Ανάπτυξη Λογισμικού για Αλγοριθμικά Προβλήματα

3η Προγραμματιστική Εργασία

Πρόβλεψη τιμών μετοχών, ανίχνευση ανωμαλιών με τη χρήση **LSTM** νευρωνικών δικτύων στην **Python (Keras, Tensorflow)**

Απόστολος Θεοδώρου (Α.Μ.: 1115201500046)

Χειμερινό Εξάμηνο 2021-2022

Τμήμα Πληροφορικής και Τηλεπικοινωνιών
Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών

Contents

1 Περιγραφή της εργασίας

Η παρούσας εργασία αποσκοπεί στην κατασκευή και εκπαίδευση νευρωνικών δικτύων με απώτερο στόχο την πρόβλεψη μελλοντικών τιμών μετοχών, την ανίχνευση ανωμαλιών σε χρονοσειρές και την μείωση της πολυπλοκότητας χρονοσειρών. Χρησιμοποιείται η γλώσσα προγραμματισμού **Python** και συγκεκριμένα η διεπαφή **Keras**.

Δείτε αναλυτικά την εκφώνηση της εργασίας

2 Αρχεία

- **forecast_build_oneByOne.py**: κατασκευάζει και εκπαιδεύει νευρωνικά δίκτυα **LSTM** πλήθους **n** ,ένα για κάθε επιλεγμένη μετοχή. Στη συνέχεια, λαμβάνοντας ως σύνολο εκπαίδευσης το αρχικό 80% των τιμών της μετοχής, προβλέπει το τελικό 20% και αντιπαραβάλει σε κοινή γραφική παράσταση στις πραγματικές τιμές της μετοχής τις προβλέψεις.

- **forecast_build.py**: υλοποιεί και εκπαιδεύει νευρωνικό δίκτυο **LSTM** χρησιμοποιώντας το σύνολο των χρονοσειρών που έλαβε. Στη συνέχεια χρησιμοποιεί το μοντέλο για να κάνει προβλέψεις επί των χρονοσειρών και προβάλει πραγματικές και προβλεπόμενες τιμές (το τελικό 20%) σε κοινή γραφική παράσταση για κάθε χρονοσειρά.

- **forecast.py**: φορτώνει ένα ήση εκπαιδευμένο μοντέλο και το χρησιμοποιεί για την πρόβλεψη μελλοντικών τιμών μετοχών. Το μοντέλο που φορτώνεται είναι αυτό που απέδωσε τις καλύτερες προβλέψεις.

- **detect_build.py**: κατασκευάζει και εκπαιδεύει νευρωνικό δίκτυο ώστε να ανιχνεύει ανωμαλίες χρονοσειρών. Στη συνέχεια κάνει χρήση του μοντέλου με τα αποτελέσματα να παρουσιάζονται σε γραφικές παραστάσεις.

- **detect.py**: ανιχνεύει ανωμαλίες σε χρονοσειρές χρησιμοποιώντας ένα εκπαιδευμένο μοντέλο. Το χρησιμοποιούμενο μοντέλο είναι αυτό με τις βέλ-

τιστες παραμέτρους.

3 Εκτέλεση

Τα προγράμματα εκτελούνται με τις ακόλουθες εντολές:

- python3 forecast.py –d <dataset>-n <number of time series selected>
- python3 detect.py –d <dataset>-n <number of time series selected>-mae <error>

Απαραίτητη προϋπόθεση για την εκτέλεση των προγραμμάτων είναι η εγκατάσταση της γλώσσας Python (version 3.7 ή νεότερης) και των εξής βιβλιοθηκών:

- numpy
- scipy
- scikit-learn
- ipython
- matplotlib
- tensorflow
- pandas
- keras
- seaborn

4 Πειραματική Μελέτη

4.1 Α' Ερώτημα

4.1.1 Πίνακας εκτελέσεων

ΕΚΠΑΙΔΕΥΣΗ ΑΝΑ ΣΥΝΟΛΟ

learningra	*layers*	*layerssize*	*epoch*	*batchsize*	*loss*
0.01	8	32	5	32	19568
0.01	8	32	20	32	149583
0.01	4	32	5	32	0.009
0.01	4	32	20	32	0.009
0.0001	4	32	5	32	0.009
0.0001	4	32	5	5	0.008
0.0001	4	64	5	32	0.005

4.1.2 Σχόλια - παρατηρήσεις για τα αποτελέσματα

Είναι φανερό από τις πειραματικές εκτελέσεις ότι η αύξηση του αριθμού των στρωμάτων (από 4 σε 8) στο συγκεκριμένο πρόβλημα οδηγεί σε κατά πολύ μεγαλύτερο σφάλμα. Αντίθετα, τα 4 στρώματα παράγουν ένα ικανοποιητικό σφάλμα μικρότερο του 0.01 για οποιαδήποτε επιλογή υπερπαραμέτρων. Επομένως ως αριθμός στρωμάτων επιλέγεται το 4.

Διατηρώντας αμετάβλητο τον αριθμό των στρωμάτων σε 4 και κάνοντας δοκιμές με τις άλλες υπερπαραμέτρους παρατήρησα, προς έκπληξή μου, ότι η αύξηση των εποχών εκπαίδευσης δεν οδηγεί σε μείωση (αλλά ούτε και αύξηση) του σφάλματος. Έτσι επέλεξα να κρατήσω τον μικρότερο αριθμό εποχών, τις 5, για να αποφεληθώ από τους μικρότερους χρόνους εκπαίδευσης και να κάνω

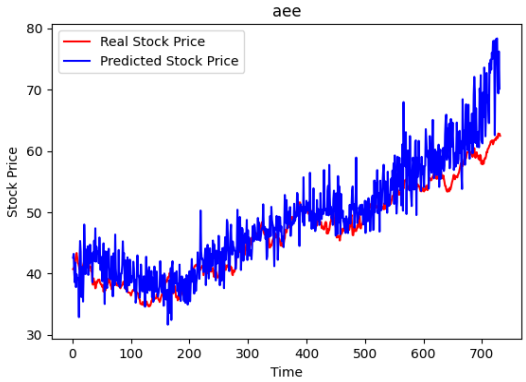
περισσότερες δοκιμές.

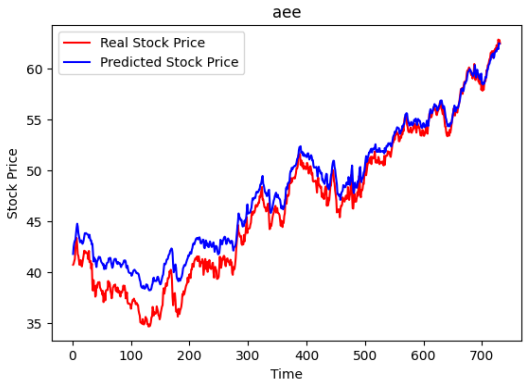
Επόμενος παράγοντας πειραματισμού αποτέλεσε ο βαθμός εκμάθησης. Η αρχική γενναία μείωσή του (από 0.01 σε 0.0001) δεν απέφερε κάποια βελτίωση, ούτε κι αύξηση του σφάλματος. Αποφάσισα να κρατήσω τον μικρότερο βαθμό για τις επόμενες δοκιμές, πιστεύοντας ότι λόγω της μεγαλύτερης ακρίβειας που παρέχει είναι πιο πιθανό να οδηγήσει σε κάποιο καλύτερο αποτέλεσμα.

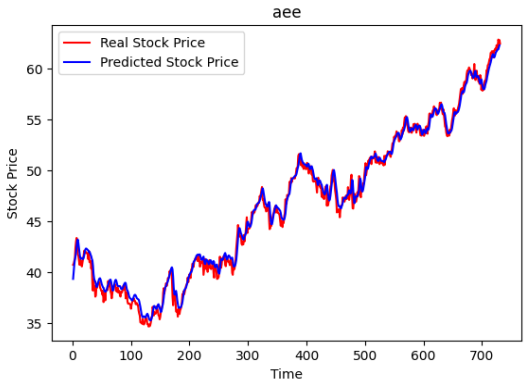
Στη συνέχεια εξέτασα το batch size, του οποίου η μείωση συνοδεύτηκε από μία μικρή μείωση του σφάλματος κατά 0.001, αλλά και από σημαντική αύξηση του χρόνου εκπαίδευσης. Στον συμβιβασμό μεταξύ του χρόνου εκπαίδευσης και του ελάχιστα βελτιωμένου σφάλματος, επέλεξα τον μικρότερο χρόνο εκπαίδευσης, ελπίζοντας ότι η μεταβολή της επόμενης παραμέτρου θα έδινε μια πιο ουσιαστική βελτίωση σε καλύτερο χρόνο.

Πράγματι, η τελευταία παράμετρος, το μέγεθος των στρώσεων, συντέλεσε με τον διπλασιασμό της στον υποδιπλασιασμό σχεδόν του σφάλματος, από 0.009 σε 0.005 . Ωστόσο η επιλογή ακόμα μεγαλύτερου μεγέθους στρώσεων δεν εξακολούθησε να μειώνει το σφάλμα, αλλά να το αυξάνει αισθητά. Συνεπώς ως καλύτερο μέγεθος κάθε στρώσης κατέληξα στο 64.

Συμπερασματικά, οι βέλτιστες παράμετροι για το μοντέλο του 1ου ερωτήματος είναι αυτές που εμφανίζονται στην τελευταία σειρά του παραπάνω πίνακα. Αξίζει να σημειωθεί ότι εκτός από μικρότερο σφάλμα, το βέλτιστα παραμετροποιημένο μοντέλο αποδίδει και καλύτερες γραφικές παραστάσεις, με πιο ομαλές καμπύλες (αποφυγή overfitting) που είναι, στη συντριπτική του πλειοψηφία πολύ κοντά στις πραγματικές τιμές των μετοχών. Αυτό μπορεί να γίνει άμεσα αντιληπτό με κάποια παραδείγματα:



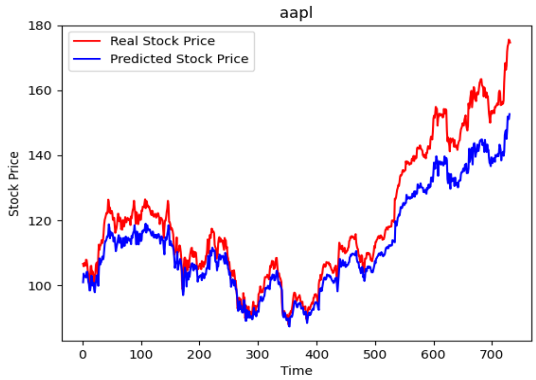


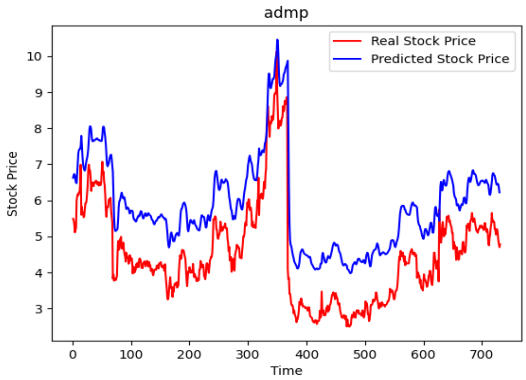


Από πάνω προς τα κάτω: η μετοχή aee με 4 στρώματα μεγέθους 32, με μέγεθος δέσμης 32, με epochs 20, 5 και 5 αντίστοιχα και learning rate 0.01 , 0.01 και 0.0001 αντίστοιχα. Παρατηρήστε πόσο οξεία γίνεται η καμπύλη στην 1η εικόνα με τις 20 εποχές εκπαίδευσης (σημάδι πιθανού overfitting). Στη 2η

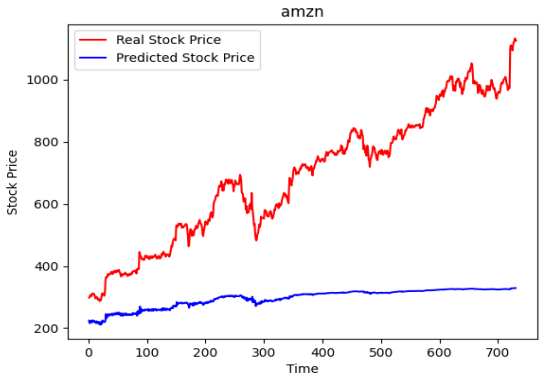
εικόνα με 5 εποχές εκπαίδευσης παρουσιάζεται μια πιο ομαλή καμπύλη, ενώ στην 3η εικόνα με 5 εποχές εκπαίδευσης και μικρότερο learning rate η καμπύλη των προβλέψεων προσεγγίζει ακόμα καλύτερα αυτή των πραγματικών τιμών.

Στο βέλτιστο μοντέλο της forecast.py η πλειοψηφία των καμπυλών έχει πολύ καλή συμπεριφορά, ανάλογη της 3ης εικόνας. Ωστόσο υπάρχουν και ορισμένες καμπύλες οι οποίες δεν προσεγγίζουν το ίδιο καλά τις πραγματικές τιμές, αλλά δεν τα πάνε κι άσχημα. Δείτε παρακάτω και θα καταλάβετε:

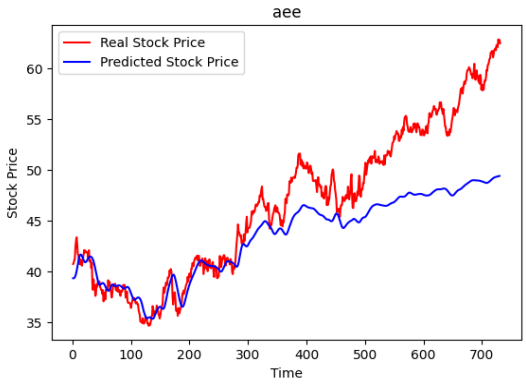




Επιπλέον, υπάρχουν λίγες στον αριθμό καμπύλες οι οποίες δεν κατέστη δυνατό να προσεγγιστούν καλά από οποιοδήποτε μοντέλο δοκίμασα, ακόμα και για μοντέλα με μικρό σφάλμα. Μία χαρακτηριστική τέτοια καμπύλη είναι αυτή της μετοχής amzn:



Τέλος θα ήθελα να τονίσω ότι η εκπαίδευση χρονοσειρών στο σύνολο των χρονοσειρών φαίνεται να είναι πιο αποτελεσματική από την αντίστοιχη ανά χρονοσειρά, γεγονός που πιστεύω είναι λογικό και αποδίδεται στο μεγαλύτερο πλήθος δεδομένων στα οποία εκπαιδεύτηκε το 1ο μοντέλο σε σχέση με το 2ο:



Η εκτίμηση της μετοχής aee όπως προβλέφθηκε από το μοντέλο που εκπαιδεύτηκε μόνο από τη συγκεκριμένη χρονοσειρά

4.2 Β' Ερώτημα

4.2.1 Πίνακας εκτελέσεων

dropout	*learning*	*layerssiz*	*epoch*	*batchsize*	*loss*	*val_loss*
0.2	0.0001	128	5	32	nan	nan
0.2	0.0001	64	5	32	0.7891	0.7360
0.2	0.0001	32	5	32	0.7823	0.7399
0.2	0.0001	32	20	32	0.7816	0.7348
0.2	0.0001	16	5	32	0.7817	0.7341
0.2	0.0001	16	15	32	0.7816	0.7355
0.2	0.0001	16	5	32	0.7853	0.7395
0.4	0.0001	16	5	8	0.7795	0.6490
0.2	0.0001	32	5	8	0.7722	0.6147
0.25	0.0001	32	5	8	0.7721	0.6061

4.2.2 Σχόλια - παρατηρήσεις για τα αποτελέσματα

Ακολουθώντας την ίδια μεθοδολογία όπως και στο 1ο ερώτημα πειραματίστηκα με τις υπερπαραμέτρους και έκανα τις εξής παρατηρήσεις:

- Όπως και στο 1ο ερώτημα, το πολύ μεγάλο μέγεθος στρώσης οδηγεί σε υψηλό σφάλμα. Τελικά κατέληξα στον αριθμό 32, αν και οι διαφορές μεταξύ 16, 32 και 64 είναι πολύ μικρές.
- Σε αντίθεση με το 1ο ερώτημα, η αύξηση του αριθμού των εποχών εκπαίδευσης στο συγκεκριμένο πρόβλημα φαίνεται να βοηθάει λίγο στην ελάττωση του σφάλματος, όμως όχι αρκετά ώστε να αξίζει την χρονική αναμο-

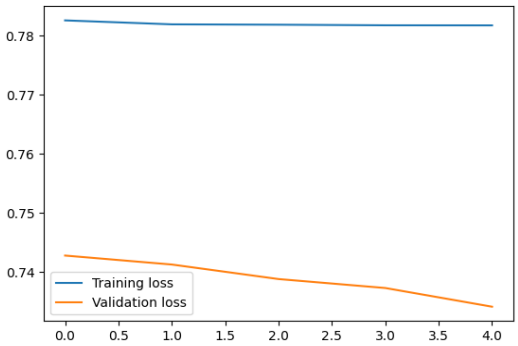
νή.

- Η ελάττωση του αριθμού δέσμης έχει κι αυτή έναν μικρό μεν, θετικό δε αντίκτυπο στο σφάλμα και έναν ακόμη πιο ισχυρό αντίκτυπο στο validation loss. Για αυτό υιοθετήθηκε το 8 ως μέγεθος δέσμης.

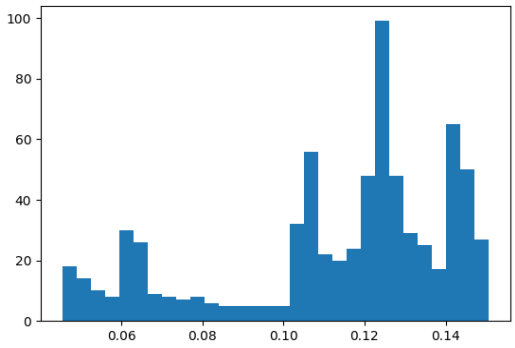
- Τέλος, το ποσοστό του dropout παρατηρήθηκε ότι μπορεί να φέρει μια μικρή βελτίωση αν αυξηθεί λίγο πάνω από το 0.2 .

  Συγκεφαλαιώνοντας, αν και διεξήχθησαν πολλές δοκιμές, δεν κατάφερα να πετύχω εξίσου καλά αποτελέσματα με το 1ο υποερώτημα. Το μικρότερο σφάλμα που παρατήρησα ήταν το 0.7721 για τις τιμές των υπερπαραμέτρων της τελευταίας γραμμής του πίνακα. Άλλες προσπάθειες με αλλαγή άλλων παραμέτρων, όπως το lookback και ο αριθμός των στρωμάτων αποδείχθηκαν κι αυτές άκαρπες.

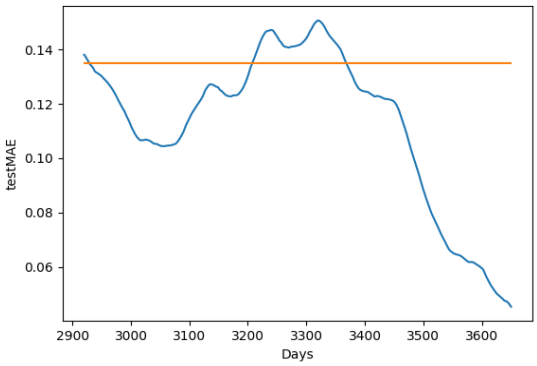
Παρακάτω παραθέτω κάποιες γραφικές παραστάσεις όπου διακρίνονται οι ανωμαλίες με κόκκινα σημεία:



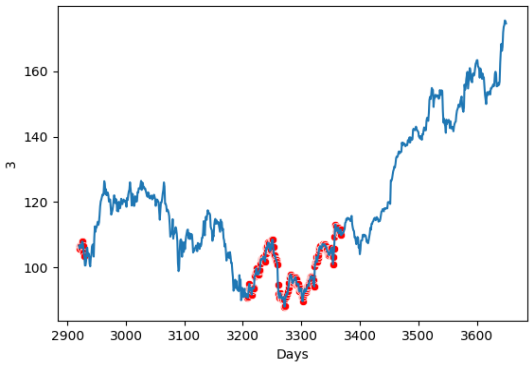
Το σφάλμα παραμένει σταθερό κοντά στο 0.7 παρά την εκπαίδευση. Το validation loss δείχνει να μειώνεται δημιουργώντας ένα gap με το σφάλμα.



Στο ιστόγραμμα για κάποια μετοχή βλέπουμε ότι αρκετές από τις προβλέψεις βρίσκονται προς τη μεριά του μεγαλύτερου σφάλματος που παρατηρήθηκε.



Οι ανωμαλίες παρουσιάζονται στην αρχή και προς τη μέση του χρονικού άξονα. Στη συνέχεια το σφάλμα έχει κατακόρυφη πτώση, οπότε οι προβλέψεις των τελευταίων τιμών θα είναι πιο κοντά στις πραγματικές



Η γραφική παράσταση της μετοχής όπου επισημαίνονται με κόκκινο οι ανωμαλίες