Ανάπτυξη Λογισμικού για Αλγοριθμικά Προβλήματα

Εργασία 3

Ελευθερία Έλληνα 1115201800228

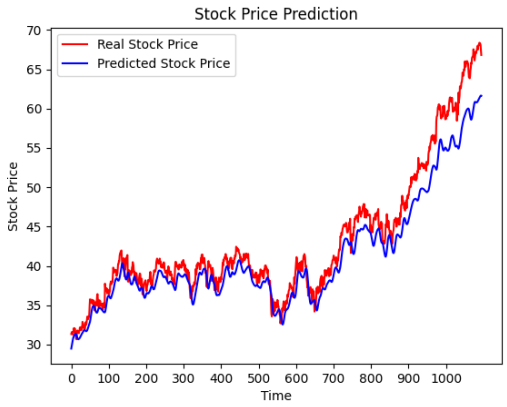
Στυλιανός Ψαρά 1115201800226

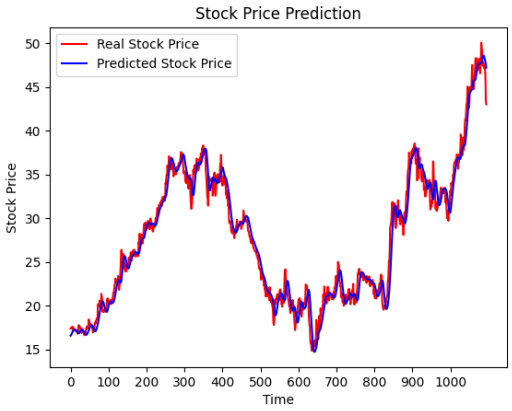
ΕΡΩΤΗΜΑ Α:

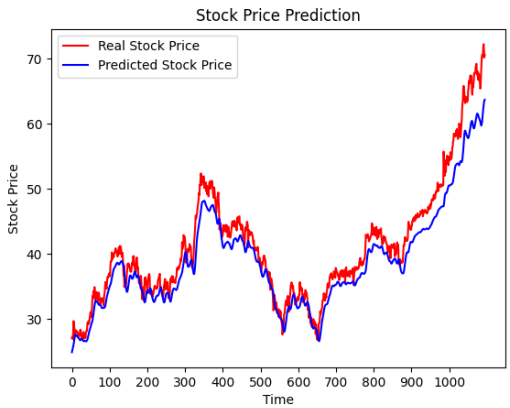
Layers = 4, units = 100, dropout = 0.2, epochs = 10, batch size = 32

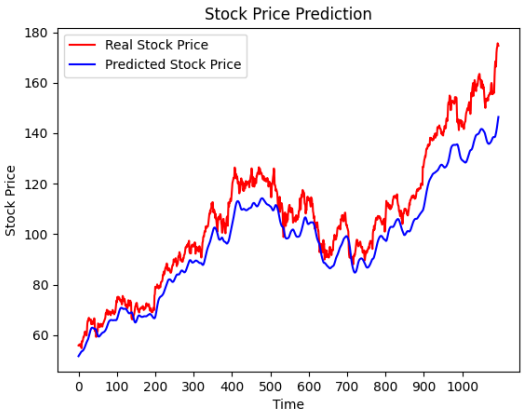
Training set = 70%, Testing set = 30%

Train ανά χρονοσειρά:

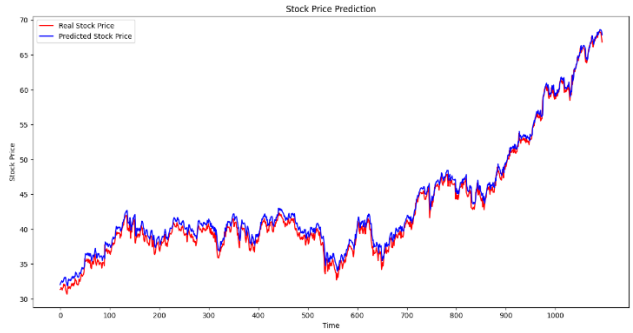


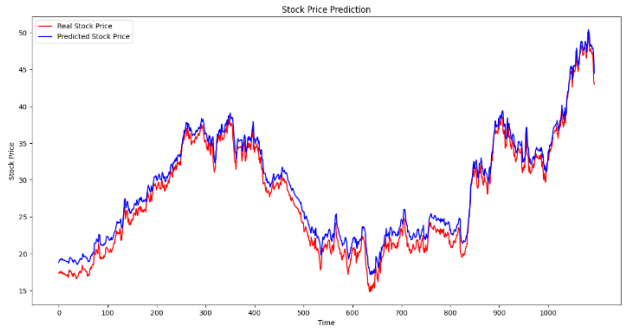


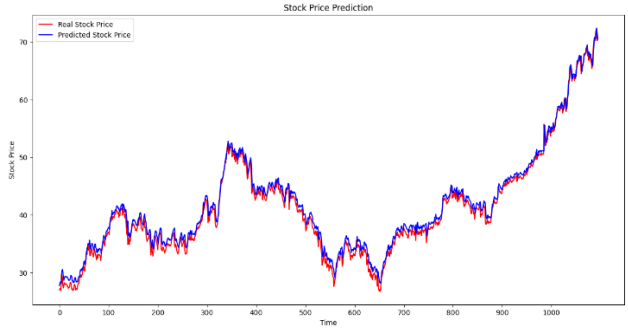


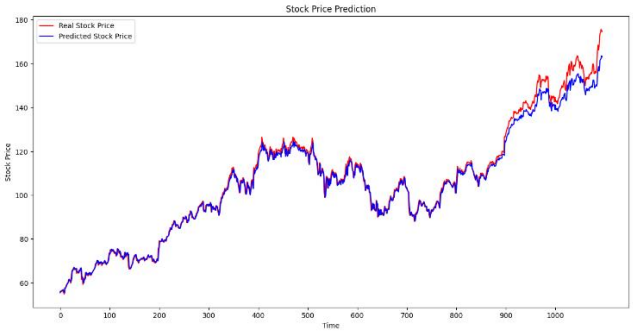


Train συνόλου: (το training γίνεται με 20 χρονοσειρές και το testing με n από αυτές)









Σχολιασμός:

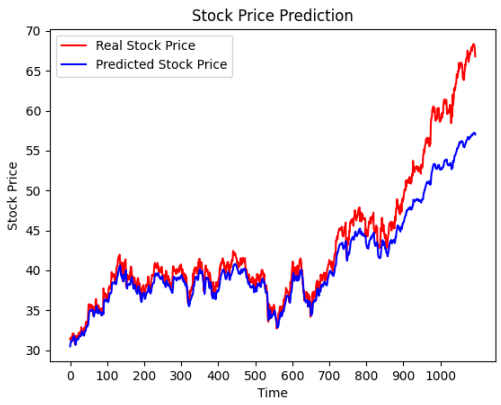
Κάναμε δοκιμές με διάφορες τιμές υπερπαραμέτρων, και διάφορους συνδυασμούς αυτών. Καταλήξαμε με τις πιο πάνω τιμές ως βέλτιστες, τόσο σε αποτέλεσμα γραφικών όσο και σε χρόνο, και για τους δύο τρόπους training.

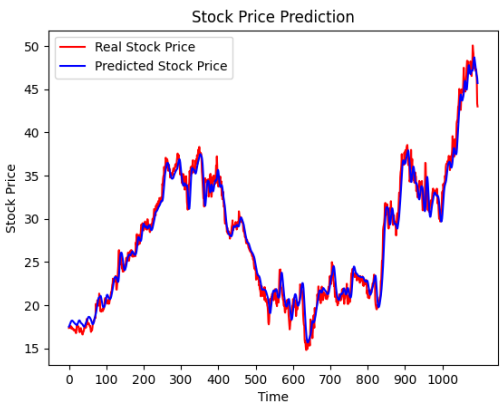
Στην συνέχεια παρατηρήσαμε ότι, με περισσότερες χρονοσειρές για training, τα αποτελέσματα τόσο πιο κοντά σε overfit έβγαιναν για το σύνολο, σε αντίθεση για τις 1 – 1, και γι' αυτό επιλέξαμε 20 έναντι όλων των συμβολοσειρών.

Επίσης, το batch size, όσο μεγαλύτερη τιμή είχε (χρησιμοποιήσαμε δυνάμεις του 2), τόσο γρηγορότερα έτρεχε το fit, αλλά έδινε σαν αποτέλεσμα κακές τιμές, γι' αυτό χρειαζόταν προσαρμογή των υπόλοιπων υπερπαραμέτρων για την βελτίωση των τιμών. Αλλά γενικότερα με οποιονδήποτε συνδυασμό, όταν είχε μικρότερη τιμή είχε καλύτερα αποτελέσματα.

Για τα epoch, παρατηρήσαμε ότι όσο περισσότερα epoch είχαμε, τόσο περισσότερο γινόταν train το μοντέλο, αλλά ήταν και πιο χρονοβόρο. Μάλιστα σε πολύ μεγάλο αριθμό epoch, είχαμε αποτελέσματα κοντά σε overfit.

Στο ακόλουθο παράδειγμα χρησιμοποιήσαμε το ίδιο batch size, και πειραματιστήκαμε με την αναλογία των epoch – units και παρατηρήσαμε ότι καταλήγουμε σε καλά αποτελέσματα στο training 1 - 1, όταν η τιμή των epoch είναι μεγαλύτερη από τα units, σε αντίθεση με την τελική επιλογή τιμών των υπερπαραμέτρων μας. Στο παράδειγμα πιο κάτω έχουμε epoch = 100 και units = 10.





Εμείς καταλήξαμε στην απόφαση να πάρουμε λίγα epoch και πολλά units, με μικρό dropout, ώστε να πετύχουμε τα επιθυμητά αποτελέσματα, σε λογικό χρόνο.

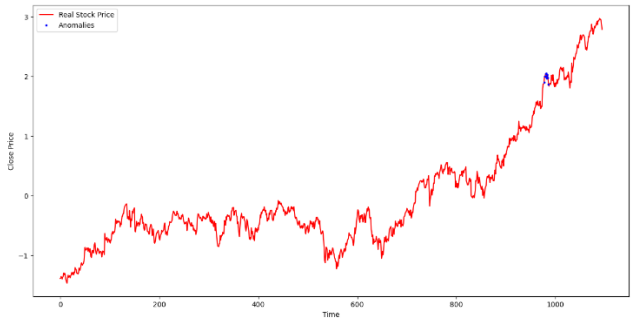
ΕΡΩΤΗΜΑ Β:

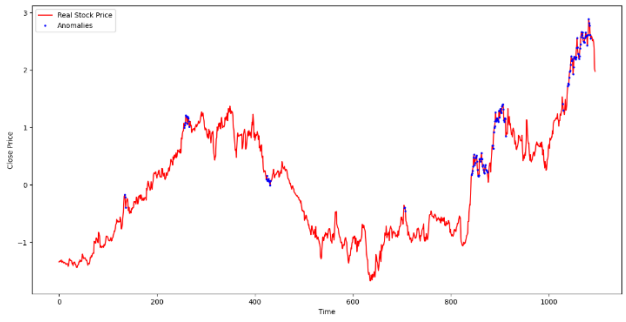
Layers = 4, units = 100, dropout = 0.2, epochs = 10, batch size = 32

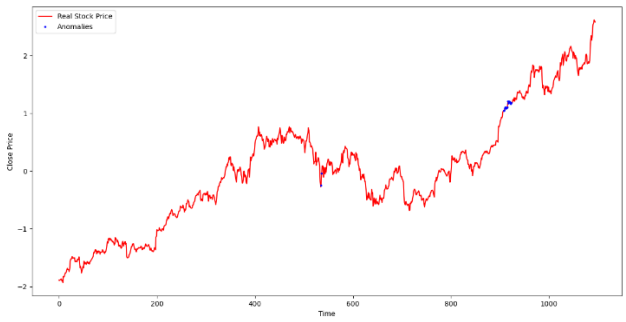
Training set = 70%, Testing set = 30%

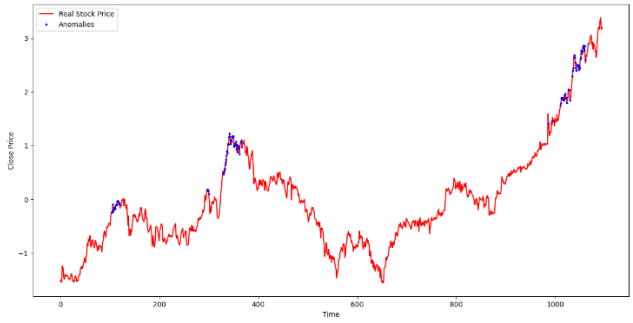
Threashold = 0.65

Train συνόλου: (το training γίνεται με 20 χρονοσειρές και το testing με n από αυτές)









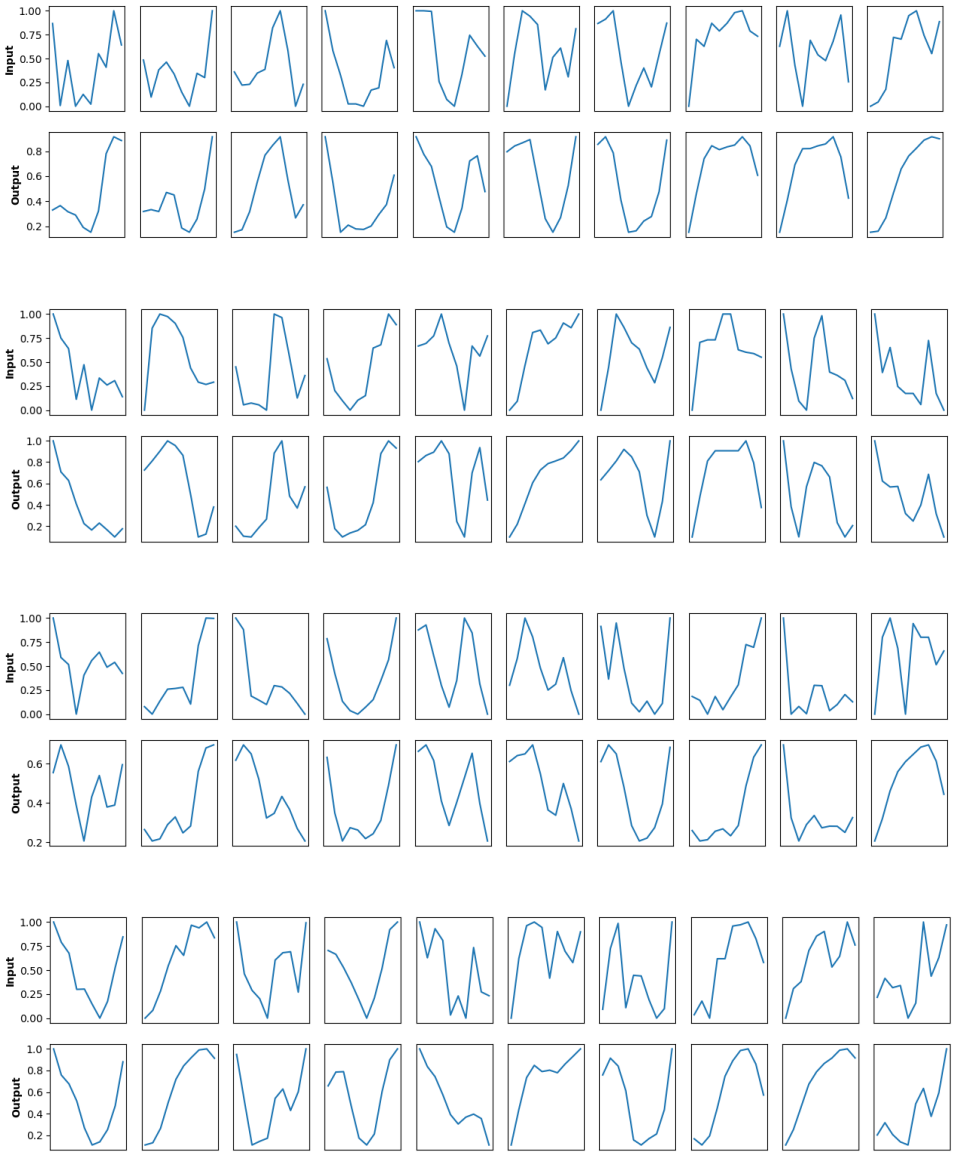
Μετά από πολλές δοκιμές τιμών υπερπαραμέτρων, οι τιμές που καταλήξαμε ήταν αυτές του μοντέλου στο Α, αφού έδωσαν αναμεσά σε άλλες τιμές, τα καλύτερα αποτελέσματα. Η τιμή του threashold που χρησιμοποιήσαμε για τις γραφικές μας, επιλέχθηκε βάση της γραφικής που απεικονίζεται το train mae loss, σύμφωνα με το tutorial.

Επίσης, παρατηρήσαμε κατά τις διάφορες δοκιμές, πως οι ανωμαλίες κυρίως εμφανίζονται σε σημεία όπου οι γραφικές έχουν απότομη αλλαγή στον άξονα Y.

ΕΡΩΤΗΜΑ Γ:

Encoded Layers = 3, Decoded Layers = 4, filters = 32 , epochs = 500, batch size = 16

Train συνόλου: (το training γίνεται με 360 χρονοσειρές και το testing με 110 από αυτές)



Αρχικά παρατηρήσαμε ότι με την ανάλογη αύξηση των layers του encoder, όσο και του decoder, είχαμε καλύτερα αποτελέσματα, σε συνδυασμό με την μετατροπή της τιμής των filter (σε δυνάμεις του 2). Για να έχουμε το επιθυμητό μέγεθος διάστασης στον encoder, δηλαδή από 10 που είναι το αρχικό window length να γίνεται 3, χρειάστηκε στο τελευταίο layer, το filter να είναι ίσο με 1. Κατά την αλλαγή του συγκεκριμένου filter παρατηρήσαμε λάθος διαμόρφωση των διαστάσεων του encoder, αλλά καλύτερα αποτελέσματα μεταξύ των αρχικών και decoded χρονοσειρών, γραφικά. Γι' αυτό επιλέξαμε την ορθή λειτουργία του encoder μέσω του filter = 1, για να έχουμε σωστό αποτέλεσμα για το νέο input csv αρχείο (3650/10*3=1095). Όπως παρατηρείται στις πιο πάνω γραφικές, τα αποτελέσματα έχουν απώλεια πληροφορίας (ακριβείας), αλλά αποτυπώνεται η κεντρική ιδέα του γραφήματος, όπως συμβαίνει και στις γραφικές του tutorial που μας δώθηκε σαν reference.

ΕΡΩΤΗΜΑ Δ:

Χρησιμοποιήσαμε τα αρχεία με τις αρχικές χρονοσειρές και τις encoded χρονοσειρές, σε αναλογία 100 χρονοσειρών για input και 10 για query, [τα αρχεία περιέχονται στο φάκελο που σας έχει σταλεί με ονόματα input_G_initial.csv, query_G_initial.csv, input_G_encoded.csv, query_G_encoded.csv].

Όσον αφορά την εύρεση πλησιέστερου γείτονα βλέπουμε μεγάλη διαφορά χρόνου εκτέλεσης στις αρχικές χρονοσειρές σε σχέση με τις encoded, πράγμα λογικό, λόγω μείωσης πολυπλοκότητας. Αλλά βλέπουμε τις περισσότερες φορές, ότι ο υπολογισμός του approximate με τον true nearest neighbour, έχει μεγαλύτερη ακρίβεια στις αρχικές παρά στις encoded, δηλαδή υπάρχει μεγαλύτερο ποσοστό επιτυχίας εύρεσης του πραγματικού κοντινότερου γείτονα στις αρχικές χρονοσειρές.

Όσον αφορά το Clustering, βλέπουμε διαφορά χρόνου μεταξύ αρχικών και encoded. Η κατανομή στα clusters και τα silhouettes φαίνεται να έχουν καλά αποτελέσματα, θετικές τιμές, τόσο στις αρχικές όσο και στις encoded.