# ***Εργασία 3***

Ν.Νικολακόπουλος
Α.Τσούκο
Α.Μ.: 1115201800133
Α.Μ.: 1115201800202

---

Σας παραδίδουμε τα αρχεία σε .ipynb δηλαδή σε colab notebook. Τα φορτώνεται στο drive σας και τα τρέχετε με τη σειρά τα κελιά.
Επίσης σας αποστέλουμε τα links για να το τρέξετε στο δικό μας drive χωρίς να χρειαστεί να τα κατεβάσετε(είναι readonly νομίζω). Οι παράμετροι που ζητώνται βρίσκονται στο πρώτο κελί και μπορείτε να τις αλλάξετε.
Επίσης σας παρέχεται η δυνατότητα στα Α, Β να διαλέξετε αν θέλετε να κάνετε train ανά μετοχή ή ανά σύνολο μετοχών αλλάζοντας το flag_input σε 'i' ή 'n' αντίστοιχα. Στη συνέχεια δίνεται το id ή τον αριθμό n.
Υπάρχει η δυνατότητα να γίνει save το μοντέλο αλλά είναι αποθηκευμένο στο δικό μας drive. Για τα Α,Β η εκπαίδευση ενός συνόλου μετοχών με n=20 παίρνει περίπου 3 λεπτά.

Το Α περιγράφεται πιο κάτω.

**B.** Παρατηρούμε ότι ο μέσος αριθμός από epochs που χρειάζονται ώστε να γίνει training χωρίς overfitting είναι περίπου 3-5, διότι το training loss παραμένει σε χαμηλά επίπεδα και το validation loss παρόλο που ξεκινά χαμηλά έχει τάσεις να ανεβεί διότι γίνεται overfitting. Ένα παράδειγμα με n=20 μετοχές είναι:

```
Epoch 1/10
46/46 [==============================] - 29s 525ms/step - loss: 0.0047 - mean_absolute_error: 0.0507 - val_loss: 0.0036 - val_mean_absolute_error: 0.0359
Epoch 2/10
46/46 [==============================] - 23s 504ms/step - loss: 0.0013 - mean_absolute_error: 0.0248 - val_loss: 0.0034 - val_mean_absolute_error: 0.0377
Epoch 3/10
46/46 [==============================] - 23s 504ms/step - loss: 8.9972e-04 - mean_absolute_error: 0.0200 - val_loss: 0.0088 - val_mean_absolute_error: 0.0461
Epoch 4/10
46/46 [==============================] - 23s 507ms/step - loss: 7.2647e-04 - mean_absolute_error: 0.0179 - val_loss: 0.0130 - val_mean_absolute_error: 0.0542
Epoch 5/10
46/46 [==============================] - 23s 505ms/step - loss: 6.2221e-04 - mean_absolute_error: 0.0165 - val_loss: 0.0108 - val_mean_absolute_error: 0.0573
```

Το batch size πρέπει να είναι ανάλογο με το μέγεθος του input που δίνουμε για να κάνουμε εκπαίδευση, ώστε να τελειώσει σε εύλογο χρονικό διάστημα και συγχρόνως να έχει καλά αποτελέσματα. πχ στο προηγούμενο παράδειγμα με τις 20 μετοχές και n_timestamps=50 , βάζουμε batch_size=1024, διότι το μέγεθος

του input είναι (58350,50,1) , το οποίο είναι αρκετά μεγάλο, ενώ αν κάναμε train ανά μετοχή θα χρειαζόταν να βάλουμε αρκετά μικρότερο batch_size πχ 64.

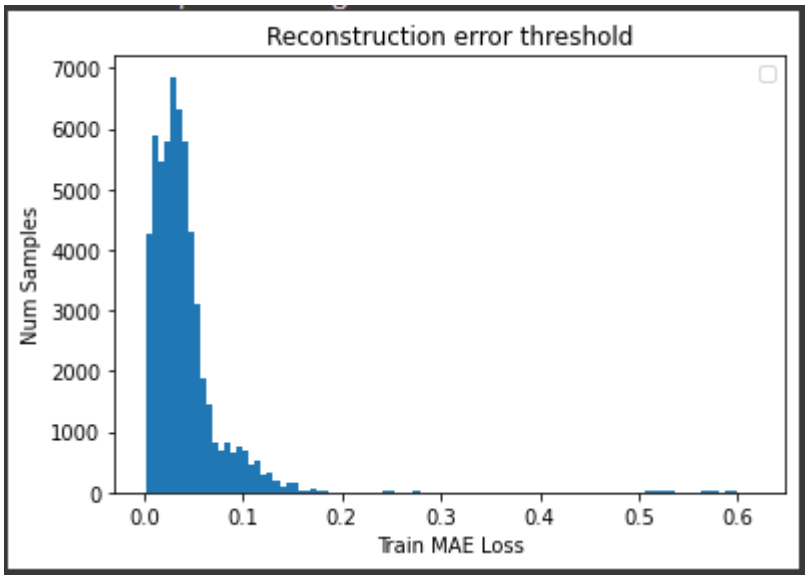
Τα dropout στρώματα εφαρμόζονται σε κάθε επίπεδο μετά από κάθε lstm layer. Η κύρια λειτουργία τους είναι η αποφυγή του Overfit. Το frequency rate που τους δίνουμε σαν είσοδο επιφέρει σημαντικές αλλαγές στην εκπαίδευση του μοντέλου μας αφού μπορούμε να κρατήσουμε χαμηλά τα training και validation loss. Παρατηρήσαμε ότι για dropout=0.5 τα αποτελέσματα είναι αρκετά ικανοποιητικά. Κάναμε και άλλους πειραματισμούς πχ βάλαμε 0.2 ή δεν χρησιμοποιήσαμε καθόλου και ήταν προφανές ότι υπήρχε overfitting και τα αποτελέσματα για "άγνωστες" από το μοντέλο χρονοσειρές δεν ήταν καθόλου καλά (μεγάλο validation loss).

```
Epoch 1/10
46/46 [==============================] - 29s 525ms/step - loss: 0.0047 - mean_absolute_error: 0.0507 - val_loss: 0.0036 - val_mean_absolute_error: 0.0359
Epoch 2/10
46/46 [==============================] - 23s 504ms/step - loss: 0.0013 - mean_absolute_error: 0.0248 - val_loss: 0.0034 - val_mean_absolute_error: 0.0377
Epoch 3/10
46/46 [==============================] - 23s 504ms/step - loss: 8.9972e-04 - mean_absolute_error: 0.0200 - val_loss: 0.0088 - val_mean_absolute_error: 0.0461
Epoch 4/10
46/46 [==============================] - 23s 507ms/step - loss: 7.2647e-04 - mean_absolute_error: 0.0179 - val_loss: 0.0130 - val_mean_absolute_error: 0.0542
Epoch 5/10
46/46 [==============================] - 23s 505ms/step - loss: 6.2221e-04 - mean_absolute_error: 0.0165 - val_loss: 0.0108 - val_mean_absolute_error: 0.0573
```

Για το training χρησιμοποιούμε το 80% των χρονοσειρών και το υπόλοιπο 20% για να κάνουμε προβλέψεις. Αν θέλουμε να κάνουμε εκπαίδευση για πολλές χρονοσειρές τότε κάνουμε concat το 80% και το υπόλοιπο 20% το κρατάμε ως έχει, ξεχωριστά, για να κάνουμε πρόβλεψη ανά μετοχή.

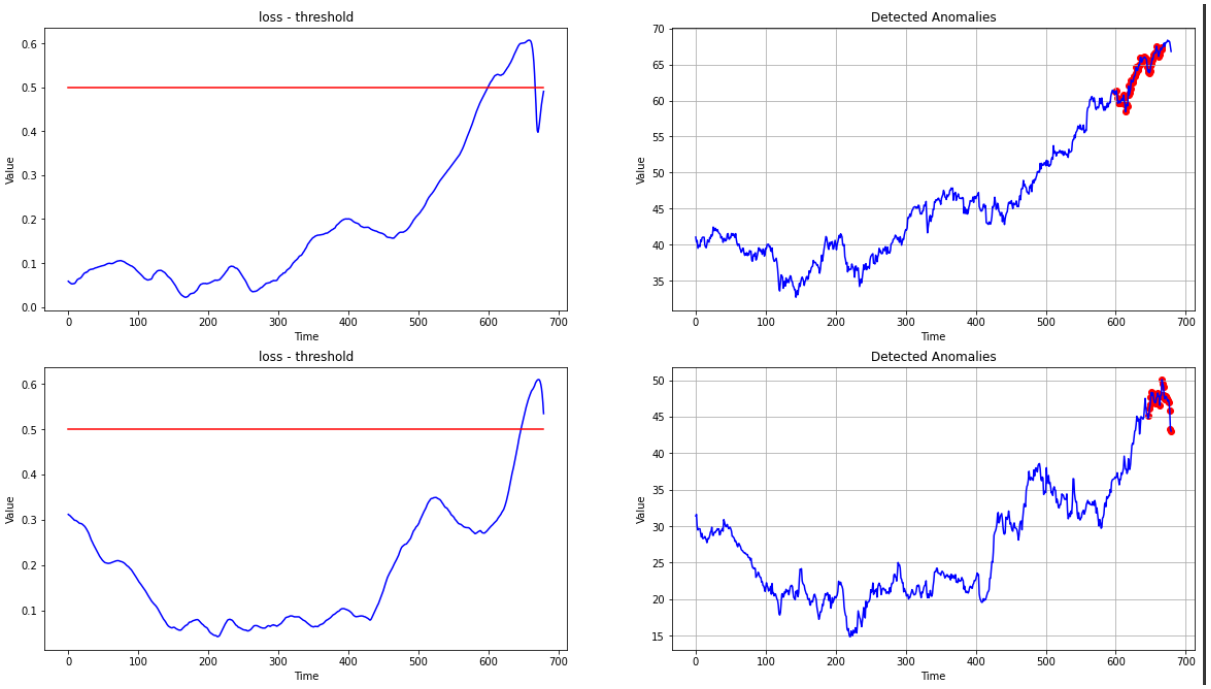
Για τα μοντέλα χρησιμοποιήσαμε 2 ή 3 στρώματα για το encoding και τον ίδιο αριθμό για το decoding. Για παραπάνω αριθμό στρωμάτων παρατηρούνται σημαντικές αλλοιώσεις στα αποτελέσματα.

Μια καλή τιμή για το κατώφλι όταν κάνουμε εκπαίδευση πολλών μετοχών είναι 0.5.

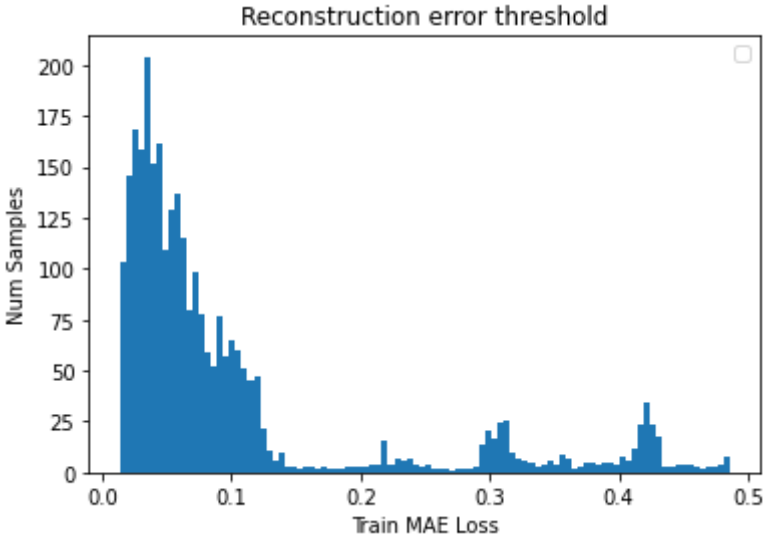


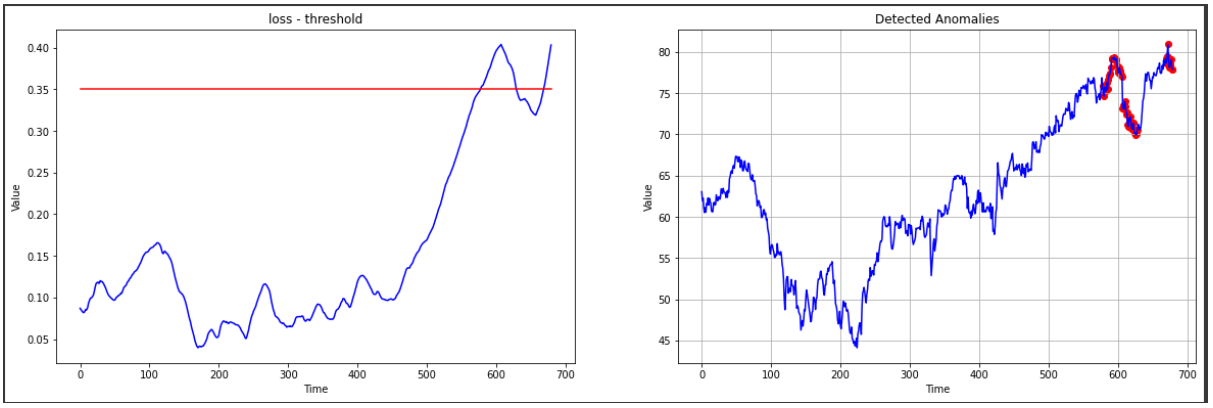
Θέλουμε να επιλέξουμε μια τιμή, λίγο μικρότερη από την μέγιστη. Το training έγινε για n=20 χρονοσειρές.

Επίσης, τα αποτελέσματα για τα anomalies είναι αυτά που περιμέναμε για το συγκεκριμένο κατώφλι και μοντέλο. Παρακάτω παραθέτονται διαγράμματα αναγνώρισης ανωμαλιών ανά χρονοσειρά(είναι το test_set, δηλαδή το 20% της χρονοσειράς που χωρίσαμε στην αρχή για το training και για το test).

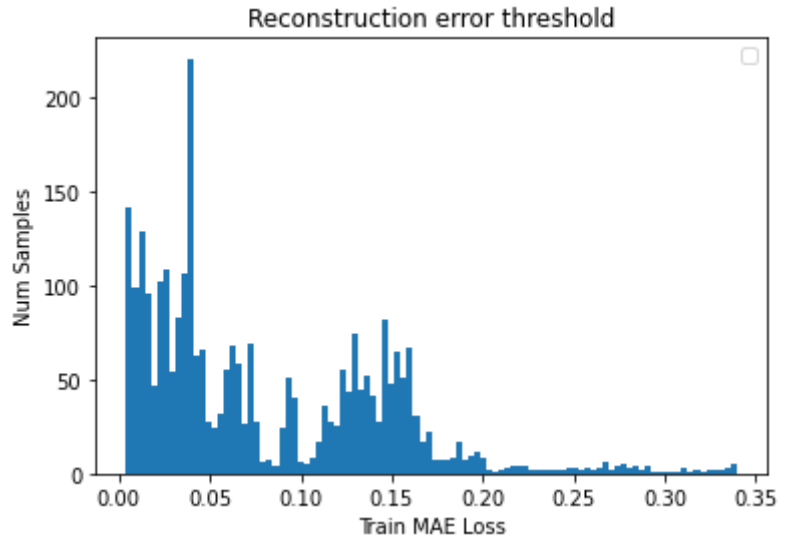


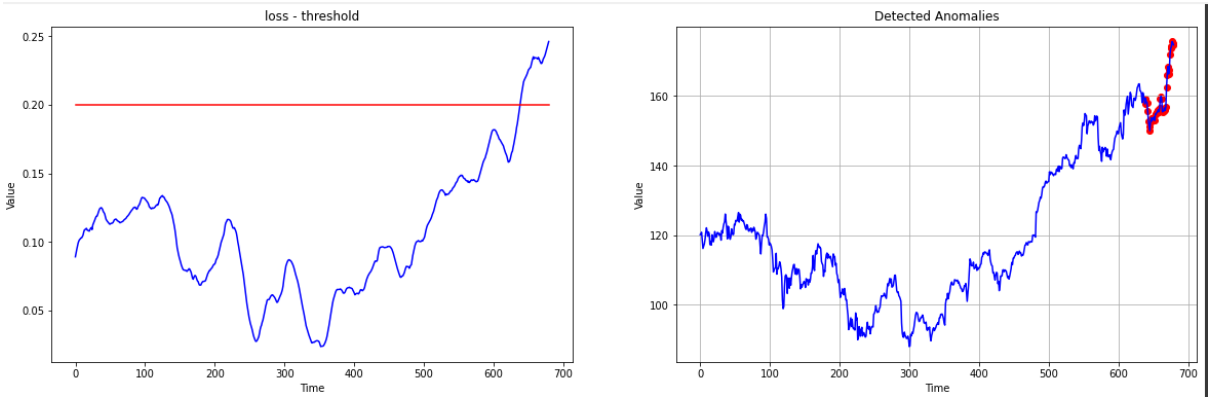

Για κατώφλι ανά μετοχή πρέπει να επιλέξουμε ανάλογη τιμή με την καθεμιά. Πχ στην 'etn' επιλέξαμε το 0.3 το οποίο έχει επίσης αναμενόμενα αποτελέσματα:






Για την ‘aapl’ επιλέξαμε 0.2 για να έχει αντίστοιχα αποτελέσματα.



**Α.** Ισχύουν τα ίδια για το Α, όπως στο Β αφού χρησιμοποιήθηκε το ίδιο μοντέλο. Παρατηρούμε ότι ο μέσος αριθμός από epochs που χρειάζονται ώστε να γίνει training χωρίς overfitting είναι περίπου 8. Το batch_size για πολλές μετοχές είναι ίσο με 1024, ενώ για train ανά μετοχή είναι ίσο με 64. Επιπλέον, χρησιμοποιούμε τρία LSTM επίπεδα με dropout ίσο με 0.3. Τέλος, για το training χρησιμοποιούμε το 80% των χρονοσειρών και το υπόλοιπο 20% για να κάνουμε προβλέψεις. Αν θέλουμε να κάνουμε εκπαίδευση για πολλές χρονοσειρές, τότε κάνουμε concat το 80% και το υπόλοιπο 20% το κρατάμε ως έχει, ξεχωριστά, για να κάνουμε πρόβλεψη ανά μετοχή.
Ο αριθμός των timestamps που παρατηρήσαμε ικανοποιητικά αποτελέσματα ήταν για n_timestamps=50 , αφού κατά αυτόν τον τρόπο το μοντέλο μας έχει την δυνατότητα να "κοιτάει αρκετά πίσω" και να μπορεί να προβλέψει με μεγάλη επιτυχία την προβλεπόμενη τιμή. Δοκιμάστηκαν και μικρότερο look back, πχ 10 αλλά δεν είχε τόσο καλά αποτελέσματα. Βέβαια, μεγαλύτερος αριθμός σε Lookback σημαίνει περισσότερο χώρο και χρόνο αλλά αυτά τα αρνητικά δεν ήταν σε μεγάλο βαθμό για n_timestamps=50.

**Γ.** Στο Γ ισχύουν τα παραπάνω για τις παραμέτρους που χρησιμοποιήθηκαν στα προηγούμενα ερωτήματα. Ωστόσο, αυτό που αλλάζει είναι ότι πλέον ότι ο αριθμός των διαστημάτων που χωρίζεται (windows) είναι πολύ μικρότερος αφού πλέον θέλουμε να μειώσουμε την διάσταση των χρονοσειρών (αυτό γίνεται με τον τρόπο που αναφέρθηκε στις συζητήσεις στο eclass) που πλέον τα διαστήματα που παράγονται δεν δημιουργούνται με απλό shift κατά 1, αλλά έχει την μορφή διαίρεσης το window size. Το split αλλάζει από
x1,x2,..,x10 -> predict x11 **,** x2,x3,..,x11 -> predict x12 , … ΑΛΛΆΖΕΙ ΣΕ:
x1,x2,..,x10 -> predict x11 **,** x11,x12,..,x20 -> predict x21 , …
Έτσι έχουμε μικρότερο αριθμό από σημεία στο τελικό timeseries.

Παρατηρούμε ότι με 2 layers παράγονται αποτελέσματα πολύ κοντά στα πραγματικά. Ο αριθμός των filters που επιλέγουμε είναι 16 και μειώνεται στα μισά στο επόμενο layer. Το latent dim είναι 3 ο οποίος είναι καλός αριθμός αφού ρίχνει την πολυπλοκότητα του window size κατά πολύ και η ακρίβεια παραμένει στα αποτελέσματα. Δοκιμάσαμε και μεγαλύτερες διαστάσεις αλλά είχαν περισσότερα σημεία ή ήθελαν περισσότερο χρόνο για να παραχθούν.

Τα αποτελέσματα είναι πολύ κοντά στα πραγματικά και έχουν πολύ μεγάλη ακρίβεια.

**Δ.** Με τα αρχεία εισόδου και αναζήτησης από το ερώτημα Γ, λαμβάνουμε τα αποτελέσματα γρηγορότερα, καθώς έχει μειωθεί η πολυπλοκότητα. Για το λόγο αυτό, όμως, υπάρχει και μεγαλύτερη ανακρίβεια.