ΜΗΧΑΝΙΚΗ ΜΑΘΗΣΗ

ΕΡΓΑΣΙΑ 2021-2022

**ΚΩΝΣΤΑΝΤΙΝΟΣ ΚΑΤΣΕΡΕΛΗΣ - 3170065**

**ΜΑΡΙΝΑ ΣΑΜΠΡΟΒΑΛΑΚΗ - 3180234**

*(Ο κώδικας, τα αποτελέσματα και τα visualisations βρίσκονται στα notebooks, παρακάτω θα παρατεθούν επεξηγήσεις και η περιγραφή των όσων κάναμε για την οικονομία χώρου)*

**Μέρος 1ο (10 Μονάδες)**

1. Κατεβάσαμε τα δεδομένα από το Yahoo Finance για τα τελευταία 5 έτη.
2. Βλέποντας τα δεδομένα, παρατηρήσαμε ότι η στήλη **Close** καθώς και η στήλη **Adj Close** έχουν τα ίδια δεδομένα και έτσι αποφασίσαμε να αφαιρέσουμε την **Adj Close**. Επίσης, μετατρέψαμε την στήλη **Date** σε *datetime* type για να μπορούμε να χρησιμοποιούμε τις ιδιότητες των ημερομηνιών και αφαιρέσαμε τις NA τιμές.  
   Χρησιμοποιώντας την *describe()* παρατηρήσαμε πως οι διαφορετικές μεταβλητές μας βρίσκονται σε αρκετά διαφορετικές κλίμακες και έτσι θα χρειαστεί σίγουρα κανονικοποίηση.Τέλος, προσθέσαμε στο dataset μας μια μεταβλητή “**Up/Down**” με βάση τα δεδομένα της στήλης **close**. Στην ουσία Μας περιγράφει αν η σημερινή τιμή κλεισίματος είναι μεγαλύτερη απο την χθεσινή (Δηλαδή αν θα είχαμε κέρδος την συγκεκριμένη μέρα σε long θέση). Αυτή η μεταβλητή χρησιμοποιήθηκε για την Logistic Regression.
3. **Για το Preprocessing:**  
   Χρησιμοποιήσαμε τον *MInMaxScaler* για να κάνουμε το normalization. Ο *MInMaxScaler* για κάθε τιμή στο dataset, μετατρέπει όλες τις τιμές σε ένα εύρος [0, 1]. Ο μαθηματικός τύπος που χρησιμοποιεί είναι ο παρακάτω:  
   Text

   Description automatically generated with medium confidence  
   **Για το Learning:**   
   Στην **Linear Regression**, αρχικά χρησιμοποιήσαμε την *train\_test\_split* για να χωρίσουμε τα δεδομένα σε *train* και *test*. Η μεταβλητή X περιέχει τις στήλες **Open**, **High**, **Low**, **Volume** μετά το scaling και στην Y την στήλη **Close** μετά το scaling. Στη συνέχεια χρησιμοποιήσαμε το cross validation για να βρούμε το *R2 Score* του training καθώς και το αντίστοιχο score για το testing. Μετέπειτα, χρησιμοποιήσαμε τον *KFold* για 10 φορές και βρήκαμε το *R2 Score* καθώς και το Mean Squared Error για κάθε επανάληψη.

Έχοντας βρει το βέλτιστο μοντέλο, προχωρήσαμε την οπτικοποίηση των παραμέτρων του. Έτσι φτιάξαμε την γραφική παράσταση στην οποία κατέληξε αλλά και ένα dataframe με τις πραγματικές τιμές του test dataset και τις προβλέψεις για σύγκριση.

Όσον αφορά την **Logistic Regression**, αρχικά η μεταβλητή X περιέχει τις στήλες Open,High,Low,Volume και Close μετά το scaling και η Y την στήλη Up/Down. Η στήλη αυτή έχει 1 όταν η τιμή Open είναι μεγαλύτερη από την Close και 1 αλλιώς. Χρησιμοποιούμε αυτή τη στήλη γιατί θέλουμε να κάνουμε classification, κάτι που

δεν γίνεται με την Linear Regression. Όμοια με πριν, χρησιμοποιήσαμε την train\_test\_split για να χωρίσουμε τα δεδομένα σε train και test. Σε αυτή την περίπτωση ψάχνουμε το ROC AUC score για το training και το testing για αυτό χρησιμοποιούμε το cross validation. Τέλος, βρίσκουμε το Accuracy score καθώς και το ROC AUC score.

1. Στο νευρωνικό δίκτυο χρησιμοποιήσαμε το framework *Pytorch* για την δημιουργία της αρχιτεκτονικής και των προβλέψεων. Αρχικά δημιουργήσαμε την κλάση του δικτύου μας και ορίσαμε πως θα έχει 2 hidden layers τα τα οποία θα μετασχηματίζουν την κάθε είσοδο σε 25 εξόδους (Το νούμερο αυτο επιλέχθηκε καθώς είναι ο median μεταξύ των input nodes, δηλαδή 50 και του output node, 1).

Στη συνέχεια φορτώσαμε το dataset και το κάναμε scale. Η custom dataset class **TimeSeriesDataset** που φτιάξαμε, υλοποιεί στην ουσία ένα «κυλιόμενο παράθυρο» πάνω από τα δεδομένα μας έτσι ώστε να εκπαιδεύουμε σε 50 τιμές κάθε φορά.

Μετά τον διαχωρισμό σε train και test δεδομένα, ορίσαμε μια μέθοδο για training και μια για testing και προχωρήσαμε σε εκπαίδευση σε 20 *Epochs* (Οποιαδήποτε τιμή πάνω απο 20 μας έδινε diminishing returns και έτσι καταλήξαμε στο 20).

Τελικά πλοτάραμε το *MSE* και δημιουργήσαμε με αυτό έναν πίνακα στον οποίο συγκρίνουμε τα μοντέλα μας.

1. **Bonus – ARIMA**  
     
   Το μοντέλο αυτό αντί να χρησιμοποιεί προηγούμενες τιμές της μεταβλητής πρόβλεψης σε μια παλινδρόμηση, χρησιμοποιεί προηγούμενα σφάλματα πρόβλεψης.   
   Είναι πολύ σημαντικό να τονίσουμε πως για να χρησιμοποιήσουμε το ARIMA πρέπει τα δεδομένα μας να είναι σταθερά. Για αυτό τον λόγο χρησιμοποιήσαμε τα γνώσεις μας στην Στατιστική και εφαρμόσαμε τον κανόνα της Null Hypothesis, σύμφωνα με τον οποίο, η Null Hypothesis γίνεται δεκτή μόνο όταν το pvalue είναι μεγαλύτερο από 0.05. Οπότε αν θεωρήσουμε ότι η Null Hypothesis είναι ότι τα δεδομένα μας είναι σταθερά, φτιάξαμε μία συνάρτηση που λειτουργεί ακριβώς έτσι και μάς δίνει το pvalue. Αρχικά λοιπόν βλέπουμε ότι το pvalue είναι μεγαλύτερο από το 0.05, οπότε η Null Hypothesis πρέπει να απορριφθεί, άρα τα δεδομένα μας δεν είναι σταθερά. Η επόμενη επιλογή ήταν να χρησιμοποιήσαμε λογαρίθμους. Ακολουθούμε την ίδια διαδικασία αλλά το pvalue παραμένει υψηλό. Στη συνέχεια δοκιμάσαμε έναν πολύ διαδεδομένο τρόπο στο ARIMA, να αφαιρούμε την συνιστώσα της τάσης από μία συγκεκριμένη τιμή, στην συγκεκριμένη περίπτωση από τους λογαρίθμους, χρησιμοποιώντας την συνάρτηση shift(). Τώρα το pvalue είναι όντως μικρότερο από το 0.05, οπότε η Null Hypothesis είναι αποδεκτή και τα δεδομένα μας είναι σταθερά.   
   Τώρα που τα δεδομένα μας είναι σταθερά, μπορούμε να προχωρήσουμε στον χωρισμό των δεδομένων μας όπως κάνουμε παραδοσιακά στα μοντέλα πρόβλεψης. Χωρίσαμε τα δεδομένα μας σε train και test σε αναλογία 9:1 και φτιάξαμε το μοντέλο μας ώστε για κάθε τιμή του testing dataset, να εμφανίζεται η τιμή που είναι η σωστή, αυτή που προέβλεψε το μοντέλο μας καθώς και το error της πρόβλεψης. Το τελευταίο υπολογίζεται από την διαφορά της πραγματικής τιμής από την τιμή που βρήκε το μοντέλο διαιρεμένο με την πραγματική τιμή.  
     
   Στο σημείο αυτό πρέπει να μιλήσουμε για την αρχιτεκτονική του ARIMA. Το ARIMA παίρνει τις τιμές p,d,q κατά την εκπαίδευση. Το p δείχνει τον αριθμό των autoregressive terms,το d τον αριθμό των nonseasonal differences που χρειάζονται για την σταθερότητα και το q είναι ο αριθμός των lagged forecast errors. Η εκφώνηση ζητούσε το πρώτο q να είναι 1 και το δεύτερο 2. Εμείς επιλέξαμε για τα p,d τα 4 και 2 αντίστοιχα καθώς είδαμε πως με αυτές τις τιμές είχαμε τα καλύτερα αποτελέσματα στο διάγραμμα. Στη συνέχεια κάνουμε fit το μοντέλο μας και χρησιμοποιούμε την συνάρτηση forecast(), η οποία χρησιμοποιεί τις προηγούμενες τιμές για να προβλέψει τις επόμενες. Τέλος κάνουμε predict στο κάθε μοντέλο, βρίσκουμε το Mean Squared Error για κάθε μοντέλο και συγκρίνουμε τα αποτελέσματα.

Text

Description automatically generated