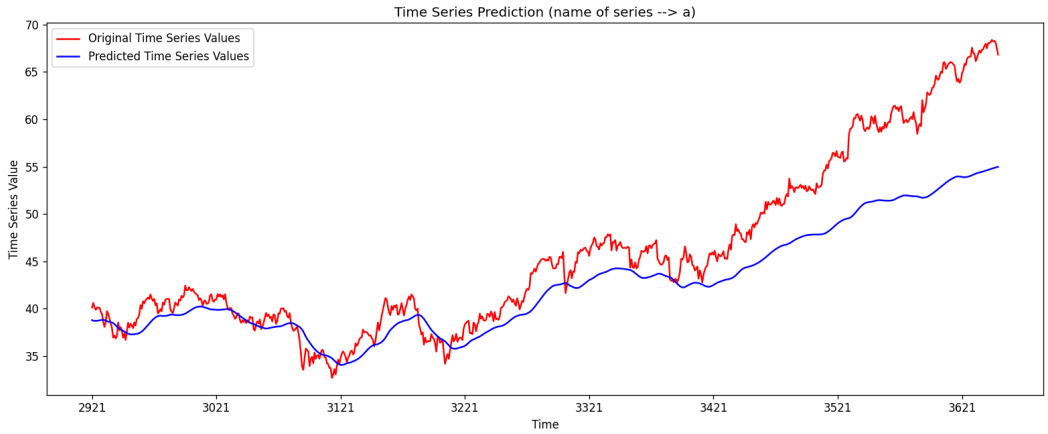
Τα μοντέλα ανά χρονοσειρά είχαν αρκετά καλή απόδοση παρά την απλότητα του μοντέλου. Οι καμπύλες μάθησης (loss vs epochs) για το training set και για το validation set παρατηρήθηκαν ότι ήταν πολύ κοντά μεταξύ τους στο τέλος της εκπαίδευσης, με τις καμπύλες να μοιάζουν να επιπεδοποιούνται σχεδόν, κάτι που υποδηλώνει ότι η απόδοση του μοντέλου τόσο σε δεδομένα που έχει ξαναδεί, όσο και σε δεδομένα που δεν έχει ξαναδεί, παραμένει σταθερή ακόμα και με αύξηση των εποχών εκπαίδευσης. Τα παραπάνω υποδηλώνουν ότι έχουμε πολύ καλό fit, δεν έχουμε overfit, και πετυχαίνουμε και πολύ χαμηλό training/validation loss κάτι που θέλουμε. Μερικές ενδεικτικές καμπύλες μάθησης για κάποια εκ των μοντέλων ανά χρονοσειρά (aaba, aapl, abc):

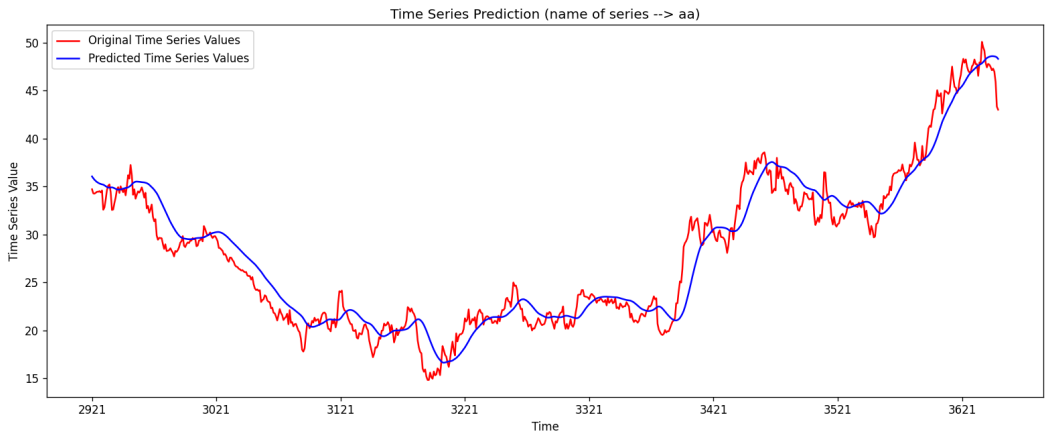


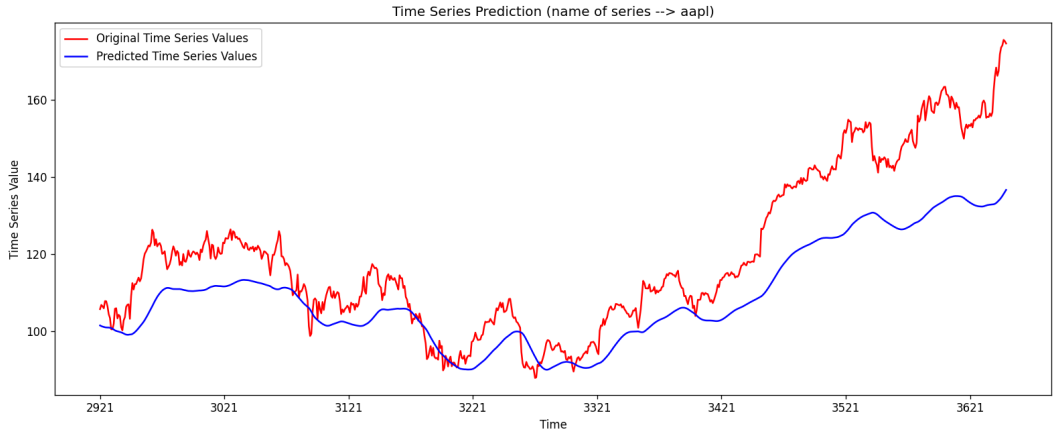


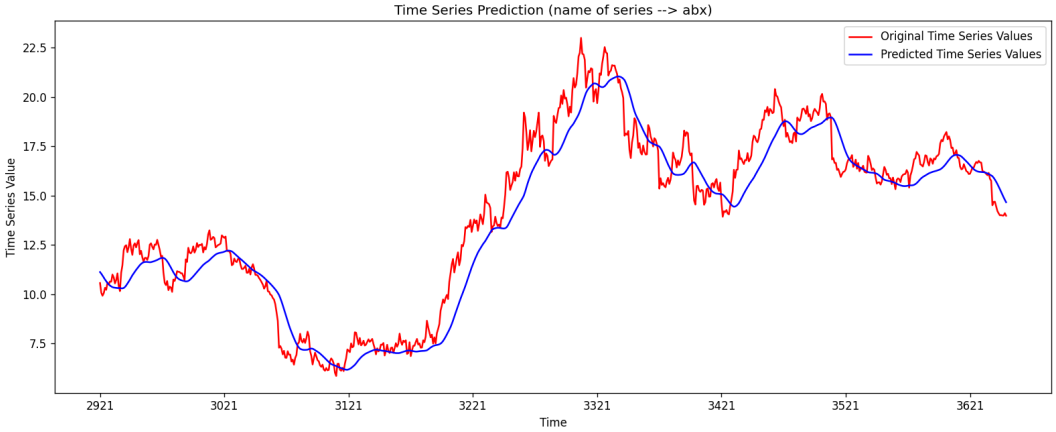


Ως προς τις προβλέψεις τώρα, πάλι τα μοντέλα ανά χρονοσειρά τα πηγαίνουν ικανοποιητικά. Οι predicted χρονοσειρές ακολουθούν σε γενικό βαθμό τις πραγματικές χρονοσειρές, χωρίς να προβλέπουν γρήγορα όλες τις απότομες αυξομειώσεις των πραγματικών χρονοσειρών, κάτι που περαιτέρω υποστηρίζει ότι πράγματι δεν έχουμε overfitting. Ωστόσο σε μερικές παρατηρείται ότι είναι χαμηλότερα από τις πραγματικές (καλό prediction ως προς τον άξονα των x, όχι τόσο καλό ως προς τον άξονα των y). Μερικές ενδεικτικές καμπύλες για κάποια εκ των μοντέλων ανά χρονοσειρά:



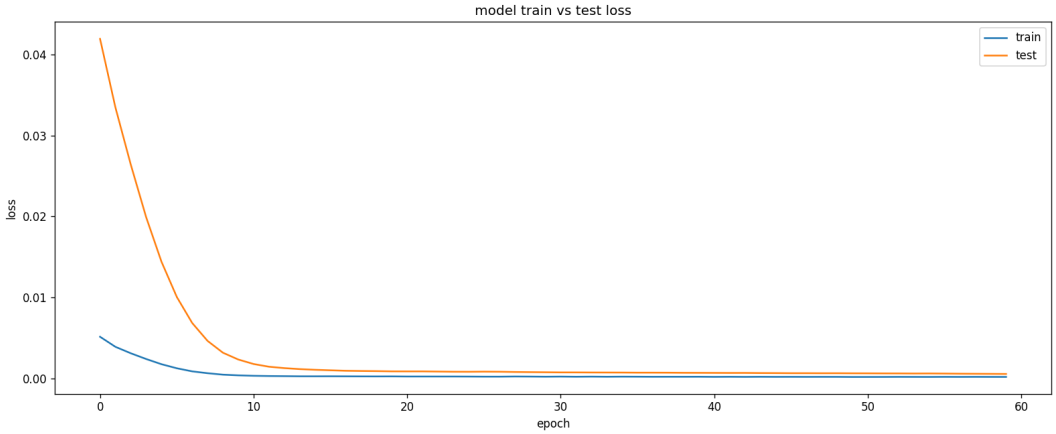




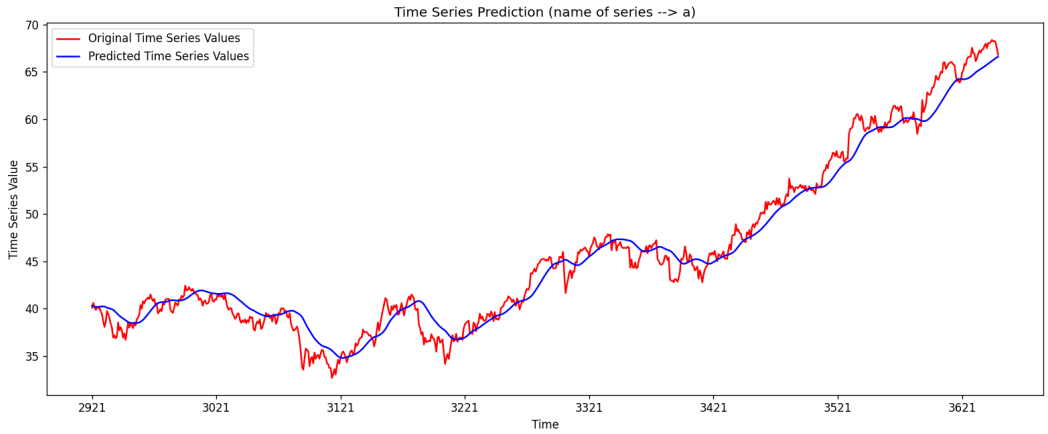


Ανά σύνολο χρονοσειρών:

Το μοντέλο που εκπαιδεύτηκε ως προς το σύνολο των χρονοσειρών, είχε εξίσου καλή καμπύλη μάθησης, με τις καμπύλες του training και validation loss να είναι πολύ κοντά μεταξύ τους, και να επιπεδοποιούνται πάλι στο τέλος του training, δηλαδή πάλι έχουμε σύγκλιση, καλό fit και όχι overfitting. Επίσης το training/validation error είναι πάλι σε χαμηλό επίπεδο.



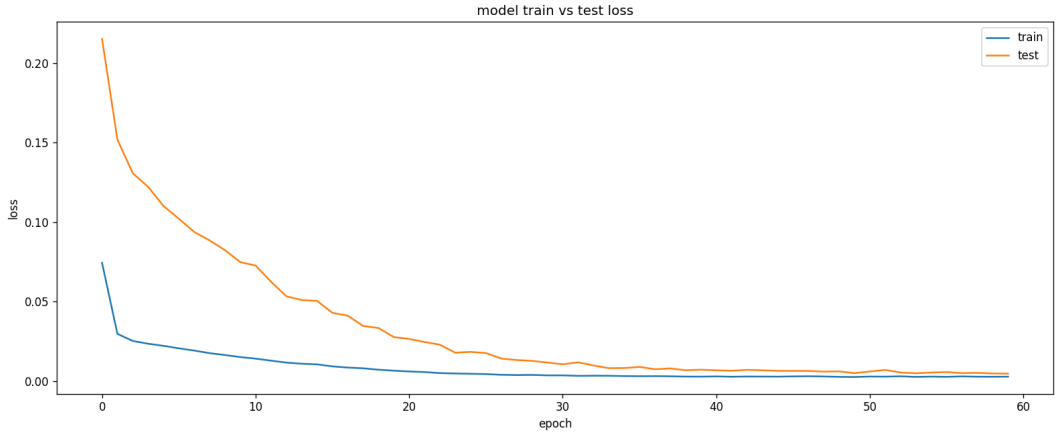
Το μοντέλο που εκπαιδεύτηκε στο σύνολο των χρονοσειρών όμως παρατηρήθηκε ότι είχε καλύτερη συμπεριφορά, αφού βοήθησε λίγο στο προαναφερόμενο θέμα, δηλαδή μείωσε σε μερικές χρονοσειρές την απόσταση ως προς τον άξονα των y, των predicted από τις πραγματικές καμπύλες. Για ένα ενδεικτικό παράδειγμα συγκρίνετε την predicted χρονοσειρά a παραπάνω όπως προβλέφθηκε από το μοντέλο ανά χρονοσειρά, με την predicted χρονοσειρά a παρακάτω όπως προβλέφθηκε από το μοντέλο ανά σύνολο:

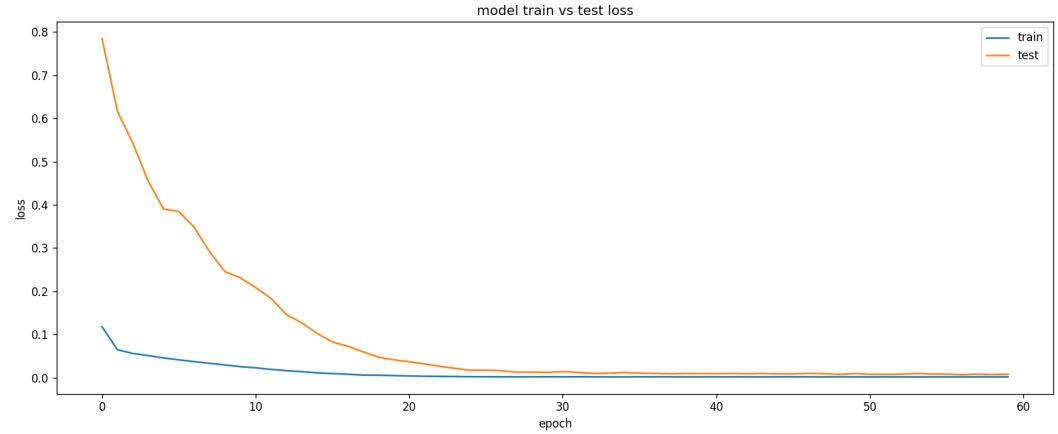


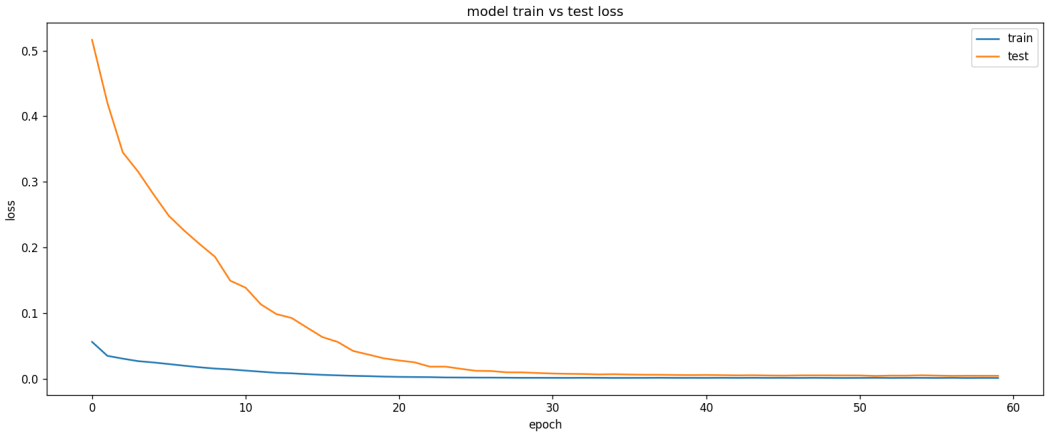
Παρατηρούμε ότι μια μικρή απόκλιση της predicted από την πραγματική χρονοσειρά που υπήρχε στο μοντέλο ανά χρονοσειρά, έχει πλέον καλυφθεί από το μοντέλο ανά σύνολο. Τέτοιες βελτιώσεις παρατηρήθηκαν και σε άλλες χρονοσειρές, όχι όμως σε όλες, σε αρκετές η απόδοση/πρόβλεψη ήταν η ίδια.

Hyperparameter Tuning: Για να βελτιώσουμε λίγο τα μοντέλα, πειραματιστήκαμε κατ'αρχήν με προσθήκη ενός παραπάνω layer, χωρίς αυτό να οδηγήσει σε ιδιαίτερη βελτίωση. Η αύξηση των hidden units όμως κάθε επιπέδου από 64 σε 100, παρατηρήθηκε ότι άμβλυνε το προαναφερόμενο πρόβλημα ακόμα περισσότερο, αφού σε μερικές χρονοσειρές που είχαν απόκλιση, μετατόπισε τις predicted χρονοσειρές ακόμα πιο κοντά στις πραγματικές χρονοσειρές, μια βελτίωση που παρατηρήθηκε τόσο στα μοντέλα ανά χρονοσειρά, όσο και στο μοντέλο στο σύνολο. Παράλληλα, οι καμπύλες μάθησης των tuned μοντέλων τόσο ανά χρονοσειρά όσο και στο σύνολο, παρέμειναν εξίσου καλές, όπως φαίνεται και στα παρακάτω διαγράμματα:

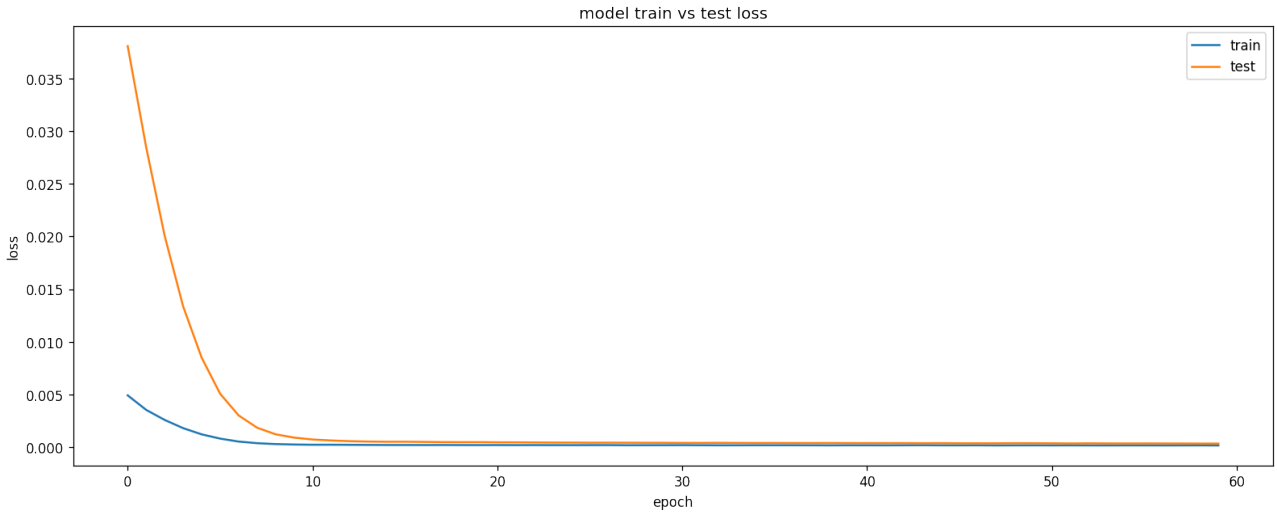
Καμπύλες μάθησης του tuned μοντέλου, για τα μοντέλα ανά χρονοσειρά aaba, aapl, abc:



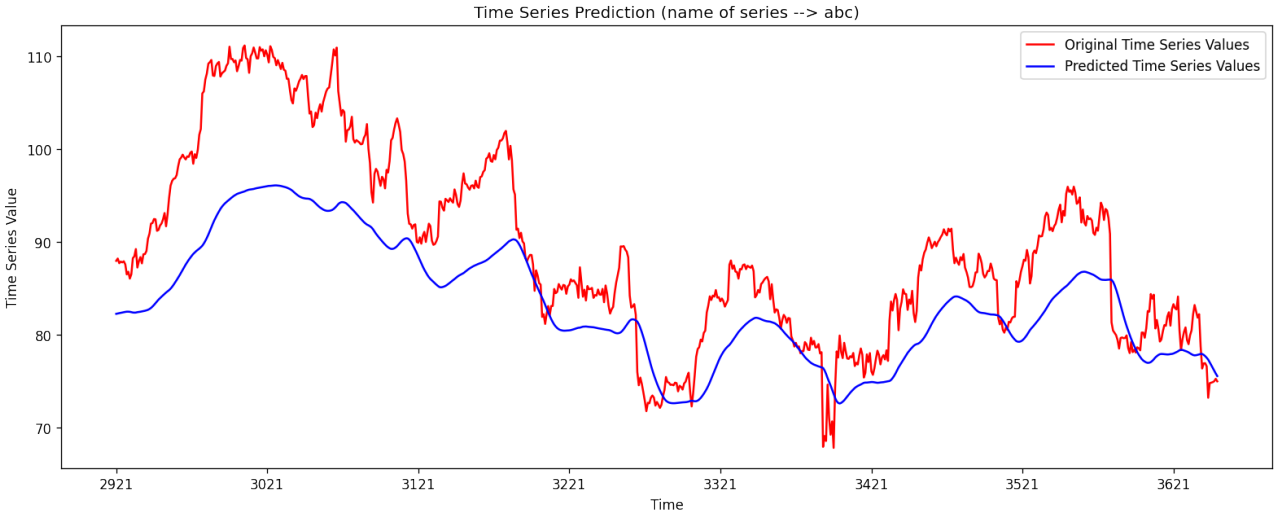


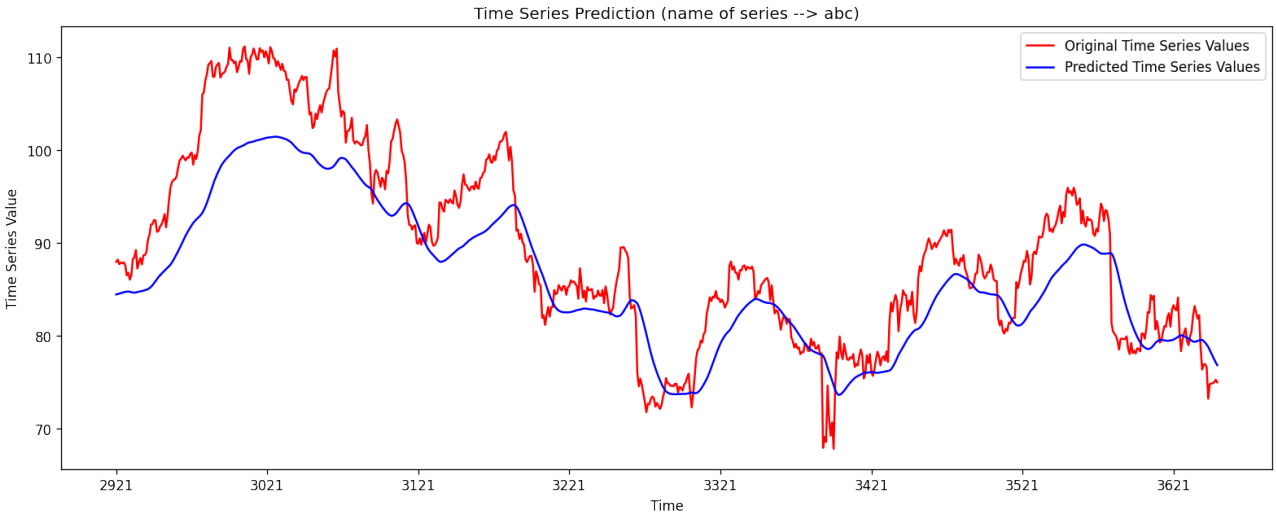


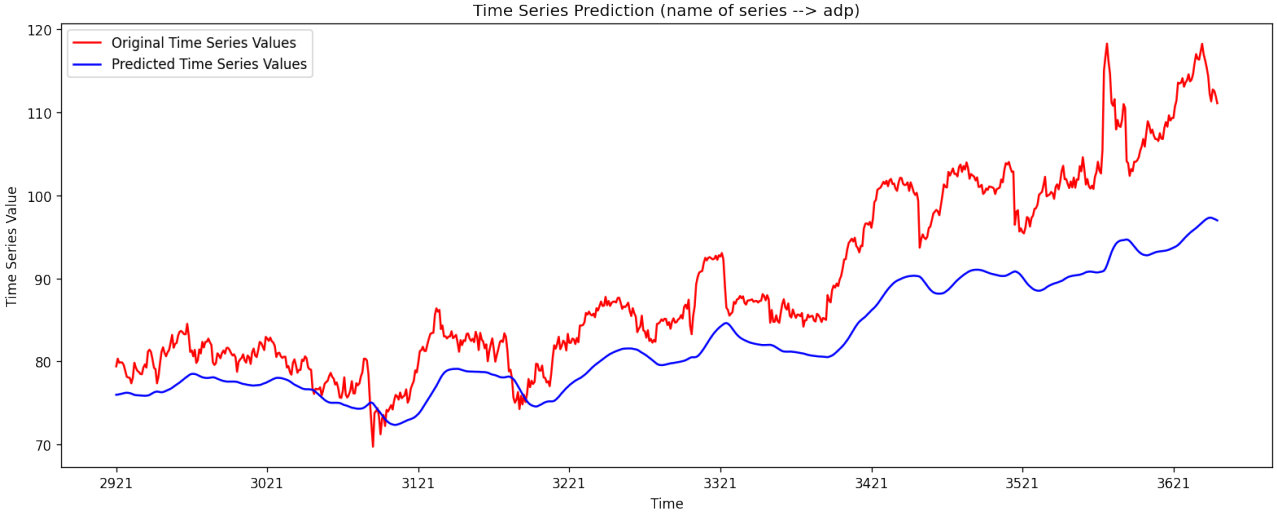
Καμπύλη μάθησης του tuned μοντέλου, για το μοντέλο στο σύνολο:

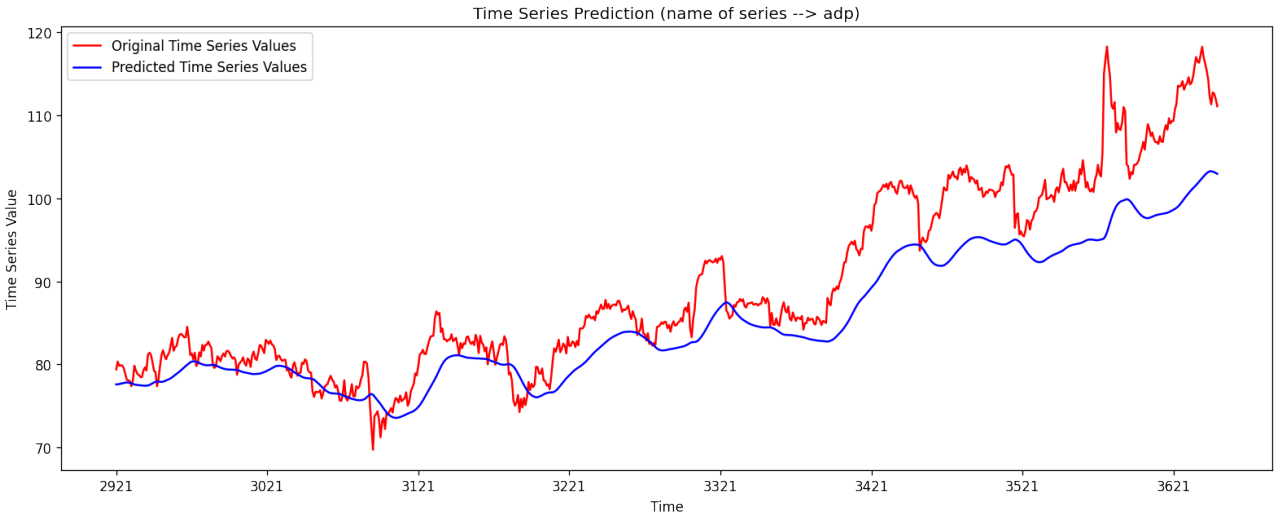


'Οσον αναφορά τις βελτιώσεις που πήραμε από το tuned μοντέλο, συγκρίνετε παρακάτω μερικά μοντέλα ανά χρονοσειρά πριν (όπου κάθε layer είχε 64 units) και μετά (όπου κάθε layer έχει 100 units) :

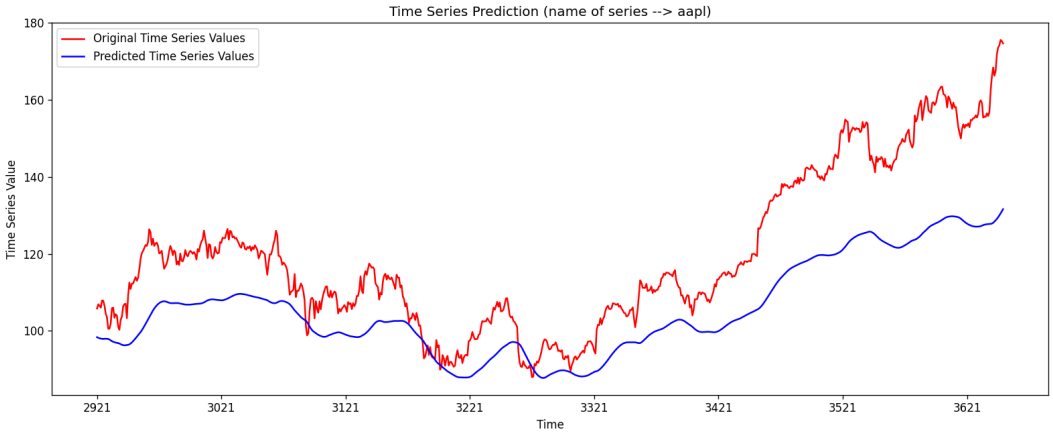


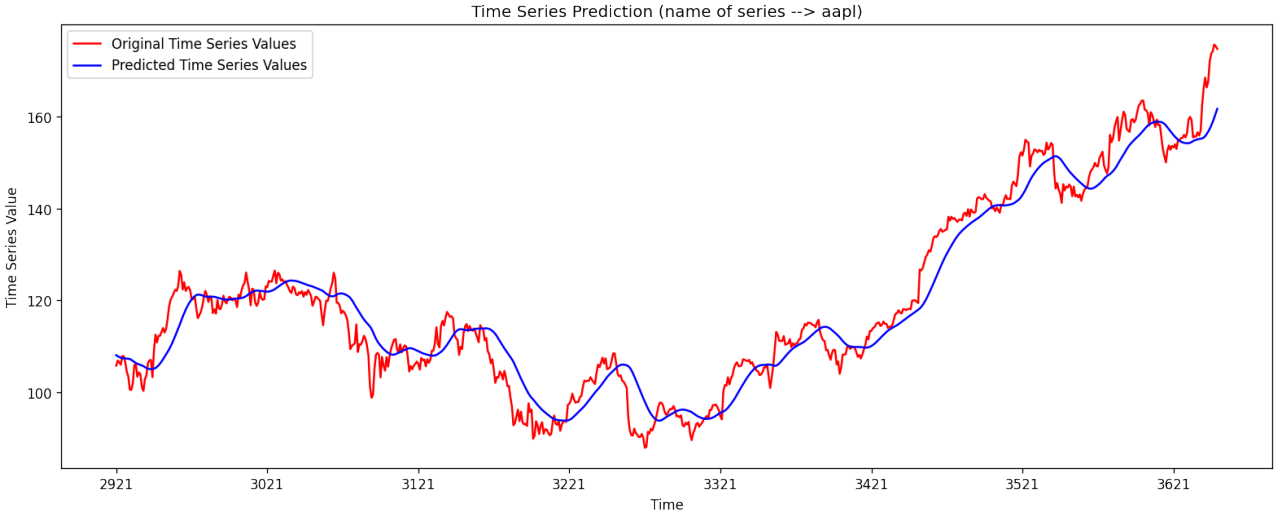






Όπως αναφέρθηκε ίδιες βελτιώσεις παρατηρήθηκαν και ανάμεσα στα μοντέλα ανά σύνολο. πχ Συγκρίνετε την πρόβλεψη του μοντέλου στο σύνολο για την aapl πριν (64 units) με την πρόβλεψη του μοντέλου στο σύνολο για την aapl μετά (100 units):


Time Series Prediction (name of series --> aapl)
Original Time Series Values
Predicted Time Series Values
Time Series Value
Time
180
160
140
120
100
2921
3021
3121
3221
3321
3421
3521
3621


Time Series Prediction (name of series --> aapl)
Original Time Series Values
Predicted Time Series Values
Time Series Value
Time
160
140
120
100
2921
3021
3121
3221
3321
3421
3521
3621

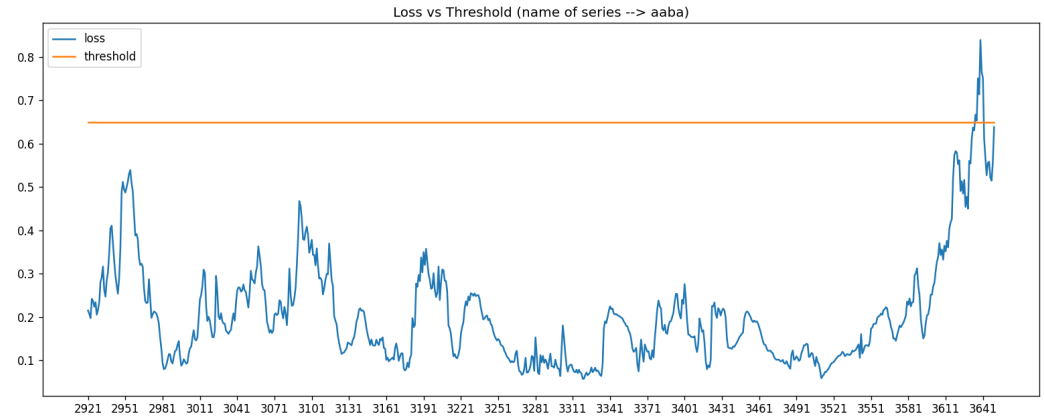
2 (B)

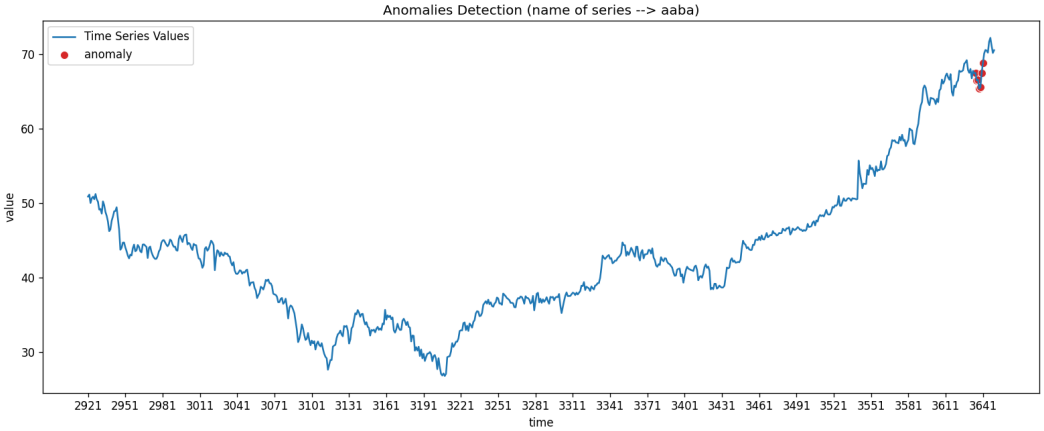
Αρχιτεκτονική-Υπερπαράμετροι:

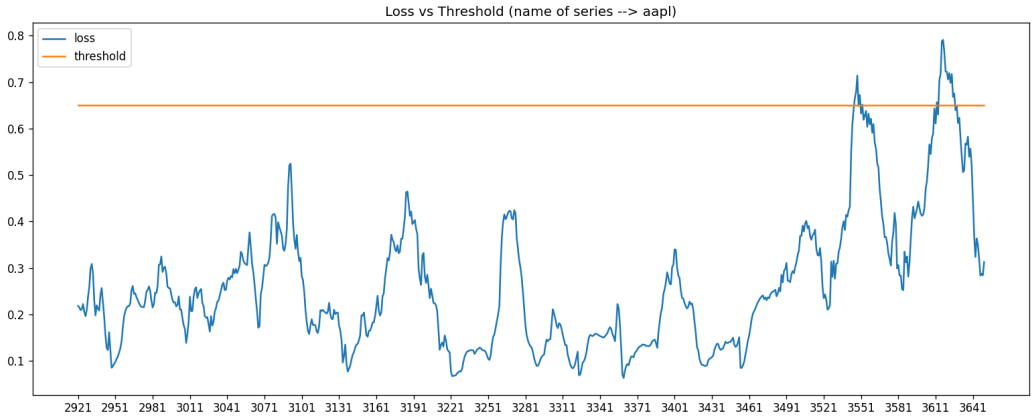
Για το πρόβλημα του anomaly detection, χρησιμοποιήθηκε ένα LSTM autoencoder μοντέλο, με ένα επίπεδο encoding και ένα επίπεδο decoding με number of hidden units 64 το καθένα και με πιθανότητα dropout 0.2 σε κάθε επίπεδο, και output layer, ένα dense layer με ένα μόνο unit, δεδομένου ότι το πρόβλημά μας, είναι πρόβλημα αναπαραγωγής της επόμενης τιμής μιας ακολουθίας, δεδομένου των προηγούμενων τιμών (άρα έχουμε έξοδο του μοντέλου μία μόνο αριθμητική τιμή) και μετά σύγκριση της με την πραγματική επόμενη τιμή για να υπολογισθεί το reconstruction error. Για το window του sampling χρησιμοποιήσαμε την τιμή $w = 30$, δηλαδή χωρίζουμε την κάθε χρονοσειρά σε υπο-ακολουθίες 30 διαδοχικών τιμών, και θέλουμε να προβλέψουμε την επόμενη τιμή. Επίσης, επειδή η εκπαίδευση έγινε τώρα σε όλο το σύνολο των 359 χρονοσειρών, και ακόμα και για αυτό το απλό μοντέλο ήταν πολύ αργή, εκπαιδεύσαμε για μόλις 10 εποχές με batch size 32. Με περισσότερες εποχές θα είχαμε αναμενόμενα μικρότερο reconstruction loss για αρκετές από τις χρονοσειρές. Επίσης με μεγαλύτερα batch sizes παρατηρήθηκε ότι το validation loss ξεκινούσε πολύ ψηλότερα από το training loss και κατά συνέπεια σε 10 εποχές δεν προλάβαινε να υπάρξει κανένα είδος σύγκλισης. Επίσης χρησιμοποιήθηκε ο adam εδώ σαν optimizer, και σαν loss function το mean absolute error, εφόσον μας ενδιαφέρει το minimization του reconstruction error.

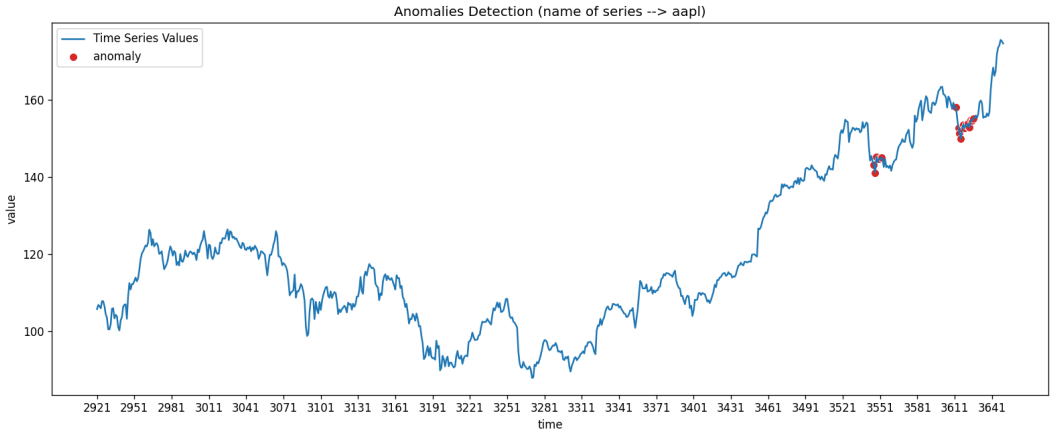
Αποτελέσματα-Σχολιασμός:

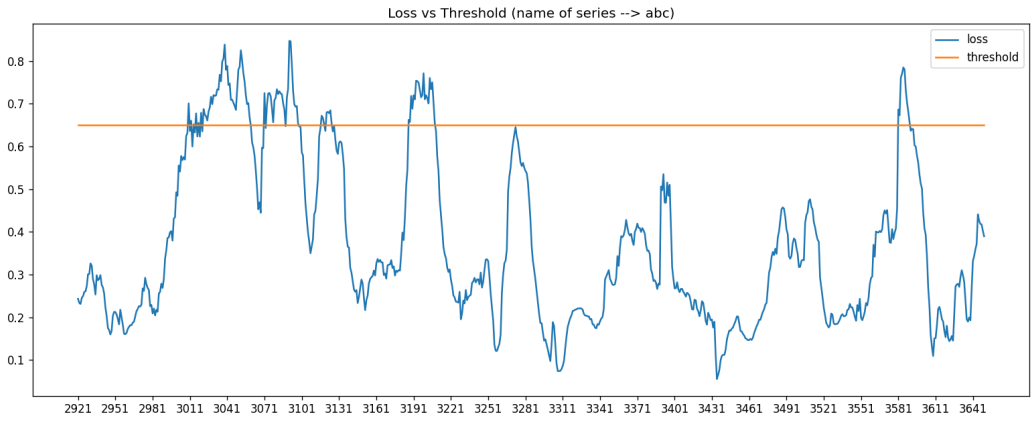
Τα αποτελέσματα του μοντέλου ήταν ικανοποιητικά, με το validation loss να πέφτει γύρω στο 0.40. Για κάποιες εκ των χρονοσειρών το reconstruction error ήταν αρκετά μικρό, οπότε έχουν λίγες έως καθόλου anomalies. Ωστόσο υπάρχουν και χρονοσειρές που έχουν αρκετά μεγάλο reconstruction error, και συνεπώς έχουν αρκετά περισσότερες και πυκνότερες anomalies. Απόπειρες βελτίωσης του μοντέλου, με προσθήκη επιπλέον επιπέδων encoding/decoding, με διαφορετικό πλήθος units, πέρα από το ότι καθυστερούσαν υπερβολικά το training (κάτι λογικό αφού τώρα κάνουμε training σε όλες τις χρονοσειρές) κάτι που καθιστούσε το fine tuning αρκετά επίπονο, δεν παρατηρήθηκε να επιφέρουν ιδιαίτερη βελτίωση, με το validation loss να ξεκινάει πάλι πολύ ψηλότερα, και συνεπώς να μην καταφέρνει να πέσει αρκετά χαμηλά μετά από 10 εποχές. Μερικές ενδεικτικές γραφικές παραστάσεις για κάποιες χρονοσειρές με threshold mae = 0.65:

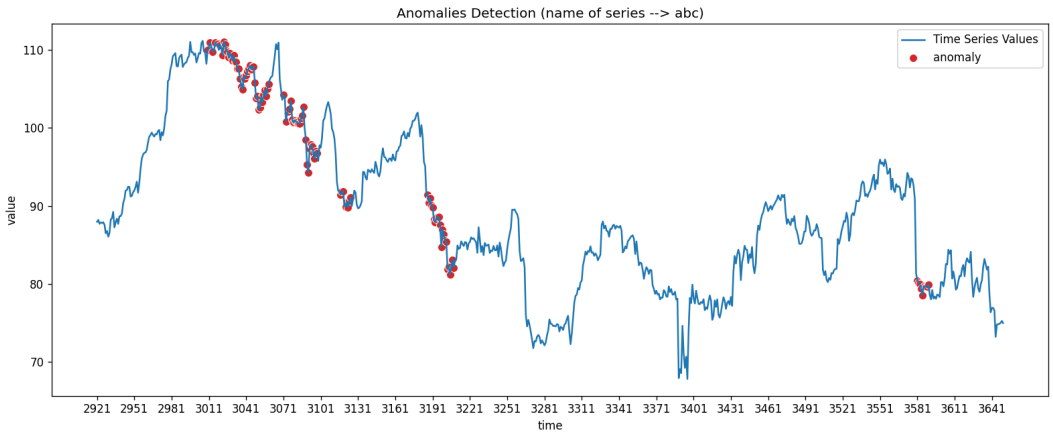





Loss vs Threshold (name of series --> aapl)
loss
threshold
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
2921 2951 2981 3011 3041 3071 3101 3131 3161 3191 3221 3251 3281 3311 3341 3371 3401 3431 3461 3491 3521 3551 3581 3611 3641


Anomalies Detection (name of series --> aapl)
Time Series Values
anomaly
value
160
140
120
100
2921 2951 2981 3011 3041 3071 3101 3131 3161 3191 3221 3251 3281 3311 3341 3371 3401 3431 3461 3491 3521 3551 3581 3611 3641
time


Loss vs Threshold (name of series --> abc)
loss
threshold
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
2921 2951 2981 3011 3041 3071 3101 3131 3161 3191 3221 3251 3281 3311 3341 3371 3401 3431 3461 3491 3521 3551 3581 3611 3641

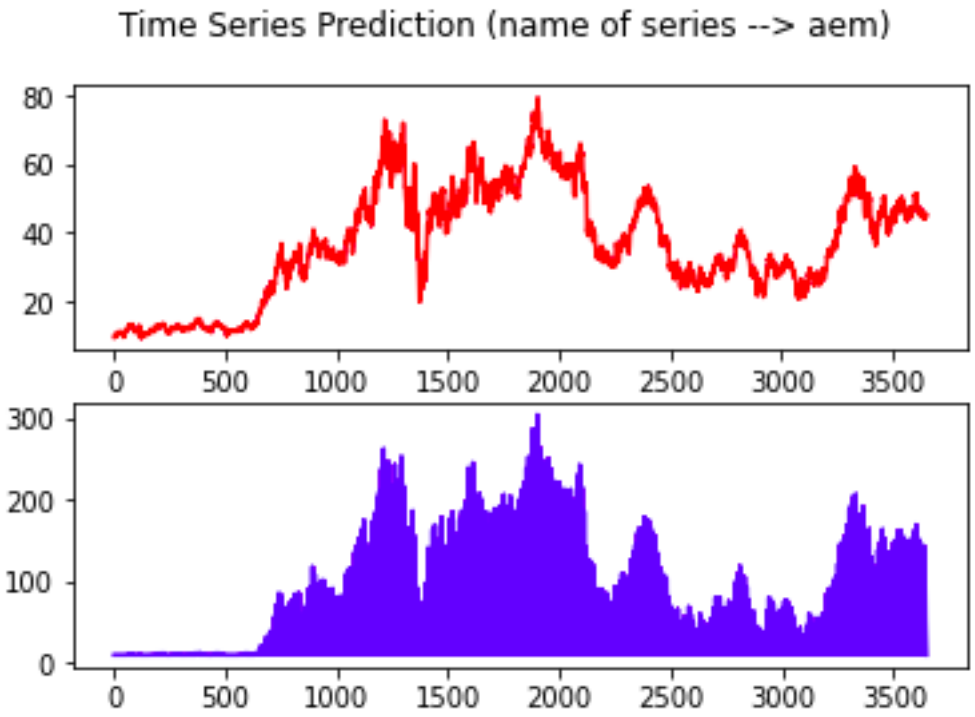

Anomalies Detection (name of series --> abc)
Time Series Values
anomaly
value
110
100
90
80
70
2921 2951 2981 3011 3041 3071 3101 3131 3161 3191 3221 3251 3281 3311 3341 3371 3401 3431 3461 3491 3521 3551 3581 3611 3641
time

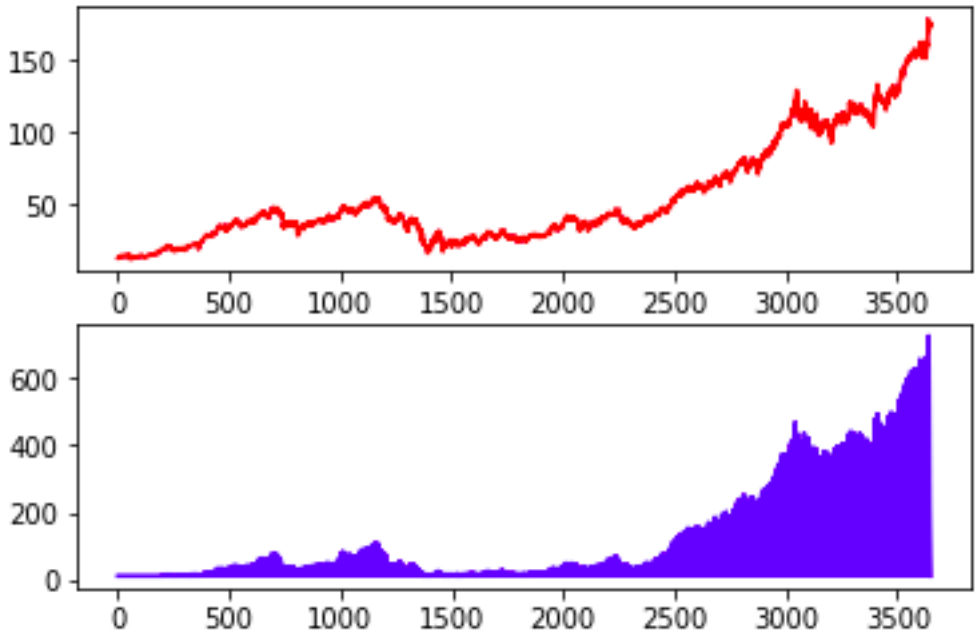
3 (Γ)

Αρχιτεκτονική-Υπερπαράμετροι:

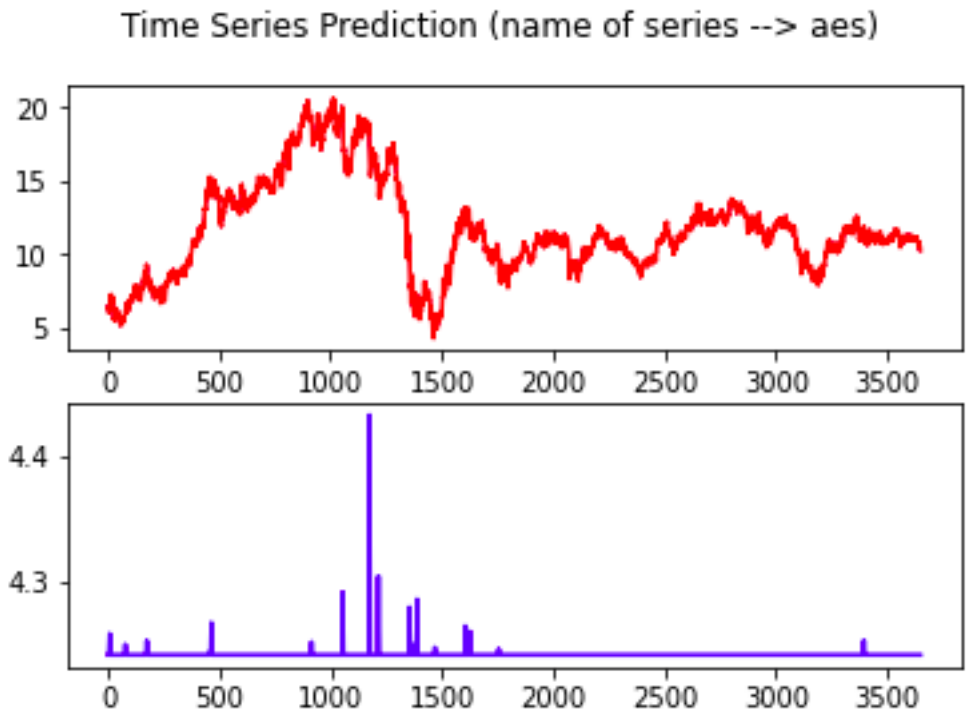
Για το πρόβλημα του encoding, χρησιμοποιήθηκε ένα convolution autoencoder μοντέλο, με 2 επίπεδα **cond1d**, 1 επίπεδο maxpooling1d πριν το **encoding** επίπεδο που υλοποιείται με ένα flatten και ένα dense layer. Έπειτα ακολουθούν dense, reshape επίπεδα που σχεδόν επαναφέρουν τις διαστάσεις σε αυτό πριν το flatten επίπεδο και έπειτα εμφανίζεται ένα upsampling και ένα conv1d layer για να επαφερθούν οι διαστάσεις σε αυτές της εισόδου. Για sampling χρησιμοποιήθηκαν οι default τιμές του window length και του latent dimension (encoding dimension) με αποτέλεσμα οι χρονοσειρές να συμπιέζονται στα 3/10 του αρχικού μεγέθους τους. Η εκπαίδευση των χρονοσειρών είναι αρκετά γρήγορη σε αυτό το μοντέλο χωρίς να υπάρχει η ανάγκη για ένα πολύ μεγάλο αριθμό εποχών κατά το training του μοντέλου (1 λεπτό σχεδόν για την ολοκλήρωση του και εκτύπωσης αποτελεσμάτων). Χρησιμοποιήθηκαν 30 εποχές. Με περισσότερες εποχές, το loss μειώνεται περαιτέρω το οποίο loss function εδώ είναι το binary_crossentropy. Ωστόσο, παρατηρήθηκε ότι στο κομμάτι του πόσο όμοιες είναι οι encoding χρονοσειρές με τις αρχικές, η loss function, και να μειωθεί περαιτέρω δεν επηρεάζει τα αποτελέσματα ως προς το καλύτερο, μάλιστα μπορεί και να τα χειροτερέψει. Επίσης, με περισσότερα conv1d layers παρατηρήθηκε ότι οι encoding χρονοσειρές που βγάινουν τείνουν στο να έχουν σε όλα τα σημεία τους μια σταθερή τιμή, άρα τα πολλά conv1d layers δημιουργούν ανεπιθύμητη συμπεριφορά. Αυτό ενδεχομένως συμβαίνει διότι οι convolutional autoencoders τείνουν να ανιχνεύουν σημεία των χρονοσειρών όπου θα υπάρξει κάποια απότομη αλλαγή. Όσα περισσότερα επίπεδα τόσο υψηλότερες απαιτήσεις στο τι πρέπει να ανιχνευθεί σε μια χρονοσερειρά. Επειδή οι απαιτήσεις είναι υψηλές, δεν ανιχνεύει κάτι και ως αποτέλεσμα, οι χρονοσειρές που προκύπτουν έχουν όλα τους τα σημεία να είναι περίπου μια σταθερή τιμή. Άρα χρειάζονται λίγα conv1d layers στο μοντέλο, όχι πολλά. Επιλέχθηκε batch size 1024. Αρχικά ήταν 256 αλλά σε μερικά tests, το 1024 φαίνεται να βγάζει λίογ καλύτερες προσεγγίσεις, αλλά δεν υπάρχει κάποια μεγάλη διαφορά. Πιο συγκεκριμένα, το σχήμα των encoded καμπυλών είναι το ίδιο αλλά στο μικρότερο batchsize, οι χρονοσειρές φαίνεται να έχουν λίγο μια μεγαλύτερη απόκλιση από τα σημεία της αρχικής καμπύλης

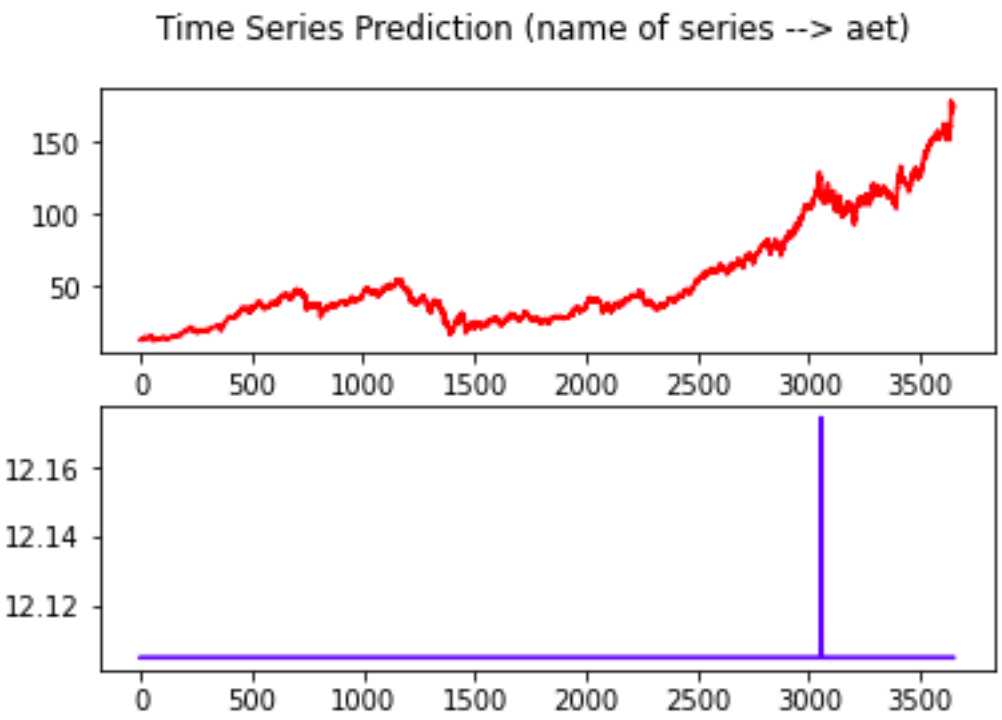
Δυστυχώς, με οποιεσδήποτε παραμέτρους και να τρέξει ο κώδικας, στηρίζεται στην τυχαιότητα σε κάποιο ουσιώδη βαθμό. Συνήθως βγάζει αποτελέσματα αυτής της μορφής:





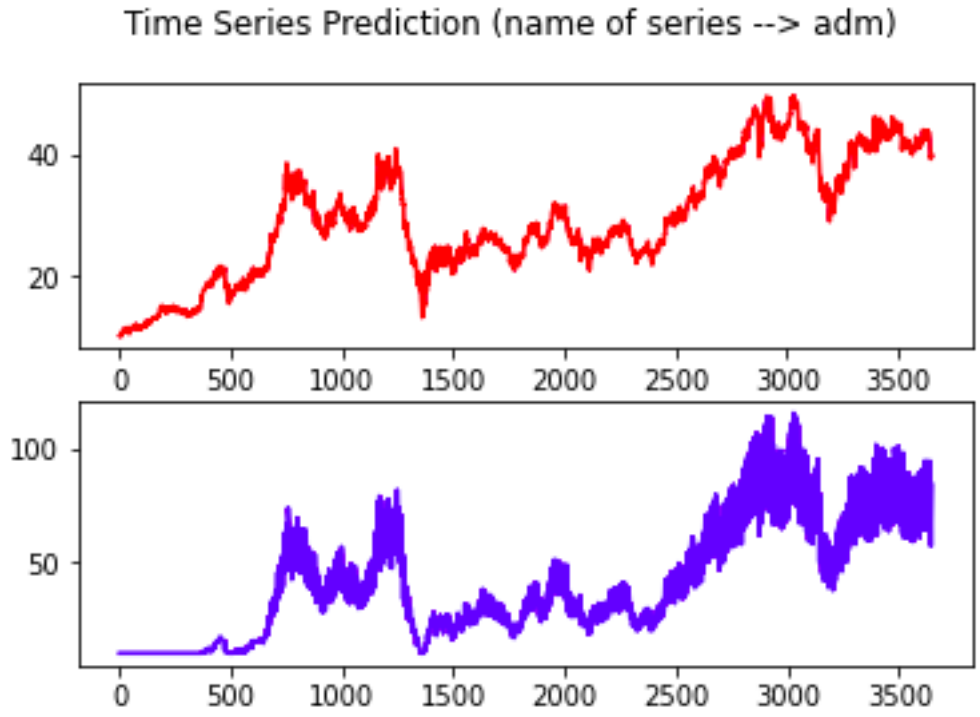
Παρατηρείται ότι προσεγγίζουν υπό μια έννοια το σχήμα της αρχικής καμπύλης αλλά τείνει συνεχώς τιμές τις χρονοσειράς να πηγαίνουν κοντά στο 0 με αποτέλεσμα να φαίνεται σαν επιφάνεια παρά σαν καμπύλη σε αυτό το σχήμα. Ωστόσο, υπάρχουν και μερικές εκτελέσεις που καταφέρνουν κακές οπτικές προσεγγίσεις όπως για παράδειγμα:

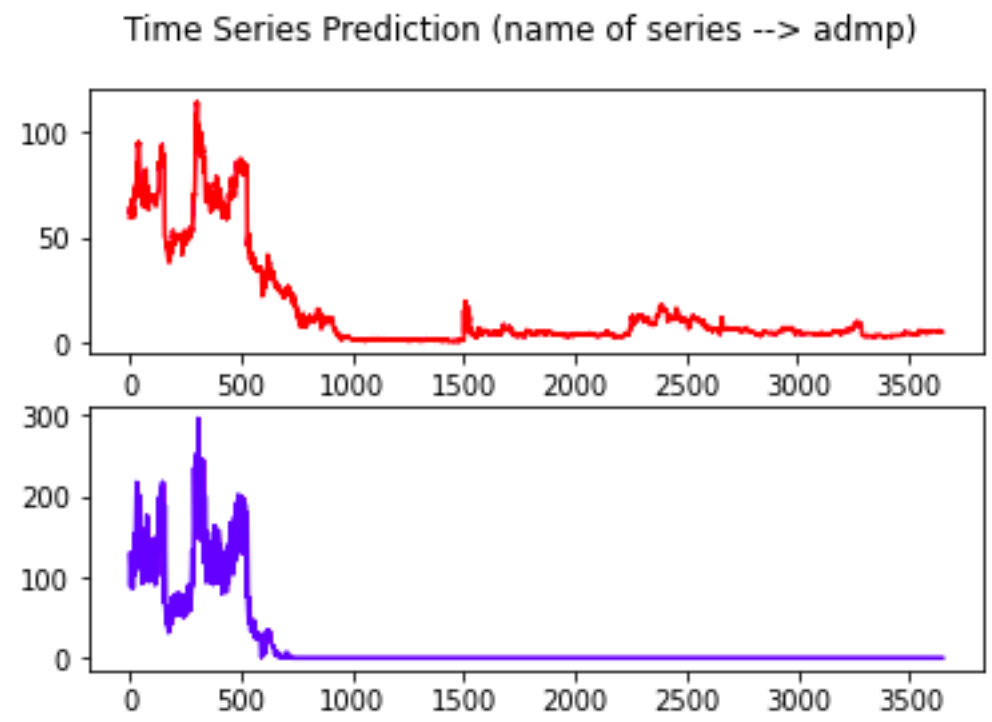


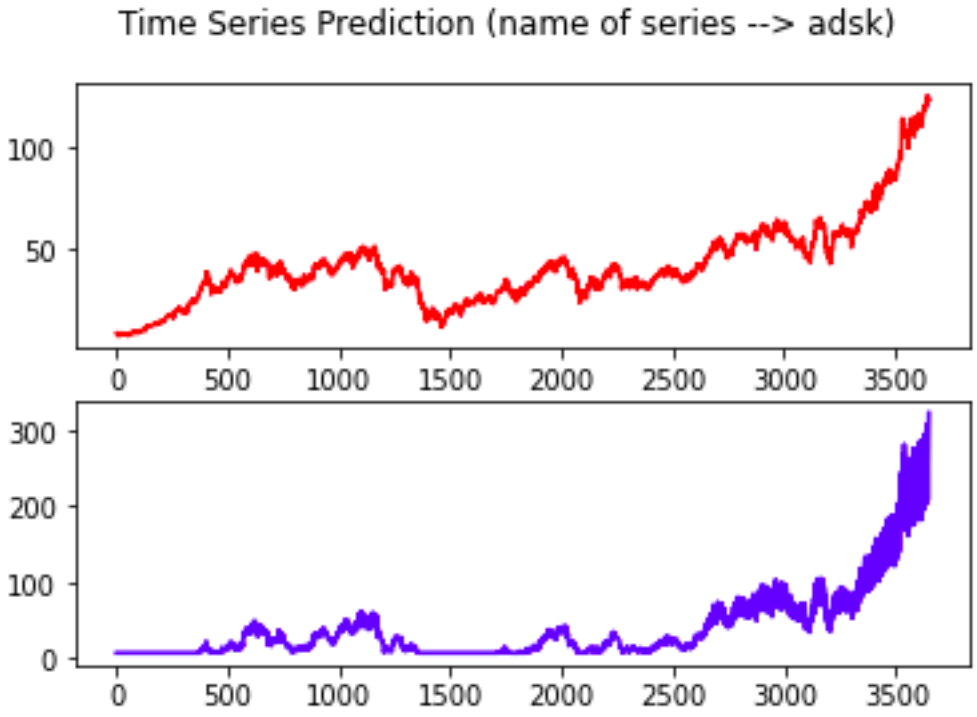


Αυτά τα αποτελέσματα δεν είναι τυχαία, με μια προσεκτική παρατήρηση, οι απότομες αυξήσεις στις encoded χρονοσειρές συμβαίνουν εκεί που η αρχική χρονοσειρά εμφανίζει κάπως πιο απότομες αλλαγές. Αλλά δεν είναι οι προσεγγίσεις που θέλουμε. Αυτή η περίπτωση όμως δεν εμφανίζεται συχνά.

Είναι εμφανές ότι η τυχαιότητα δημιουργεί στο μοντέλο διαφορετικές συμπεριφορές σε διαφορετικές εκτελέσεις για το ίδιο dataset. Συνεπώς αποφασίστηκε να κατασκευαστεί μια συνάρτηση η οποία αφαιρεί την τυχαιότητα από το πρόγραμμα δίνοντας μια παράμετρο seed στα σημεία που υπάρχει τυχαιότητα. Έπειτα από κάποιες δοκιμές με διαφορετικά seeds, βρέθηκε seed το οποίο στο colab έβγαζε πολύ καλές προσεγγίσεις χρονοσειρών με λιγότερα σημεία, συγκεκριμένα:

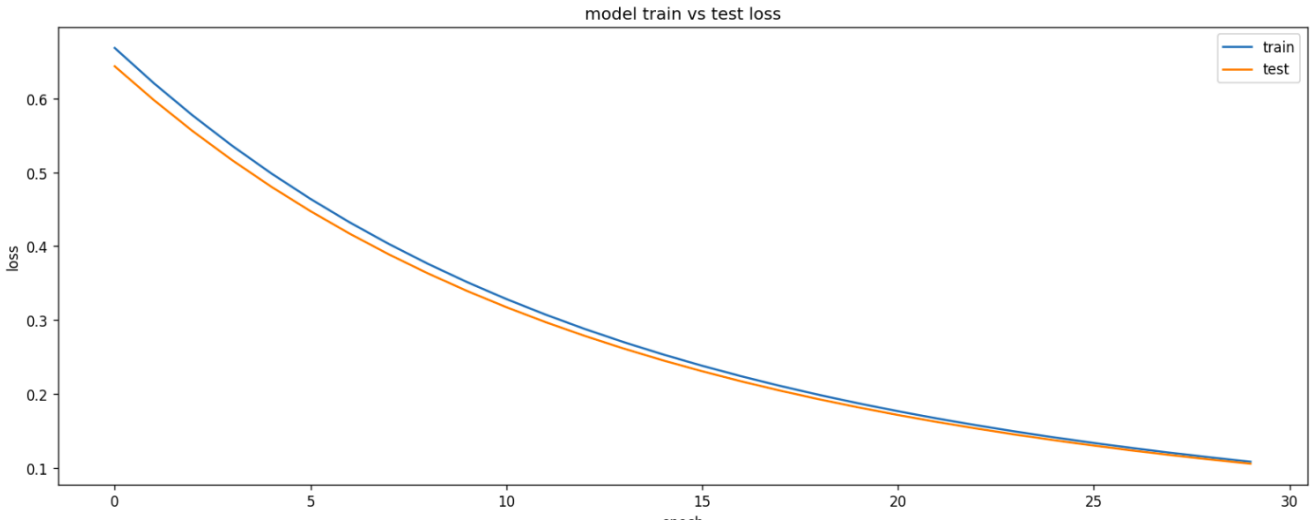






Επιλέχθηκε αυτό το μοντέλο για εμφανείς λόγους (με seed 6969).

Τέλος, το loss είναι:



Με περισσότερες εποχές μπορεί να μειωθεί περαιτέρω το loss αλλά όπως αναφέρθηκε προηγουμένως, δεν βοηθά ιδιαίτερα στην βελτίωση του encoding

4 (Δ)

Η εκτέλεση του clustering για τις encoded χρονοσειρές του καλού μοντέλου που βρήθηκε είχε ως αποτέλεσμα τα clusterings που έβγαιναν για τις αρχικες χρονοσειρές και για τις τελικές να είναι αρκετά παρόμοια, ορισμένες φορές σχεδόν πανομοιότυπα. Και προφανώς, με την μείωση τις διάστασης των χρονοσειρών στα 3/10, η ταχύτητα του clustering αυξήθηκε αρκετά, κάοοιες φορές ήταν 8-10 φορές πιο γρήγορο.

Όσον αφορά την συνάρτηση search, επειδή η καμπύλες ακόμα και οι encoded έχουν πολύ μεγάλη διάσταση, ο continuous frechet καθυστερεί αρκετά. Ανεξαρτήτως αυτού, έχει παρατηρηθεί ότι η τυχαιότητα σε όλες τις παραλλαγές του search μπορεί να αλλάξει τις τιμές των μετρικών στα αποτελέσματα. Ωστόσο, όσο μεγαλύτερα είναι τα δείγματα των datasets, αυτή η επιρροή από την τυχαιότητα μειώνεται αρκετά με αποτέλεσμα οι τιμές των μετρικών των αρχικών και των encoded χρονοσειρών να είναι συχνά αρκετά όμοιες. Προφανώς, με την μειωμένη διάσταση των encoded χρονοσειρών, η ταχύτητα των encoded datasets είναι αρκετά καλύτερη. Ο λόγος όμως του χρόνου της προσεγγιστικής μεθόδου με το χρόνο της ακριβής μεθόδου παραμένει κοινός και στις αρχικές και στις encoded χρονοσειρές όταν τα datasets είναι μεγάλα.

Συμπερασματικά, το επιλεγμένο encoding των χρονοσειρών είναι αρκετά καλό ώστε τα αποτελέσματα των χρονοσειρών στην search και στο clustering να μοιάζουν πολύ, άρα δεν χάνεται πολύ πληροφορία αλλά κερδίζεται αρκετός χρόνος στον υπολογισμό τους.