Θεωρία Αποφάσεων
1η Εργασία

Χειμερινό Εξάμηνο 2024 – 2025

24 Νοεμβρίου 2024

ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ
UNIVERSITY OF PATRAS

Τμήμα Μηχανικών Η / Υ και Πληροφορικής
Πολυτεχνική Σχολή

Όνομα: Μηλτιάδης
Επώνυμο: Μαντές
Α.Μ.: 1084661
E – mail: up1084661@ac.upatras.gr
Εξάμηνο: 9ο

Διδάσκων: Δημήτριος Κοσμόπουλος
Τομέας Εφαρμογών και Θεμελιώσεων της Επιστήμης των Υπολογιστών
Επιλεγόμενο Μάθημα – CEID_NE5237

ΘΕΜΑ: Πρόβλεψη Τιμών Μετοχών με Γραμμική Παλινδρόμηση

Περιεχόμενα

0 Εισαγωγή

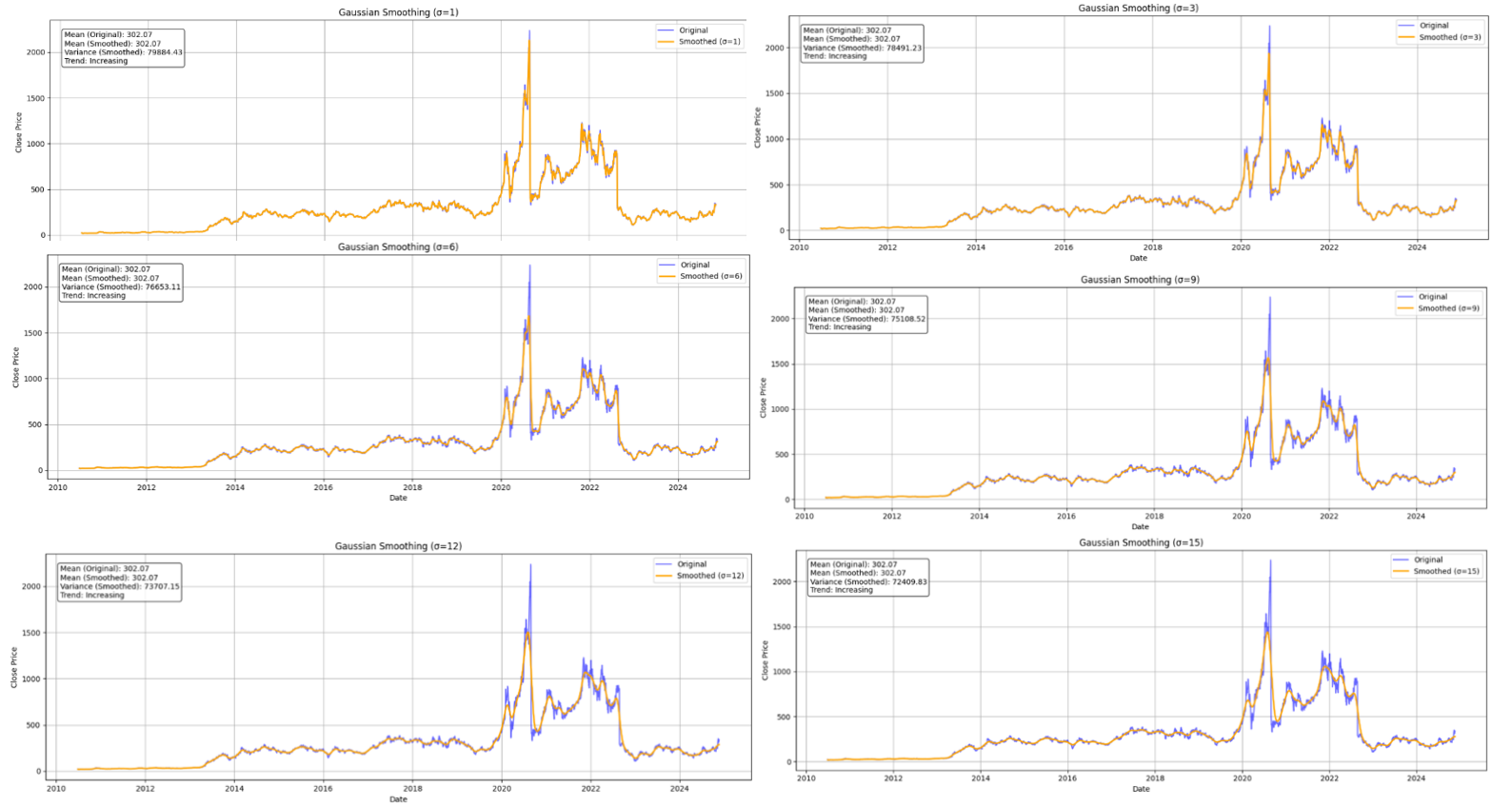
Το σύμβολο μετοχής που έχει επιλεχθεί είναι αυτό της **Tesla (TSLA)** και στο dataset περιλαμβάνονται οι τιμές κλεισίματός της στο χρηματιστήριο από τις 29/06/2010 έως τις 14/11/2024. Τα ωμά δεδομένα αποθηκεύονται στο αρχείο close_prices.csv από όπου θα φορτώνονται στην συνέχεια για περαιτέρω προεπεξεργασία και στατιστική μελέτη. Ως **training data** ορίζουμε τις τιμές κλεισίματος μέχρι τις 31/12/2023, ως **validation data** τις τιμές από 01/01/2024 έως 14/11/2024 και ως **test data** θέλουμε τις τιμές από τις 15/11/2024 και μετά.

Data Preprocessing

Ξεκινάμε την επεξεργασία με την εφαρμογή ενός **Gaussian Filter** στην αρχική χρονοσειρά με σκοπό να απομακρυνθούν οι ακραίες τιμές (outliers) και τυχόν θόρυβος και τελικά να προκύψει μια χρονοσειρά πιο εξομαλυμένη. Σκοπός είναι η τιμή της παραμέτρου sigma να είναι τέτοια ώστε να έχουμε ικανοποιητική αποθορυβοποίηση αλλά να μην συμβεί υπερβολικό smoothing στα δεδομένα και χαθεί έτσι σημαντική πληροφορία που υπάρχει ανάμεσα στις βραχυπρόθεσμες συσχετίσεις.

Για να κρίνουμε ποια τιμή sigma θα ήταν πιο κατάλληλη σχεδιάζουμε για ένα εύρος τιμών [1, 3, 6, 9, 12, 15] την αρχική και την εξομαλυμένη χρονοσειρά, ενώ ταυτόχρονα υπολογίζουμε τη μέση τιμή, τη διασπορά και τη τάση (ανοδική ή καθοδική) σε κάθε εξομαλυμένη χρονοσειρά. Έτσι, μπορούμε να δούμε οπτικά σε ποια περίπτωση το smoothing δεν «καταστρέφει» την αρχική πληροφορία και τα short-term variations, καθώς μας ενδιαφέρει η βραχυπρόθεσμη πρόβλεψη τιμών κλεισίματος (επόμενη μέρα). Γενικά, μια μεγάλη τιμή του sigma, όπως επιβεβαιώνουμε και από την Εικόνα 1, εφαρμόζει ισχυρή εξομάλυνση κάνοντας τα δεδομένα πιο αργά στο να αντιδράσουν σε μικρές διακυμάνσεις. Συνεπώς, μας συμφέρει η επιλογή ενός χαμηλότερου sigma, όπως οι τιμές 3 ή 6. **Επιλέγουμε ωστόσο να εργαστούμε με τιμή sigma = 3**, καθώς έτσι το μοντέλο φαίνεται να μαθαίνει καλύτερα τις τρέχουσες τάσεις.

Εικόνα 1: Gaussian Filtering for different values of sigma

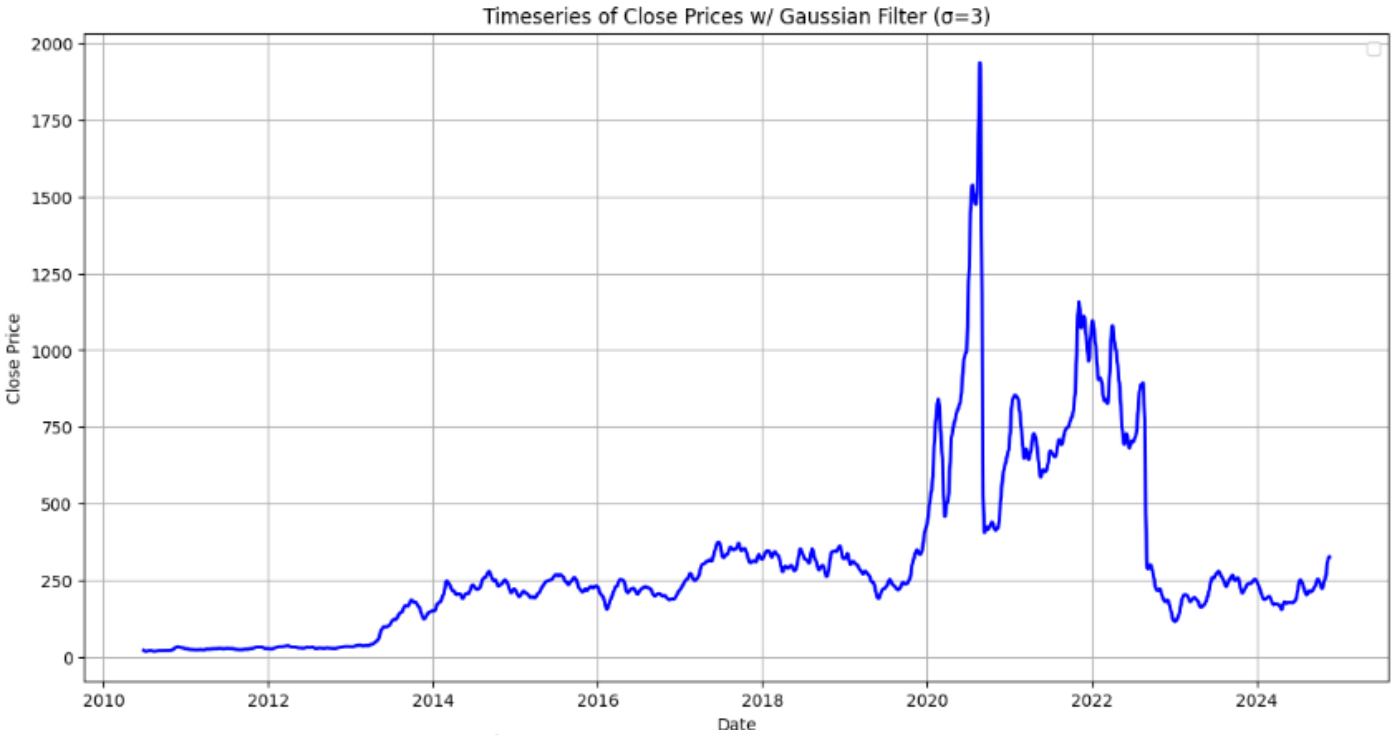


Γενικά, παρατηρώντας τη τελική χρονοσειρά που προκύπτει στην Εικόνα 2 μπορούμε να εξάγουμε τις εξής πληροφορίες για τα δεδομένα:

1. Οι τιμές κλεισίματος της μετοχής της TSLA παρουσιάζουν μια **έντονη ανοδική τάση** που ξεκινά περίπου το 2019, κορυφώνεται στα τέλη του 2021 και στη συνέχεια σημειώνει μια **σημαντική πτώση** το 2022.
2. Υπάρχουν **περίοδοι υψηλής μεταβλητότητας**, ιδιαίτερα κατά την περίοδο 2020-2022 πιθανόν λόγω COVID-19, με απότομες αυξήσεις και μειώσεις.
3. Μετά τη σημαντική πτώση το 2022, οι τιμές κλεισίματος φαίνεται να **σταθεροποιούνται**, κινούμενες γύρω από χαμηλότερα επίπεδα τιμών κατά την περίοδο 2023-2024.

Συνολικά, βλέπουμε ότι η εξαρτημένη μεταβλητή (τιμή κλεισίματος της μετοχής) παρουσιάζει μια **μη-γραμμική συμπεριφορά** και έχει ισχυρή **εξάρτηση από πρόσφατες, βραχυπρόθεσμες μεταβολές** και όχι από εξομαλυμένες, μακροπρόθεσμες τάσεις.

Αυτό το συμπέρασμα στο οποίο καταλήξαμε μας βοηθάει να επιλέξουμε επίσης το είδος των καθυστερημένων γνωρισμάτων (lagged features) με τα οποία θα τροφοδοτήσουμε τα μοντέλα πρόβλεψης στη συνέχεια. Γενικά, η δυναμική της τιμής κλεισίματος της TSLA αποτυπώνεται καλύτερα μέσω ενός μοντέλου που θα λαμβάνει υπόψη τις **πρόσφατες ημερήσιες μεταβολές**, καθώς αυτές οι διακυμάνσεις έχουν μεγαλύτερη προβλεπτική αξία σε σύγκριση με μια εξομαλυμένη θεώρηση εβδομαδιαίων τάσεων, η οποία θα απέκρυπτε τις λεπτομερείς καθημερινές μεταβολές. Συνεπώς, θα επιλέξουμε να χρησιμοποιήσουμε τις **ημερήσιες τιμές κλεισίματος ως χαρακτηριστικά για την εκπαίδευση** του μοντέλου μας, αντί για τους εβδομαδιαίους μέσους όρους.



Εικόνα 2: Smoothed timeseries with sigma = 3

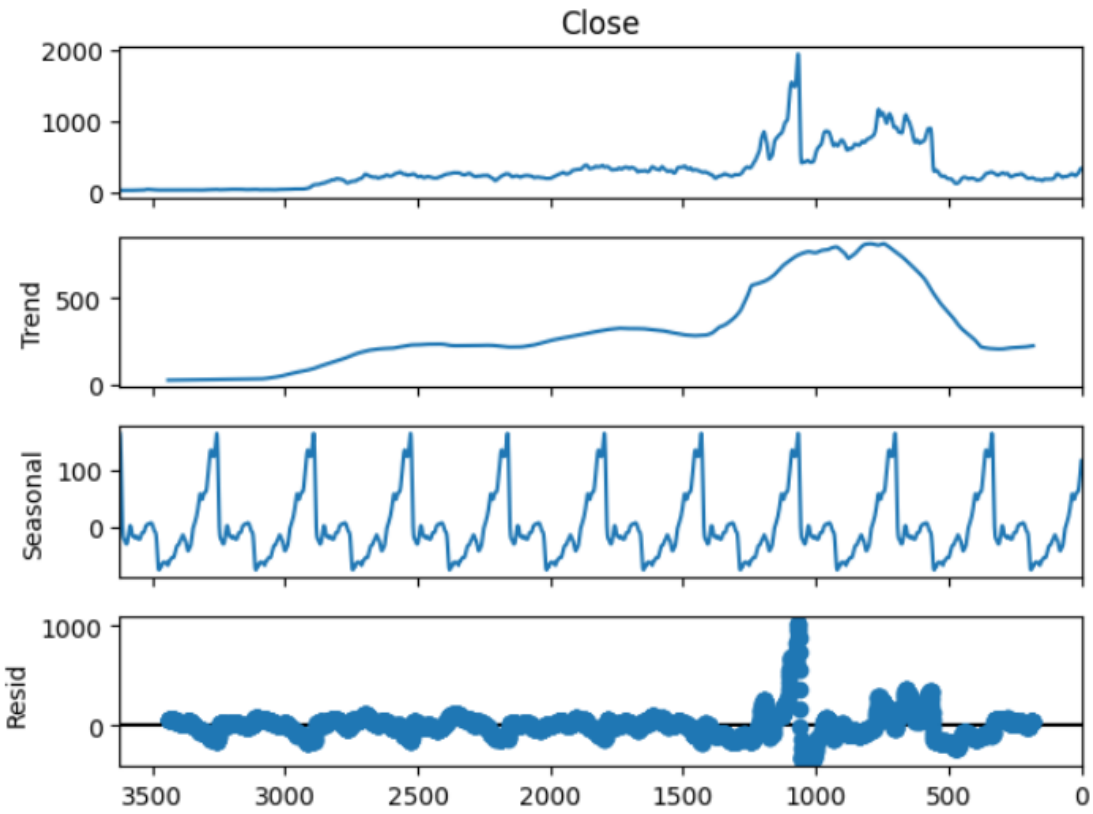
Statistic Study

Αρχικά, με την χρήση του **Simple Moving Average (SMA)** στοχεύουμε στο να εντοπίσουμε τη συνολική απόδοση της μετοχής σε μεσοπρόθεσμο ορίζοντα, καθώς απομονώνει τις πιο σημαντικές διακυμάνσεις, καθιστώντας τις τάσεις πιο ορατές. Όταν η τιμή κλεισίματος είναι πάνω από την τιμή του SMA, αυτό μπορεί να υποδηλώνει ανοδική τάση. Αντίθετα, όταν είναι κάτω από την SMA, μπορεί να υποδηλώνει πτωτική τάση. Συγκεκριμένα, από την Εικόνα 3 βλέπουμε πάλι ότι η μεταβλητότητα είναι **χαμηλή** για τις περιόδους πριν το 2019 και μετά το 2023, όπου οι τιμές είναι πιο σταθερές και κοντά στο SMA, και **υψηλή** κατά την περίοδο 2020-2022. Άρα, μας βοηθά συνολικά να ξεχωρίσουμε αν η απότομη αλλαγή στο διάστημα 2020-2022 είναι πραγματική ή προσωρινή. Πιθανότατα είναι **προσωρινή**, αφού καθώς σχετίζεται με ασυνήθιστα υψηλή μεταβλητότητα.

Εικόνα 3:: SMA with30-day rolling window

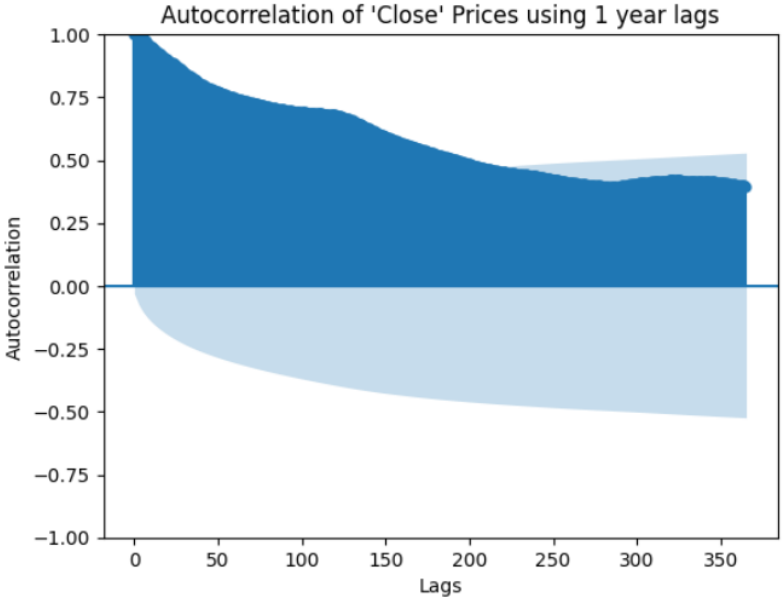
Έπειτα, επιχειρούμε να κάνουμε **seasonal decomposition** των δεδομένων για περιόδους ενός χρόνου. Από εδώ μπορούμε να εντοπίσουμε τη συνολική τάση, την εποχικότητα και τα residuals, δηλαδή τα υπόλοιπα δεδομένα αφού αφαιρεθούν η τάση και η εποχικότητα. Από την Εικόνα 4 παρατηρούμε τα εξής:

1. Η **τάση** είναι κυρίαρχα **αυξητική** πάλι **μέχρι το σημείο μέγιστης κορύφωσης** στο μέσον της χρονοσειράς. Μετά, η τάση **υποχωρεί**, αντικατοπτρίζοντας τη συνολική πτώση των τιμών. Αυτό επιβεβαιώνει τη γενική κατεύθυνση της μετοχής.
2. Όσον αφορά την **εποχικότητα** παρατηρούμε ένα **επαναλαμβανόμενο μοτίβο αυξομειώσεων με σταθερό πλάτος και συχνότητα**, το οποίο κύκλους που σχετίζονται με συγκεκριμένες περιόδους. Αυτή **η εποχικότητα είναι σταθερή** και επαναλαμβάνεται καθ' όλη τη διάρκεια της χρονοσειράς. Επίσης, δεν υπάρχουν πολύ έντονες εποχιακές διακυμάνσεις.
3. Παρατηρούμε από τα **residuals** ότι υπάρχει **μεγαλύτερη διακύμανση και ακραίες τιμές** στο υπόλοιπο **κατά τη διάρκεια της μεγάλης ανόδου και πτώσης**. Αυτό δείχνει ότι σε περιόδους έντονης μεταβλητότητας, υπάρχουν μη κανονικές αποκλίσεις από εξωγενείς παράγοντες που δεν εξηγούνται ούτε από την τάση ούτε από την εποχικότητα.



Εικόνα 4: Seasonal decomposition of 1-year periods

Τέλος, εφόσον έχουμε επιλέξει daily lagged features, υπολογίζουμε την **αυτοσυσχέτιση (ACF)** ανάμεσα στις τιμές κλεισίματος καθώς προχωράμε πίσω στο χρόνο, μέχρι να φτάσουμε τις 365 προηγούμενες τιμές. Από την Εικόνα 5 βλέπουμε πως τα καθυστερημένα χαρακτηριστικά παρουσιάζουν **ισχυρή βραχυπρόθεσμη συσχέτιση** και δεν υπάρχουν περιοδικές τάσεις, όπως εβδομαδιαία εποχικότητα. Επομένως, τα καθημερινά χαρακτηριστικά καθυστέρησης διατηρούν τις λεπτομέρειες που βοηθούν στην πιο ακριβή αποτύπωση αυτών των εξαρτήσεων. **Με υψηλό ACF, οι ξαφνικές αλλαγές ή οι τάσεις στις καθημερινές τιμές (όπως ανοδικές ή καθοδικές τάσεις) είναι πιθανό να συνεχίζονται.**



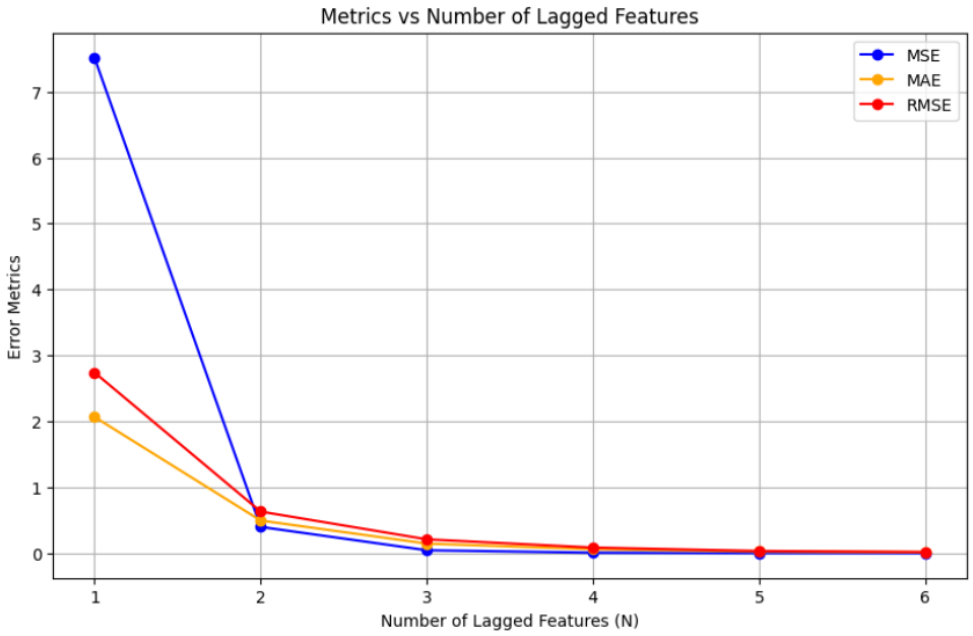
Εικόνα 5: Autocorrelation function of lagged features over a year

1 Γραμμική Παλινδρόμηση

Για την υλοποίηση του μοντέλου Γραμμικής Παλινδρόμησης ορίζουμε τη συνάρτηση **linear_regression_model( )**, η οποία παίρνει σαν ορίσματα το πλήθος των lagged features (**N**) για την εκπαίδευση και το DataFrame **data** που περιέχει όλα μας τα δεδομένα. Αρχικά, δημιουργούμε τα lags προσθέτοντας νέες στήλες στο DataFrame **data**, όπου κάθε στήλη περιέχει τιμές της στήλης 'Close' με μετατόπιση **i** θέσεις πίσω στο χρόνο, όπου **i** = 1, 2, .., **N**. Στη συνέχεια, χωρίζουμε τα δεδομένα σε training και validation sets με βάση τις ημερομηνίες που αναφέρθηκαν πριν και ορίζουμε τα χαρακτηριστικά εισόδου (X) και εξόδου (y). Για το X χρησιμοποιούμε τα **N** lags, δηλαδή τις νέες στήλες που προστέθηκαν, τόσο για το training όσο και για το validation set (**X_train**, **X_val**). Αντίστοιχα, για το y (**y_train**, **y_val**) χρησιμοποιούμε τις πραγματικές τιμές κλεισίματος της στήλης 'Close' του DataFrame. Έπειτα, δημιουργούμε το μοντέλο μέσω της κλάσης **LinearRegression( )** και καλούμε τη συνάρτηση **fit( )** πάνω στα σύνολα **X_train**, **y_train** ώστε το μοντέλο να μάθει τα βάρη (weights) και τη μεροληψία (bias) για την πρόβλεψη της τιμής κλεισίματος. Οι παράμετροι αυτές ανακτώνται στη συνέχεια με τις εντολές **model.coeff_** και **model.intercept_**. Τέλος, κάνουμε τη πρόβλεψη μέσω της συνάρτησης **predict( )** πάνω στο **X_val** και αποθηκεύουμε τις προβλέψεις στο DataFrame **y_pred**. Τα **y_val** και **y_val_pred** χρησιμοποιούνται για να υπολογίσουμε τις μετρικές σφάλματος **val_mae**, **val_rmse**, **val_mse** του συνόλου επικύρωσης. Όμοια, υπολογίζουμε και τις μετρικές σφάλματος **train_mae**, **train_rmse**, **train_mse** του συνόλου εκπαίδευσης, καλώντας την **predict( )** πάνω στο **X_train** και αποθηκεύοντας τις προβλέψεις στο DataFrame **y_train_pred**.

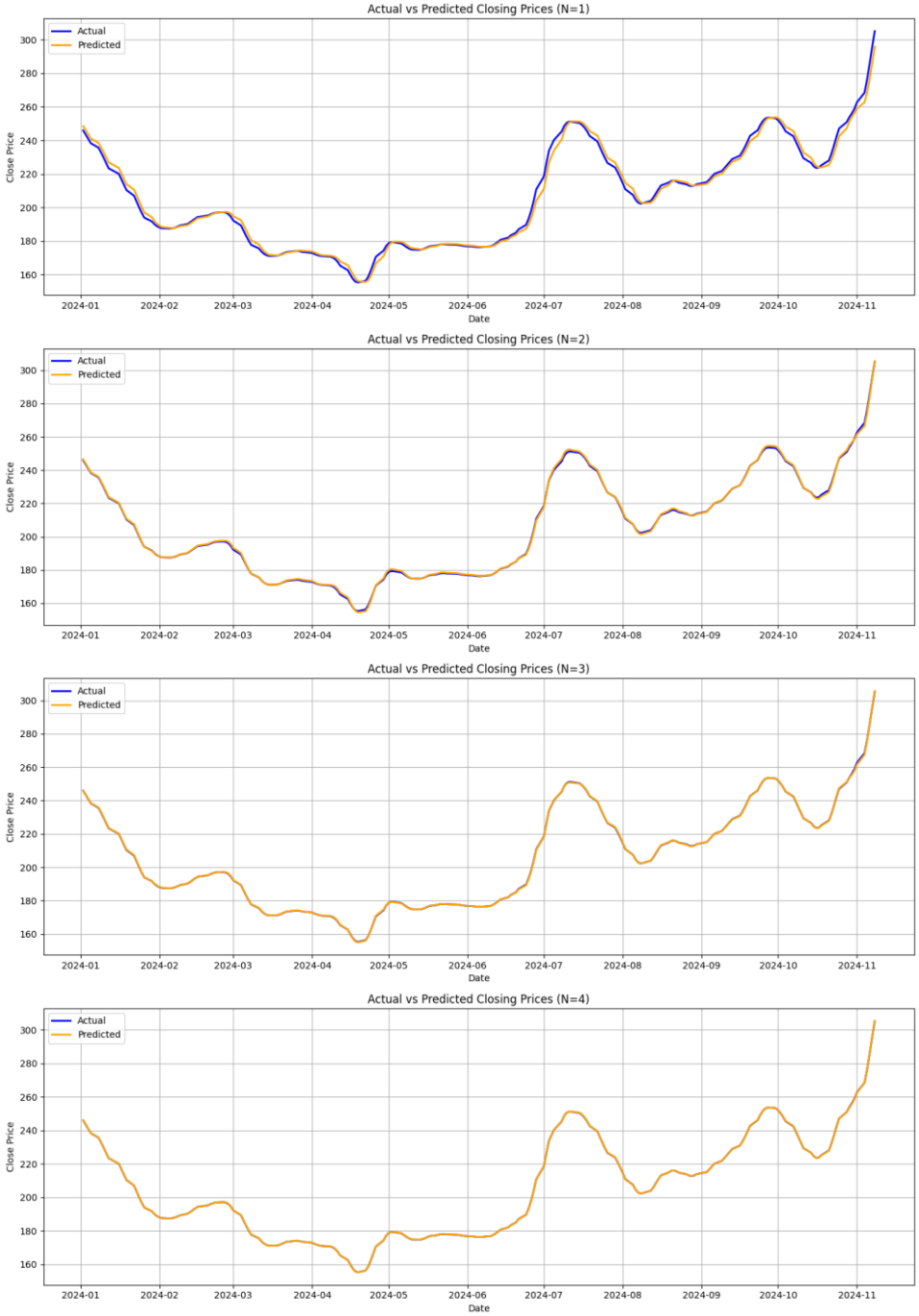
Τώρα θέλουμε να ελέγξουμε ποιο είναι το κατάλληλο πλήθος από lags που πρέπει να χρησιμοποιήσουμε ως χαρακτηριστικά εκπαίδευσης. Γι' αυτό το λόγο, σχεδιάζουμε τις 3 μετρικές σφάλματος που προκύπτουν για πλήθος lags από 1 έως 6. Αρχικά, ορίζουμε ένα DataFrame **results**, το οποίο θα περιέχει τα διαφορετικά πλήθη από lags και τις μετρικές που αντιστοιχούν σε αυτά. Έπειτα, για **i** = 1, 2, .., 6 καλούμε τη συνάρτηση **linear_regression_model( )** και αποθηκεύουμε στο **results** τις μετρικές που επιστρέφει το μοντέλο κάθε φορά, ενώ παράλληλα εκτυπώνουμε την εξίσωση του μοντέλου για κάθε περίπτωση, καθώς και τις γραφικές των **y_val**, **y_pred** για να δούμε οπτικά πόση απόκλιση έχει η πραγματική τιμή από τη προβλεπόμενη. Στο τέλος, αφού το **results** έχει γεμίσει με τις κατάλληλες τιμές το χρησιμοποιούμε για να εκτυπώσουμε τη συνολική γραφική που μας δείχνει πως μεταβάλλεται το σφάλμα σε σχέση με τα διάφορα lags τόσο για το training, όσο και το validation set.

Εικόνα 6: Validation Error Metrics vs Number of Lagged Features



Από την Εικόνα 6 βλέπουμε ότι από το validation set το **κατάλληλο πλήθος lags είναι 3-4**, καθώς από αυτό το σημείο και έπειτα τα σφάλματα τείνουν να σταθεροποιηθούν γύρω από το 0. Συνεπώς, το μοντέλο δεν χρειάζεται παραπάνω χαρακτηριστικά από αυτά για να εκπαιδευτεί σωστά, ενώ η χρήση παραπάνω από 4 θα μπορούσε να επιφέρει

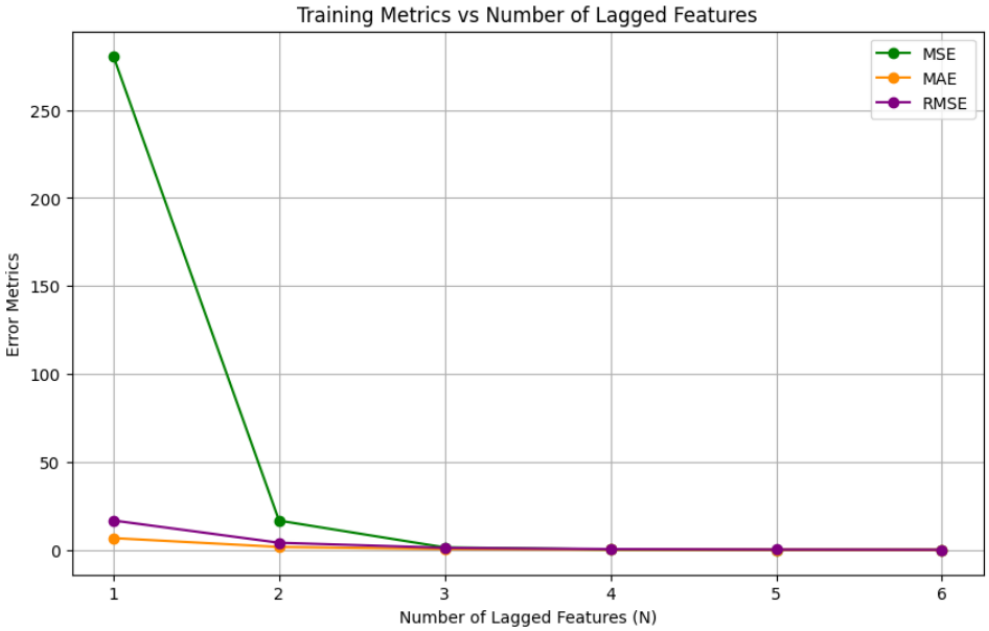
overfitting ή εισαγωγή θορύβου στις προβλέψεις. **Έτσι, τα 3 lags θα ήταν τα ιδανικότερα το μοντέλο μας.**
Επίσης, βλέπουμε γραφικά το πόσο καλά προσεγγίζει η πρόβλεψη τις πραγματικές τιμές πάνω στη χρονοσειρά του συνόλου επικύρωσης.



Εικόνα 7: Actual vs Predicted closing prices

Γενικά, παρατηρούμε ότι **για 3-4 lags το μοντέλο αποδίδει πολύ καλά στα δεδομένα του validation set**, καθώς τα δεδομένα του 2024 παρουσιάζουν μια σταθερή τάση χωρίς απότομες μεταβολές. Αυτό είναι λογικό, καθώς το μοντέλο υποθέτει μια γραμμική σχέση μεταξύ των εξαρτημένων και ανεξάρτητων μεταβλητών. Οπότε, όταν τα δεδομένα ακολουθούν flat trend η γραμμική σχέση είναι πιο ρεαλιστική και πιο εύκολο να προβλεφθεί εφόσον δεν υπάρχουν μη γραμμικά μοτίβα που θα μπορούσαν να «μπερδέψουν» το μοντέλο.

Πάμε τώρα να μελετήσουμε και τις μετρικές σφάλματος που προκύπτουν από το training set. Από την Εικόνα 8 επιβεβαιώνουμε ότι πάλι το μοντέλο φαίνεται να εκπαιδεύεται πολύ καλά με μικρό αριθμό lagged features, και η επιπλέον αύξηση των lags δεν οδηγεί σε σημαντική βελτίωση για το training set. Συγκρίνοντας έτσι τις δύο γραφικές παραστάσεις είναι εμφανές ότι είναι σχεδόν ίδιες όσον αφορά την πορεία που ακολουθεί το σφάλμα, γεγονός το οποίο σημαίνει ότι **το μοντέλο γενικεύει καλά στα νέα δεδομένα (validation set), δηλαδή δεν έχει υπερπροσαρμοστεί (overfitting) στο training set**. Άρα, μπορούμε να είμαστε σίγουροι ότι το μοντέλο είναι **αξιόπιστο** και έχει **καλή απόδοση** σε δεδομένα που δεν έχει «μάθει» κατά τη διάρκεια της εκπαίδευσης.



Εικόνα 8: Training Error Metrics vs Number of Lagged Features

Τέλος, βλέπουμε συνολικά τις μετρικές σφάλματος τόσο για το training όσο και για το validation set συγκεντρωμένες εδώ:

Metrics

	N	Train MAE	Train RMSE	Train MSE	Val MAE	Val RMSE	Val MSE
0	1	3.443139	10.900343	118.817473	2.098840	2.789584	7.781780
1	2	0.874326	2.678789	7.175912	0.475835	0.639556	0.409031
2	3	0.263221	0.781161	0.610213	0.147340	0.213131	0.045425
3	4	0.109645	0.314319	0.098796	0.059626	0.085433	0.007299
4	5	0.048284	0.145148	0.021068	0.026038	0.035424	0.001255
5	6	0.024742	0.072313	0.005229	0.014432	0.018842	0.000355

Όσον αφορά τις παραμέτρους, παρακάτω βλέπουμε τις παραμέτρους και την εξίσωση του μοντέλου Γραμμικής Παλινδρόμησης για κάθε διαφορετική τιμή του πλήθους των lagged features. **Για τη πρόβλεψη θα χρησιμοποιηθεί το μοντέλο με την εξίσωση που αντιστοιχεί σε 3 lagged features.** Οι **παράμετροι** του μοντέλου που θα χρησιμοποιηθεί σημειώνονται με έντονη γραφή πιο κάτω:

Model Parameters for N=1:
Bias: 0.3304377209988729
Weight for close_t-1: 0.999144280448192
Model Equation:
Close_t = 0.3304377209988729 + (0.999144280448192) * close_t-1

Model Parameters for N=2:
Bias: 0.4382810225101821
Weight for close_t-1: 1.9678031348385083
Weight for close_t-2: -0.9692191483338409
Model Equation:
Close_t = 0.4382810225101821 + (1.9678031348385083) * close_t-1 + (-0.9692191483338409) * close_t-2

Model Parameters for N=3:
Bias: 0.0255841839544928

Weight for close_t-1: 2.895064203117802
Weight for close_t-2: -2.8515820591799637
Weight for close_t-3: 0.9564478202406808
Model Equation:
Close_t = 0.0255841839544928 + (2.895064203117802) * close_t-1 + (-2.8515820591799637) * close_t-2 + (0.9564478202406808) * close_t-3

Model Parameters for N=4:
Bias: 0.03687320187998466
Weight for close_t-1: 3.7708830642200537
Weight for close_t-2: -5.462541660748848
Weight for close_t-3: 3.606996135626407
Weight for close_t-4: -0.915455917228488
Model Equation:
Close_t = 0.03687320187998466 + (3.7708830642200537) * close_t-1 + (-5.462541660748848) * close_t-2 + (3.606996135626407) * close_t-3 + (-0.915455917228488) * close_t-4

Model Parameters for N=5:
Bias: 0.005203760192443951
Weight for close_t-1: 4.583151471355396
Weight for close_t-2: -8.662750884007773
Weight for close_t-3: 8.453180183456398
Weight for close_t-4: -4.260641543070784
Weight for close_t-5: 0.8870460406916396
Model Equation:
Close_t = 0.005203760192443951 + (4.583151471355396) * close_t-1 + (-8.662750884007773) * close_t-2 + (8.453180183456398) * close_t-3 + (-4.260641543070784) * close_t-4 + (0.8870460406916396) * close_t-5

Model Parameters for N=6:
Bias: 0.00783940717559517
Weight for close_t-1: 5.352602883020441
Weight for close_t-2: -12.358357394601152
Weight for close_t-3: 15.7849468434362
Weight for close_t-4: -11.773799658545894
Weight for close_t-5: 4.86178012391637
Weight for close_t-6: -0.8671978433110308
Model Equation:
Close_t = 0.00783940717559517 + (5.352602883020441) * close_t-1 + (-12.358357394601152) * close_t-2 + (15.7849468434362) * close_t-3 + (-11.773799658545894) * close_t-4 + (4.86178012391637) * close_t-5 + (-0.8671978433110308) * close_t-6

Προχωράμε τώρα στη πρόβλεψη για τα νέα δεδομένα με το μοντέλο που επιλέξαμε. Αρχικά, ορίζουμε τη συνάρτηση **predict_next_day_close_price( )** με ορίσματα το DataFrame **df**, τις παραμέτρους του μοντέλου **weights**, **bias** που επιστρέφει η συνάρτηση **linear_regression_model( )** και το πλήθος των lags **N**. Η συνάρτηση αυτή αρχικά αποθηκεύει στη **lagged_features** τις τελευταίες **N** τιμές κλεισίματος μέχρι τις 14/11/2024**.** Έπειτα, εξάγουμε τις τιμές των lagged features για την τελευταία διαθέσιμη ημέρα από το DataFrame **df** με την εντολή **iloc[-1].values.** Τέλος, εκχωρούμε τη προβλεπόμενη τιμή στη μεταβλητή **prediction_data** με βάση το τύπο **close_t =** $\sum_{i=1}^{N} w_i \cdot close\_t - i + b$ αντικαθιστώντας με τα κατάλληλα βάρη και μεροληψία.

Δεδομένου ότι έχουμε γνωστά δεδομένα μέχρι τις 14/11/2024 η πρόβλεψη για την επόμενη μέρα (15/11/2024) από το μοντέλο αν τρέξουμε τους κώδικες φαίνεται παρακάτω:

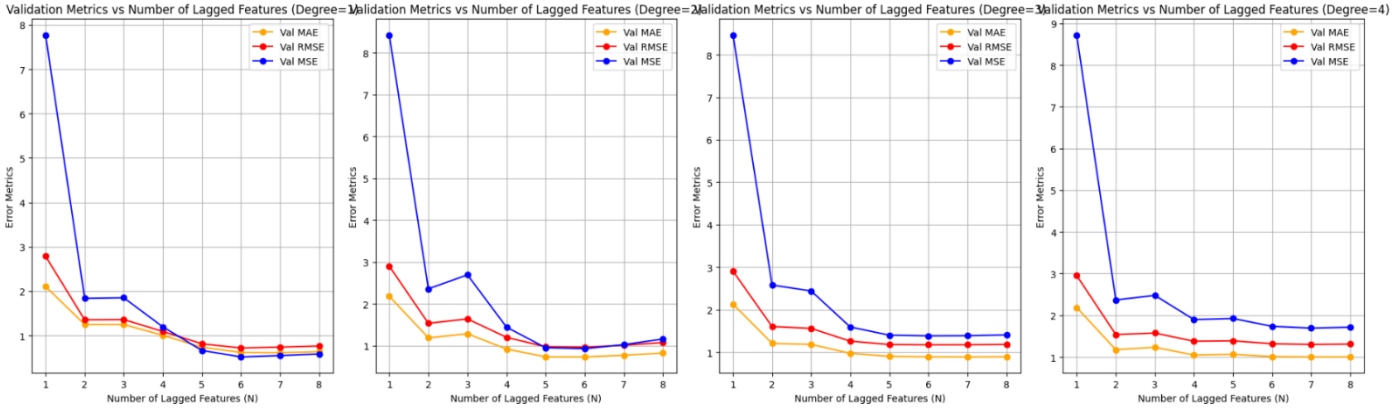
```
Predicted close price for 2024-11-15: 321.92 $
```

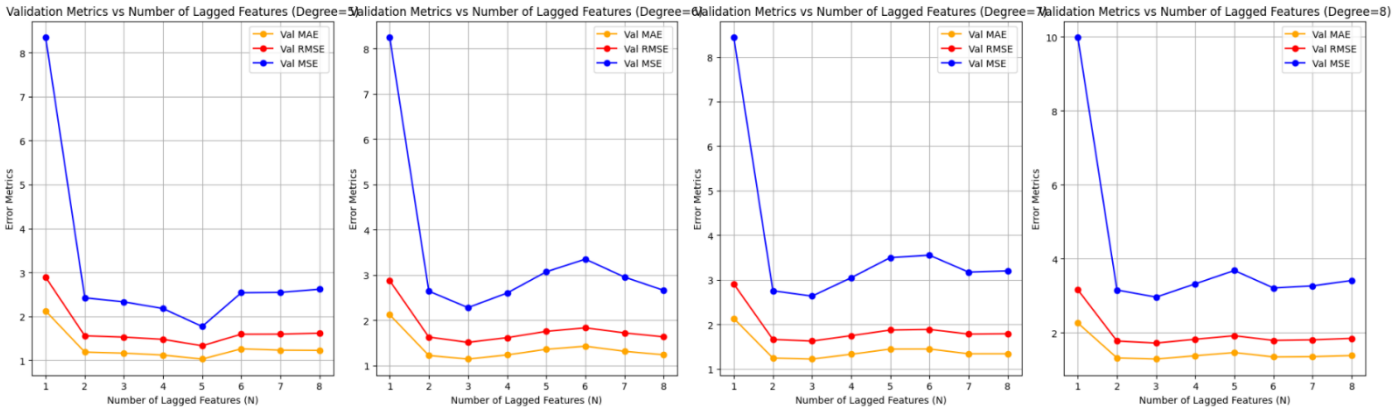
2 Πολυωνυμική Παλινδρόμηση με Κανονικοποίηση Lasso

Για την υλοποίηση του μοντέλου Πολυωνυμικής Παλινδρόμησης με Κανονικοποίηση Lasso ορίζουμε τη συνάρτηση **polynomial_regression_model_L1( )**, η οποία παίρνει σαν ορίσματα τον βαθμό του πολυωνύμου **degree**, το πλήθος των lagged features (**N**) για την εκπαίδευση και το DataFrame **data** που περιέχει όλα μας τα δεδομένα. Η δημιουργία των **N** lagged features καθώς και των συνόλων X και y για τα training και validation sets γίνεται ακριβώς με τον ίδιο τρόπο με πριν. Τα σύνολα **X_train**, **X_val** επεκτείνονται σε **X_train_poly**, **X_val_poly**, τα οποία περιέχουν νέα χαρακτηριστικά βασισμένα στις πολυωνυμικές σχέσεις μέχρι τον βαθμό που ορίζεται από τη μεταβλητή **degree**. Αυτό πραγματοποιείται καλώντας την **PolynomialFeatures( )** με βαθμό πολυωνύμου ίσο με **degree**. Μια άλλη υπερπαράμετρος που πρέπει να προσδιοριστεί κατά την εκπαίδευση του μοντέλου είναι ο συντελεστής **alpha**, ο οποίος ορίζει την ποινή για μεγάλα βάρη στη συνάρτηση κόστους. Αρχικά, ορίζουμε το **param_grid**, το οποίο περιέχει όλες τις διαφορετικές τιμές του alpha. Προκειμένου να βρεθεί η βέλτιστη τιμή του συντελεστή εφαρμόζουμε 5 – fold cross validation μέσω της συνάρτησης **GridSearchCV( )**για να αξιολογήσουμε την απόδοση του μοντέλου που δημιουργείται από τη κλήση της **Lasso( )** με μετρική το αρνητικό μέσο τετραγωνικό σφάλμα. Αυτή η διαδικασία γίνεται για **max_iter =10000** επαναλήψεις, προκειμένου να επιτευχθεί σύγκλιση. Εν τέλει, αν εκτυπώσουμε την βέλτιστη τιμή του **alpha**, θα δούμε ότι αυτή είναι το 0.01, άρα για την δημιουργία του μοντέλου στη συνέχεια αυτή θα είναι η τιμή που θα χρησιμοποιείται by default. Τέλος, καλούμε τη συνάρτηση **fit( )** πάνω στα σύνολα **X_train_poly**, **y_train** και κάνουμε πάλι τη πρόβλεψη για το validation set μέσω της συνάρτησης **predict( )** πάνω στο **X_val_poly** αποθηκεύοντας τις προβλέψεις στο DataFrame **y_val_pred**. Όλες οι άλλες παράμετροι, δηλαδή τα βάρη και η μεροληψία, καθώς και οι μετρικές σφάλματος **val_mae**, **val_rmse**, **val_mse** υπολογίζονται ακριβώς όπως και πριν. Το ίδιο συμβαίνει επίσης και για το training set και τις μετρικές **train_mae**, **train_rmse**, **train_mse.**

Στη συνέχεια, πρέπει να βρούμε ποιος είναι ο συνδυασμός εκείνος από lagged features και **degree**, ο οποίος έχει τη βέλτιστη απόδοση. Γι' αυτό το λόγο, σχεδιάζουμε τις 3 μετρικές σφάλματος που προκύπτουν για πλήθος lags από 1 έως 8 για όλους τους διαφορετικούς βαθμούς πολυωνύμου επίσης από 1 έως 8. Αρχικά, ορίζουμε ένα DataFrame **all_results**, το οποίο θα περιέχει για κάθε βαθμό πολυωνύμου τα διαφορετικά πλήθη από lags και τις μετρικές που αντιστοιχούν σε αυτά. Έπειτα, ορίζουμε το DataFrame **results**, το οποίο πάλι όπως πριν θα περιέχει τα διαφορετικά πλήθη από lags και τις μετρικές που αντιστοιχούν σε αυτά. Έπειτα, για **i** = 1, 2, .., 8 καλούμε τη συνάρτηση **polynomial_regression_model_L1( )** και αποθηκεύουμε στο **results** τις μετρικές που επιστρέφει το μοντέλο κάθε φορά. Στο τέλος, αφού το **all_results** έχει γεμίσει με τις κατάλληλες τιμές, το χρησιμοποιούμε για να εκτυπώσουμε τη συνολική γραφική που μας δείχνει πως μεταβάλλεται το σφάλμα σε σχέση με τα διάφορα lags για κάθε βαθμό πολυωνύμου στα training, validation sets.

Εικόνα 9: Validation Error Metrics vs Number of Lagged Features for each degree

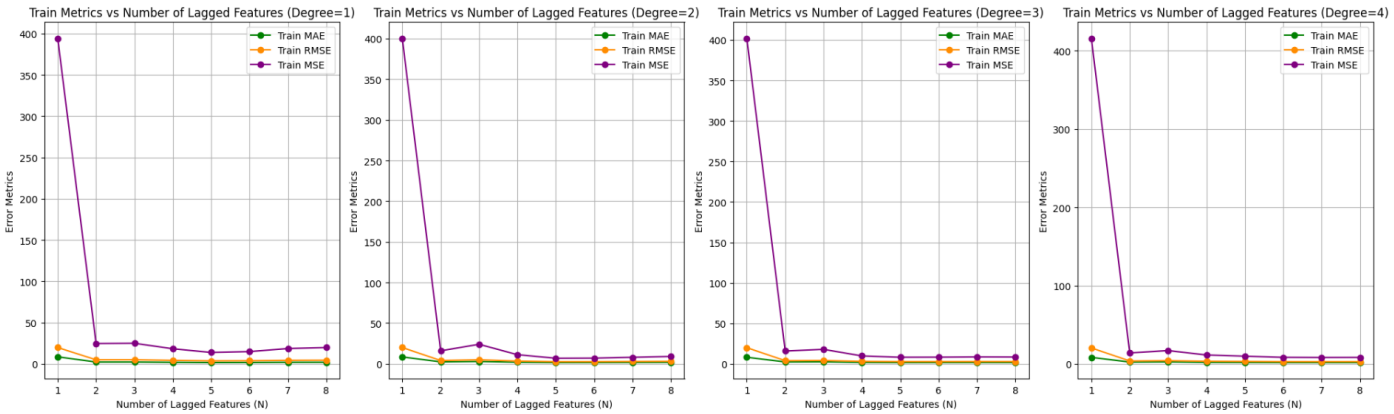


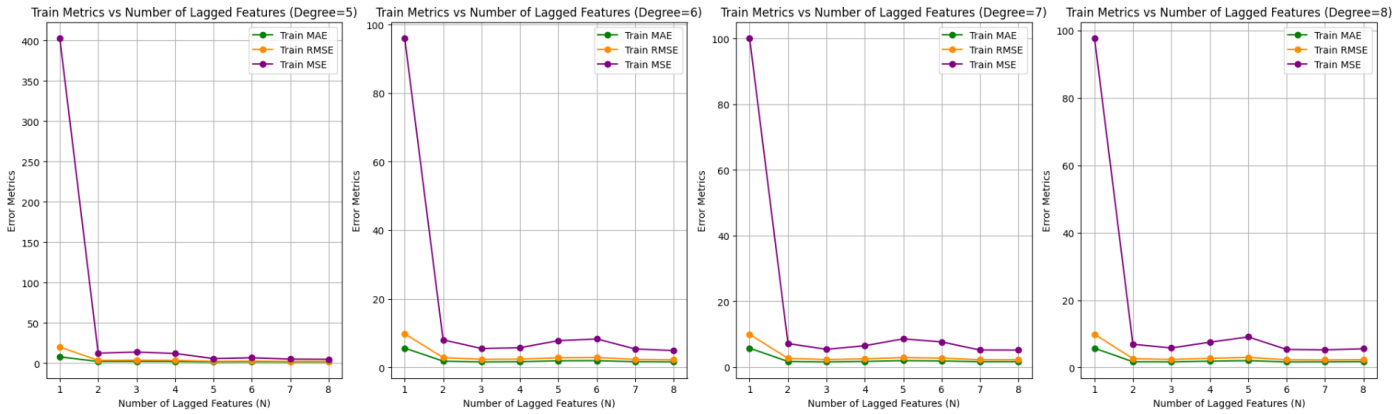


Από την Εικόνα 9 συμπεραίνουμε ότι **το καλύτερο trade – off ανάμεσα σε πολυπλοκότητα και απόδοση για το μοντέλο προκύπτει για βαθμό πολυωνύμου 3 – 4 και για πλήθος lagged features ξανά από 3 – 4**, όπως και στη Γραμμική Παλινδρόμηση. Για βαθμό πολυωνύμου πάνω από 5 βλέπουμε ότι το σφάλμα εμφανίζει ξανά ανοδική τάση, γεγονός το οποίο μας δείχνει ότι είναι πιθανόν να συμβαίνει υπερπροσαρμογή. Γενικά, εκτός από τη περίπτωση όπου **degree** = 1, βλέπουμε ότι για όλες τις διαφορετικές τιμές του **degree** το σφάλμα τείνει να σταθεροποιηθεί σε μια τιμή μετά τη χρήση 4 lagged features, οπότε η χρήση περισσότερων γνωρισμάτων δεν προσφέρει επιπλέον βελτίωση στην απόδοση του μοντέλου. Ωστόσο, σε αντίθεση με το Γραμμικό Μοντέλο όπου το σφάλμα τείνει στο 0, εδώ βλέπουμε ότι το σφάλμα σταθεροποιείται σε μια τιμή αισθητά πιο μεγάλη (κοντά το 2). Συνεπώς, ναι μεν το σφάλμα είναι σε αποδεκτό εύρος, μιας και δείχνει ότι το μοντέλο έχει φτάσει σε μια καλή ισορροπία χωρίς να χρειάζεται περισσότερη πολυπλοκότητα, ωστόσο **η Κανονικοποίηση με Lasso σίγουρα μοιάζει να αποδίδει χειρότερα από το Γραμμικό Μοντέλο**. Επίσης, όσον αφορά το βαθμό πολυωνύμου βλέπουμε πως για **degree** = 2 το μοντέλο είναι απλό και αποδοτικό αλλά μπορεί να υστερεί σε σύνθετες μη – γραμμικές σχέσεις. Από την άλλη, για **degree** = 5 και πάνω αυξάνεται η πολυπλοκότητα χωρίς σημαντική βελτίωση στην απόδοση, οδηγώντας έτσι σε πιθανά προβλήματα overfitting. Επομένως, για **degree** = 3,4 επιτυγχάνεται ο πιο ικανοποιητικός συνδυασμός, καθώς μπορούν να εντοπιστούν σωστά οι πιο σύνθετες σχέσεις ανάμεσα στα δεδομένα μας χωρίς να συμβεί «έκρηξη» στο πλήθος των συντελεστών του πολυωνύμου. **Έτσι, για να συνεχίσουμε στη πρόβλεψη επιλέγουμε N = 3 και degree = 3**.

Σχετικά με τις μετρικές σφάλματος του training set, τις βλέπουμε πιο κάτω στην Εικόνα 10:

Εικόνα 10: Validation Error Metrics vs Number of Lagged Features for each degree





Στο training set, οι μετρικές σφάλματος είναι πολύ χαμηλές, ακόμη και για υψηλούς βαθμούς (degree) και αριθμούς lagged features. Αυτό δείχνει ότι το μοντέλο μαθαίνει τα δεδομένα εκπαίδευσης πολύ καλά, πιθανώς «υπερβολικά καλά». Από την άλλη, η σταθεροποίηση των μετρικών στο validation set σε υψηλότερο επίπεδο σε σύγκριση με το training set υποδεικνύει ότι το μοντέλο **δυσκολεύεται να γενικεύσει καλά**. Αυτό είναι ένα ξεκάθαρο σημάδι ότι το μοντέλο γίνεται υπερβολικά πολύπλοκο για να ταιριάζει στα δεδομένα εκπαίδευσης, με κόστος τη γενίκευση. Συνεπώς, βλέπουμε ότι για το συγκεκριμένο μοντέλο **η απόδοση είναι χειρότερη σε σχέση με τη Γραμμική Παλινδρόμηση**, κάτι που επιβεβαιώνεται και στη συνέχεια όταν κάνουμε την πρόβλεψη.

Τελικά, οι **υπερπαράμετροι** που θα χρησιμοποιηθούν είναι:

1. **degree** = 3
2. **alpha** = 0.01
3. **max_iter** = 10,000

Η τιμή του **alpha** = 0.01 που προέκυψε από το cross validation, υποδηλώνει ότι εφαρμόζεται ισχυρότερη ποινή στους συντελεστές, οπότε το μοντέλο μπορεί να διατηρεί περισσότερη πολυπλοκότητα και να καταγράφει καλύτερα τις σχέσεις στα δεδομένα εκπαίδευσης. Έτσι, θα διασφαλίζει ότι τα πιο σημαντικά γνωρίσματα και οι υψηλότεροι συντελεστές μπορούν να παραμείνουν σημαντικοί, ενώ τα μη σημαντικά μπορεί να συρρικνωθούν κοντά στο μηδέν, ελαχιστοποιώντας έτσι το overfitting. Η τιμή του **max_iter** = 10,000 από την άλλη επηρεάζει την ταχύτητα εκπαίδευσης και τη σύγκλιση του μοντέλου.

Προχωράμε τώρα στη πρόβλεψη για τα νέα δεδομένα με το μοντέλο που επιλέξαμε. Αρχικά, ορίζουμε τη συνάρτηση **predict_next_day_close_price( )** με ορίσματα το DataFrame **df**, τις παραμέτρους του μοντέλου **weights**, **bias** που επιστρέφει η συνάρτηση **polynomial_regression_model_L1( )**, τον βαθμό **degree** του πολυωνύμου και το πλήθος των lags **N**. Η συνάρτηση αυτή αρχικά αποθηκεύει στη **lagged_features** τις τελευταίες **N** τιμές κλεισίματος μέχρι τις 14/11/2024**.** Έπειτα, δημιουργούμε τα αντίστοιχα στοιχεία μέσω της **PolynomialFeatures( )** και εξάγουμε πάλι τις τιμές των lagged features για την τελευταία διαθέσιμη ημέρα από το DataFrame **df** με την εντολή **iloc[-1].values.** Τέλος, εκχωρούμε τη προβλεπόμενη τιμή στη μεταβλητή **prediction_data_poly** με βάση τον ίδιο τύπο με πριν.

Για την εκτύπωση της εξίσωσης του μοντέλου ορίζουμε τη συνάρτηση **print_polynomial_equation( )** με ορίσματα τις παραμέτρους του μοντέλου **weights**, **bias** που επιστρέφει η συνάρτηση **polynomial_regression_model_L1( )**, τον βαθμό **degree** του πολυωνύμου και το πλήθος των lags **N**. Ξεκινάμε με την εξίσωση που περιλαμβάνει το **bias**. Έπειτα, ο εξωτερικός βρόχος διατρέχει τους βαθμούς του πολυωνύμου και ο εσωτερικός τα lagged features μέσω του **feature_idx**. Αν ο βαθμός είναι 1 προστίθεται ο όρος της μορφής weigths[feature_idx]*close_t-i + 1. Αν ο βαθμός είναι μεγαλύτερος από 1 προστίθεται ο όρος

της μορφής weigths[feature_idx]*close_t-i + 1^{degree}. Στο τέλος, η συνάρτηση εκτυπώνει την πλήρη πολυωνυμική εξίσωση **equation**.

Δεδομένου ότι έχουμε γνωστά δεδομένα μέχρι τις 14/11/2024 η πρόβλεψη για την επόμενη μέρα (15/11/2024) από το μοντέλο αν τρέξουμε τους κώδικες φαίνεται παρακάτω:

```
Predicted close price for 2024-11-15: 322.96 $
```

Τέλος, οι **παράμετροι** του μοντέλου που χρησιμοποιήσαμε τελικά (δηλαδή οι συντελεστές) αποτυπώνονται στη παρακάτω εξίσωση:

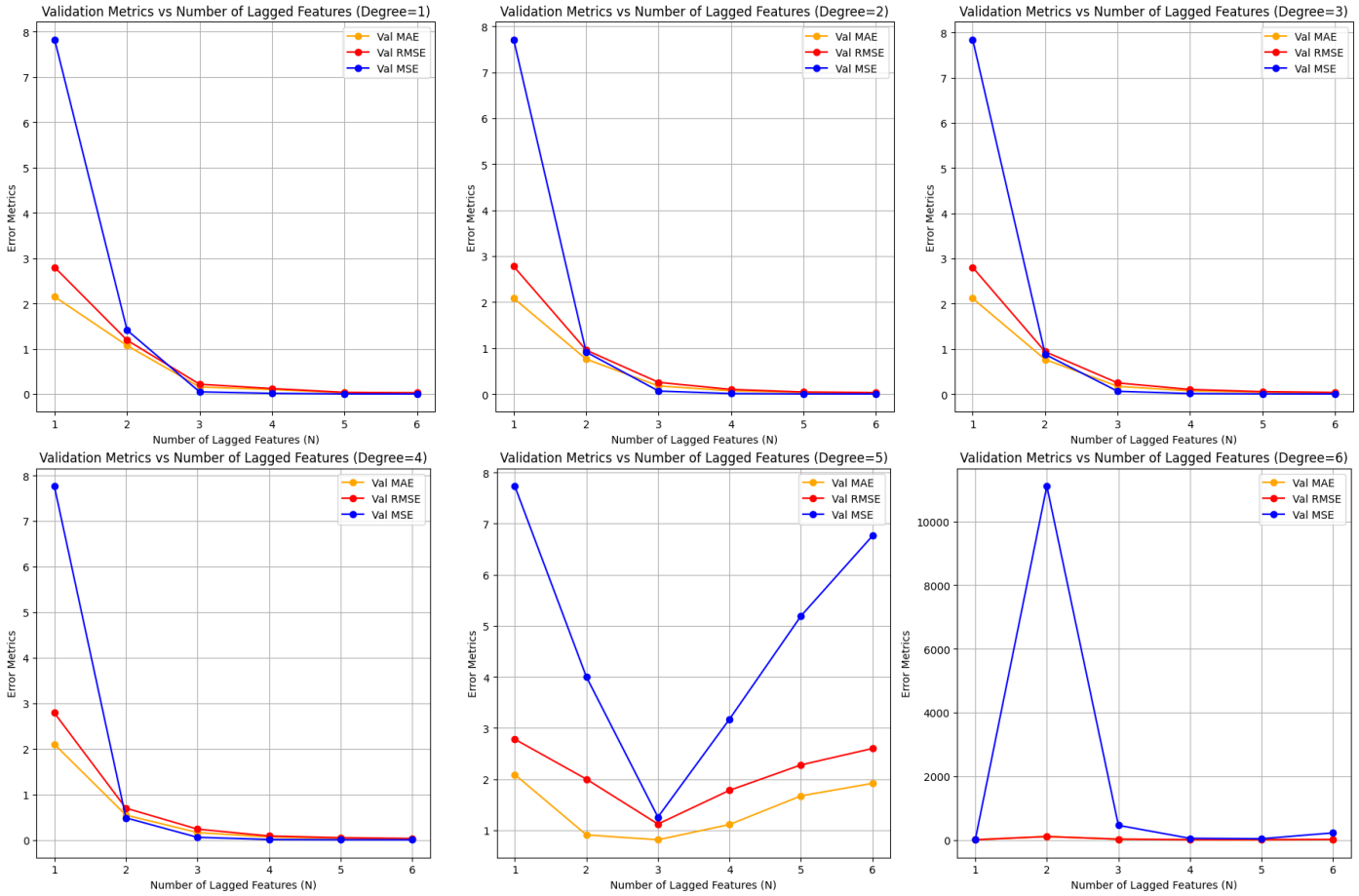
Polynomial Equation: Close_t = **3.0015269136335974** + (**0.0**) * close_t-1 + (**1.2980066191979724**) * close_t-2 + (**-0.18636784999643352**) * close_t-3 + (**-0.1304114765870124**) * close_t-1^2 + (**0.00019471155672510883**) * close_t-2^2 + (**-5.25966641051802e-05**) * close_t-3^2 + (**-2.5991818616084295e-05**) * close_t-1^3 + **(-2.7616551676277555e-05**) * close_t-2^3 + (**-2.685740580748707e-05**) * close_t-3^3

3 Πολυωνυμική Παλινδρόμηση με Κανονικοποίηση Ridge

Για την υλοποίηση του μοντέλου Πολυωνυμικής Παλινδρόμησης με Κανονικοποίηση Ridge ορίζουμε τη συνάρτηση **polynomial_regression_model_L2( )**, η οποία παίρνει σαν ορίσματα τον βαθμό του πολυωνύμου **degree**, το πλήθος των lagged features (**N**) για την εκπαίδευση και το DataFrame **data** που περιέχει όλα μας τα δεδομένα. Η υλοποίηση της συνάρτησης είναι ακριβώς η ίδια με το Lasso, με τη μόνη διαφορά ότι πλέον χρησιμοποιούμε τη συνάρτηση **Ridge( )** αντί για **Lasso( )** όπου είναι απαραίτητο. Η βέλτιστη τιμή της υπερπαραμέτρου **alpha** προκύπτει επιπλέον ότι είναι πάλι 0.01 όπως και στη προηγούμενη περίπτωση.

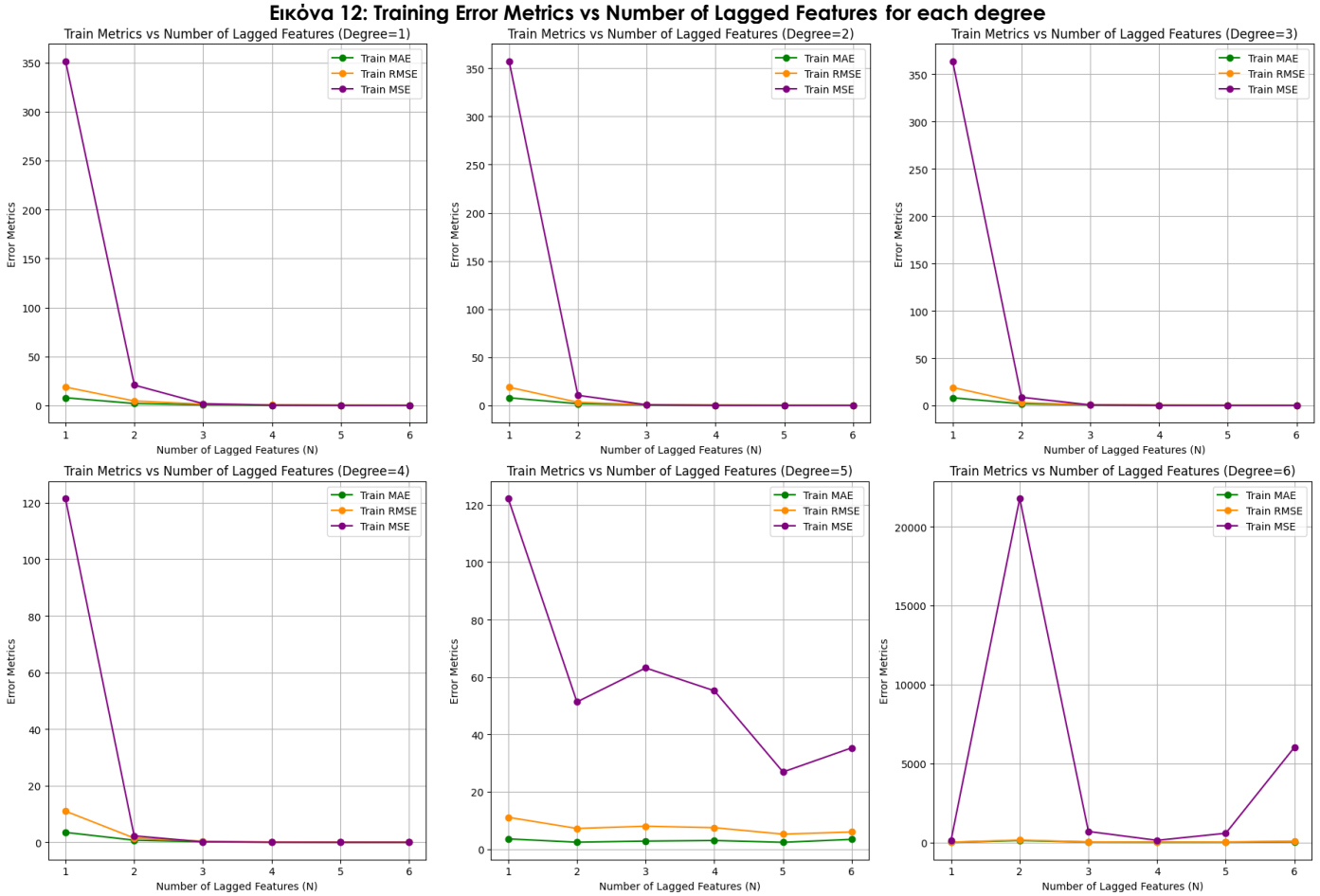
Πάλι σχεδιάζουμε ακριβώς όπως πριν τις γραφικές που προκύπτουν για lagged features από 1 έως 6 για όλους τους βαθμούς πολυωνύμου από 1 έως 6.

Εικόνα 11: Validation Error Metrics vs Number of Lagged Features for each degree



Από την Εικόνα 11 όσον αφορά το validation set συμπεραίνουμε ότι **το καλύτερο trade – off ανάμεσα σε πολυπλοκότητα και απόδοση για το μοντέλο προκύπτει για βαθμό πολυωνύμου 4 και για πλήθος lagged features ξανά από 3 – 4.** Βλέπουμε ότι το overfitting είναι πολύ πιο ξεκάθαρο μόλις ξεπεράσουμε τον βαθμό πολυωνύμου ίσο με 4, καθώς το σφάλμα «εκτοξεύεται» μόλις ξεπεράσουμε τα 4 lagged features αντί να τείνει να σταθεροποιηθεί σε κάποια τιμή. Επομένως, στη περίπτωση του Ridge η υπερβολική αύξηση του βαθμού του πολυωνύμου φαίνεται να είναι καταστροφική και η υπερπροσαρμογή είναι αναπόφευκτη, καθώς εισάγεται πάρα πολλή θορυβώδης πληροφορία με την αύξηση του πλήθους των συντελεστών του πολυωνύμου. Επιπλέον, είναι εμφανές **ότι με Κανονικοποίηση Ridge έχουμε πολύ καλύτερη απόδοση του μοντέλου σε σχέση με τη Lasso**, καθώς για βαθμούς μικρότερους ή ίσους του 4 το σφάλμα πάλι παρουσιάζει μια ξαφνική απότομη πτώση (elbow) και τείνει να σταθεροποιηθεί στο μηδέν όσο αυξάνονται τα lags. **Άρα, θα πρέπει να επιλέξουμε για το μοντέλο μας N = 3 και degree = 4.**

Εικόνα 12: Training Error Metrics vs Number of Lagged Features for each degree



Με βάση και τα σφάλματα του training set, βλέπουμε ότι όπως και στη Γραμμική Παλινδρόμηση, έτσι και τώρα οι γραφικές παραστάσεις παρουσιάζουν σημαντική ομοιότητα. Συνεπώς, το μοντέλο παρουσιάζει την ίδια συμπεριφορά τόσο στα δεδομένα που εκπαιδεύτηκε όσο και στα δεδομένα που χρησιμοποιήθηκαν για την επικύρωση. Άρα, το μοντέλο είναι **καλά ρυθμισμένο** χωρίς να παρουσιάζει σημαντικά ζητήματα overfitting ή underfitting.

Άρα, οι **υπερπαράμετρο**ι που θα χρησιμοποιηθούν πέραν το βαθμού του πολυωνύμου είναι ακριβώς οι ίδιες και επιλέγονται ακριβώς για τους ίδιους λόγους:

1. **degree** = 4
2. **alpha** = 0.01
3. **max_iter** = 10,000

Προχωράμε τώρα στη πρόβλεψη για τα νέα δεδομένα με το μοντέλο που επιλέξαμε. Οι συναρτήσεις για τη πρόβλεψη και για τον υπολογισμό της εξίσωσης του μοντέλου είναι ακριβώς οι ίδιες με πριν.

Δεδομένου ότι έχουμε γνωστά δεδομένα μέχρι τις 14/11/2024 η πρόβλεψη για την επόμενη μέρα (15/11/2024) από το μοντέλο αν τρέξουμε τους κώδικες φαίνεται παρακάτω:

```
Predicted close price for 2024-11-15: 321.88 $
```

Τέλος, οι **παράμετροι** του μοντέλου που χρησιμοποιήσαμε τελικά (δηλαδή οι συντελεστές) αποτυπώνονται στη παρακάτω εξίσωση:

Polynomial Equation: Close_t = **0.0037626617720434297** + (**0.0034151520272029403**) * close_t-1 + (**2.654595485833697**) * close_t-2 + (**-2.333589708577144**) * close_t-3 + (**0.6787751259613525**) * close_t-1^2 + (**-0.10253051571458767**) * close_t-2^2 + (**0.38664137134689214**) * close_t-3^2 + (**-0.18036209550322235**) * close_t-1^3 + (**-0.3628580054083627**) * close_t-2^3 + (**0.33648484207942775**) * close_t-3^3 + (**-0.07737360477845528**) * close_t-1^4 + (**0.0007175426753460315**) * close_t-2^4 + (**-0.004278375800224878**) * close_t-3^4

4 Συμπεράσματα

Στο πίνακα πιο κάτω βλέπουμε τις προβλέψεις που έκαναν τα 3 μοντέλα για τη τιμή κλεισίματος της μετοχής, καθώς και τη πραγματική τιμή κλεισίματος για τις 15/11/2024:

Linear Regreesion	Polynomial Regression w/ Lasso	Polynomial Regression w/ Ridge	Actual Closing Price
321.92 $	322.96 $	321.88 $	320.72 $

Το μοντέλο **Polynomial Regression w/ Ridge** έχει την πιο ακριβή πρόβλεψη (με απόκλιση 1.16 $), μαζί με το **Linear Regression** (με απόκλιση 1.2 $). Η **Polynomial Regression w/ Lasso** δίνει την πιο απομακρυσμένη πρόβλεψη (με απόκλιση 2.24 $).

Γενικά, είναι αναμενόμενο η συμπεριφορά του Polynomial Regression με Κανονικοποίηση Ridge να είναι σχεδόν ίδια με το Linear Regression για διάφορους λόγους. Αρχικά, από την προεπεξεργασία και τη στατιστική ανάλυση της χρονοσειράς παρατηρήθηκε ότι οι τιμές κλεισίματος έχουν **ισχυρή βραχυπρόθεσμη εξάρτηση**. Αυτό υποδηλώνει ότι η σχέση μεταξύ των χαρακτηριστικών (lagged features) και της τρέχουσας τιμής του στόχου (Close price) είναι **σχεδόν γραμμική**, γεγονός που επαληθεύει ότι η γραμμική παλινδρόμηση μπορεί να είναι εξίσου αποτελεσματική με τη Ridge. Επίσης, η χρονοσειρά σημειώσαμε ότι έχει μια **κυρίαρχη τάση** και όχι έντονες εποχιακές διακυμάνσεις με αποτέλεσμα οι μακροχρόνιες τάσεις να είναι ομαλές και να μην απαιτούν σύνθετες διορθώσεις από το Ridge.

Εφόσον λοιπόν η Linear Regression θα λειτουργεί ήδη αρκετά καλά, τότε η Ridge Regression δεν θα βελτιώσει σημαντικά την πρόβλεψη επειδή δεν έχει να «διορθώσει» υπερβολικά μεγάλες τιμές συντελεστών.

Η Lasso Regression από την άλλη λόγω της ποινής που εισάγει μπορεί να «μηδενίσει» συντελεστές που σχετίζονται με ορισμένα από τα lagged features, ακόμα κι αν αυτά έχουν μικρή αλλά υπαρκτή συνεισφορά στο μοντέλο. Μηδενίζοντας έτσι ορισμένα lagged features, το μοντέλο μπορεί να χάσει κρίσιμες πληροφορίες για την τάση ή τη μεταβλητότητα της χρονοσειράς, καθώς αφαιρώντας κάποια γνωρίσματα αγνοεί τη συνδυαστική τους συμβολή.

Καταλήγουμε δηλαδή ότι τα προηγούμενα δύο μοντέλα μπορούν να χειριστούν αποδοτικότερα την υψηλή συσχέτιση ανάμεσα στα lagged features και γι' αυτό εν τέλει υπετερούν.

5 Παράρτημα

Παρατίθεται το Github Repository με το συνολικό κώδικα, σχόλια και επεξηγήσεις στο αντίστοιχο Jupyter Notebook:
https://github.com/miltiadiss/Decision-Theory/blob/main/TSLA_stock_price_prediction.ipynb

Επίσης, παρατίθενται οι πηγές και η βιβλιογραφία που χρησιμοποιήθηκαν κατά τη μελέτη για την υλοποίηση της εργασίας:
https://eclass.upatras.gr/courses/CEID1039/
https://towardsdatascience.com/gaussian-smoothing-in-time-series-data-c6801f8a4dc3
https://scikit-learn.org/1.5/modules/generated/sklearn.linear_model.LinearRegression.html
https://scikit-learn.org/1.5/auto_examples/linear_model/plot_ols.html#sphx-glr-auto-examples-linear-model-plot-ols-py
https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Ridge.html