Πανεπιστήμιο Πειραιώς – Τμήμα Πληροφορικής

Πρόγραμμα Μεταπτυχιακών Σπουδών

«Προηγμένα Συστήματα Πληροφορικής»

Μεταπτυχιακή Διατριβή

Τίτλος Διατριβής	**ΠΡΟΒΛΕΨΗ ΧΡΟΝΟΣΕΙΡΩΝ ΑΕΡΙΑΣ ΡΥΠΑΝΣΗΣ ΚΑΙ ΕΝΕΡΓΕΙΑΚΗΣ ΚΑΤΑΝΑΛΩΣΗΣ ΜΕ ΤΗ ΧΡΗΣΗ ARIMA ΚΑΙ LSTM ΜΟΝΤΕΛΩΝ** **TIME SERIES PREDICTION OF POLLUTION AND ENERGY CONSUMPTION DATA USING ARIMA AND LSTM MODELS**
Ονοματεπώνυμο Φοιτητή	**ΜΑΡΙΑ ΦΟΥΣΤΑΛΙΕΡΑΚΗ**
Πατρώνυμο	**ΓΕΩΡΓΙΟΣ**
Αριθμός Μητρώου	**ΜΠΣΠ17071**
Επιβλέπων	**ΑΓΓΕΛΟΣ ΠΙΚΡΑΚΗΣ, Επίκουρος Καθηγητής**

ΔΕΚΕΜΒΡΙΟΣ 2019

Τριμελής Εξεταστική Επιτροπή

Πικράκης Άγγελος	*Θεοδωρίδης Γιάννης*	*Πελέκης Νικόλαος*
Επίκουρος Καθηγητής	*Καθηγητής*	*Επίκουρος Καθηγητής*

ΠΕΡΙΛΗΨΗ

Τα τελευταία χρόνια ο τομέας της Μηχανικής Μάθησης έχει γνωρίσει ραγδαία ανάπτυξη για το λόγο ότι υπάρχουν πλέον συστήματα με πολύ μεγάλη υπολογιστική ισχύ που μπορούν να χρησιμοποιηθούν για την αξιοποίησή των δυνατοτήτων του. Το γεγονός αυτό έχει κεντρίσει το ενδιαφέρον των ερευνητών που προσπαθούν και έχουν καταφέρει να αναπτύξουν διάφορα μοντέλα σε μια ευρεία γκάμα από εφαρμογές.

Η συγκεκριμένη διπλωματική εργασία ασχολείται με ένα μεγάλο κεφάλαιο της μηχανικής μάθησης, αυτό των νευρωνικών δικτύων και συγκεκριμένα με την κατηγορία των LSTM ανατροφοδοτούμενων νευρωνικών δικτύων. Επιπλέον, κάνοντας χρήση μεθόδων ανάλυσης χρονοσειρών με την ανάπτυξη ARIMA μοντέλων, εξετάζουμε την απόδοσή τους συγκρίνοντάς τα μεταξύ τους. Για την μελέτη και την ανάπτυξή τους χρησιμοποιήθηκαν τρία διαφορετικά σύνολα δεδομένων. Το πρώτο αφορά τη πρόβλεψη ενεργειακής κατανάλωσης από ένα νοικοκυριό, το δεύτερο την πρόβλεψη της ποιότητας του αέρα για μια συγκεκριμένη περιοχή και το τρίτο αφορά επίσης την πρόβλεψη της ποιότητας αέρα, αλλά για πολλές φυσικές τοποθεσίες. Για την χρήση τους απαιτήθηκε η προετοιμασία των δεδομένων, τόσο με τη διαχείριση των κενών τιμών όσο και με το διαχωρισμό τους σε δεδομένα εκπαίδευσης και σε δεδομένα δοκιμών. Δοκιμάστηκαν διάφορες παραλλαγές στα βασικά μοντέλα ARIMA Και LSTM και καταλήξαμε στα βέλτιστα από αυτά, ενώ οι μετρικές που χρησιμοποιήθηκαν για την απόδοσή τους ήταν το RMSE και MAE.

Από την ανάλυση αυτή δείχνει τις δυνατότητες επίλυσης μεγάλων και πολύπλοκων προβλημάτων με τη χρήση τόσο των αναδρομικών νευρωνικών δικτύων όσο και με μεθόδους ανάλυσης χρονοσειρών.

ABSTRACT

There has been great development in the field of Machine Learning in recent years, mainly since the processing power of the machines that make use of its capabilities has greatly increased. This has sparked the interest of developers and analysts who are trying and have managed to develop different models based on machine learning techniques that apply to a wide range of applications.

This thesis deals with a large chapter of machine learning science, that of neural networks and in particular the LSTM recurrent neural networks. In addition, by using timeseries analysis methods with the development of ARIMA models, we test their performance by comparing each one of them. Three different datasets were used for this study and the model's development. The first concerns the prediction of the energy consumption by a household, the second concerns the air quality forecasting for a particular area, and the third also concerns the prediction of air quality but for many natural sites. Their use required the data preparation by managing NaN values and by separating them into training datasets and tests datasets. Several variants of the ARIMA and LSTM basic models were tested, and we reached to the best of them, while the measurements used for performance were RMSE and MAE metrics.

This analysis shows the capability of both recurrent neural networks and timeseries analysis methods on solving big and complex problems like the examined.

Περιεχόμενα

1. ΧΡΟΝΟΣΕΙΡΕΣ

1.1 Ορισμός

Τα δεδομένα μιας μεταβλητής που συλλέγονται με σταθερό χρόνο δειγματοληψίας, δηλαδή σε διαδοχικά και ίσα χρονικά διαστήματα, ονομάζονται χρονική σειρά ή χρονοσειρά (time series). [17] Έστω ότι συμβολίζουμε με Y_i τις τιμές των παρατηρήσεων και με X_i τις αντίστοιχες χρονικές στιγμές (έτη, μήνες, μέρες, ώρα, δευτερόλεπτα, κτλ.). Τότε δημιουργούμε ζεύγη της μορφής $M(X_i, Y_i)$, τα οποία μπορούν να παρασταθούν στο καρτεσιανό σύστημα αξόνων. Με την ένωση αυτών των σημείων δημιουργείται ένα χρονοδιάγραμμα το οποίο αποδίδει την γενική εικόνα της εξέλιξης του υπό έρευνα φαινομένου ή χαρακτηριστικού. Παραδείγματα τέτοιων χρονοσειρών είναι οι ημερήσιες αφίξεις επιβατών σε ένα λιμάνι, ο αριθμός πελατών σε πολυκατάστημα κατά μία χρονική στιγμή t, η ημερήσια κατανάλωση ρεύματος, κτλ.

Οι χρονοσειρές χωρίζονται σε δύο μεγάλες κατηγορίες, τις συνεχής χρονοσειρές (continuous) και τις διακριτές (discrete). Συνεχής θεωρείται μια χρονοσειρά στην οποία η τιμή του φαινομένου παρατηρείται συνεχώς. Για παράδειγμα κατά την καταγραφή των σεισμών. Διακριτή είναι αυτή όπου η τιμή του φαινομένου παρατηρείται σε συγκεκριμένα χρονικά διαστήματα, για παράδειγμα οι μηνιαίες πωλήσεις ενός προϊόντος. Αυτά τα χρονικά διαστήματα ή αλλιώς δειγματοληψία αλλάζουν ανάλογα την εφαρμογή και είναι στη κρίση του εκάστοτε αναλυτή τι τιμή θα χρησιμοποιήσει. Μια άλλη κατηγοριοποίηση βασίζεται στη ποσότητα των μεταβλητών που είναι προς ανάλυση. Με βάση αυτή έχουμε δύο κατηγορίες τις μονοδιάστατες (univariate) και τις πολυδιάστατες (multivariate). [17]

1.2 Συνιστώσες χρονοσειρών

Μια χρονοσειρά συνήθως αποτελείται από τέσσερις συνιστώσες, οι οποίες μπορούν να διαχωριστούν από τα δεδομένα για τη λήψη αποφάσεων. [17] Οι συνιστώσες αυτές είναι:

- *η τάση (Trend)*
  Μια χρονοσειρά παρουσιάζει Τάση όταν για μια μεγάλη περίοδο οι τιμές της τείνουν να αυξάνονται ή να μειώνονται.

- *η εποχικότητα (Seasonality)*
  Εποχικότητα παρατηρείται όταν μια χρονοσειρά επηρεάζεται από εποχιακούς παράγοντες όπως η εποχή του χρόνου, η ημέρα της εβδομάδας ή ακόμα και από παραδόσεις που ισχύουν σε μία χώρα. Η εποχικότητα είναι πάντα σταθερή και γνωστής συχνότητας.

- *οι Κυκλικές κινήσεις (Cyclic Movements)*
  Κυκλικές κινήσεις εμφανίζονται όταν τα δεδομένα παρουσιάζουν αυξήσεις και πτώσεις που δεν έχουν σταθερή συχνότητα. Αυτές οι διακυμάνσεις παρατηρούνται κυρίως στα οικονομικά δεδομένα και η διάρκεια αυτών είναι συνήθως τουλάχιστον 2 έτη.

- *οι Ακανόνιστες Διακυμάνσεις (Irregular Fluctuations)*
  Οι Ακανόνιστες διακυμάνσεις είναι ξαφνικές αλλαγές που συμβαίνουν σε μια χρονοσειρά που είναι αδύνατο να επαναληφθούν. Είναι συστατικά μιας χρονοσειράς που δεν μπορούν να εξηγηθούν από τάσεις, εποχικότητες ή κυκλικές κινήσεις. Παράδειγμα αυτών των διακυμάνσεων είναι οι απεργίες.

Είναι πιθανό σε μια χρονοσειρά να μην υπάρχουν όλες οι συνιστώσες αλλά μόνο κάποιες από αυτές. Η εξέταση των στοιχείων αυτών γίνεται με τη χρήση κάποιου μαθηματικού υποδείγματος που φανερώνει τον τρόπο με τον οποίο οι παρατηρήσεις της χρονοσειράς προσδιορίζονται από τις συνιστώσες αυτές. Τα υποδείγματα αυτά είναι το προσθετικό μοντέλο (additive model) και το πολλαπλασιαστικό μοντέλο (multiplicative model). [17]

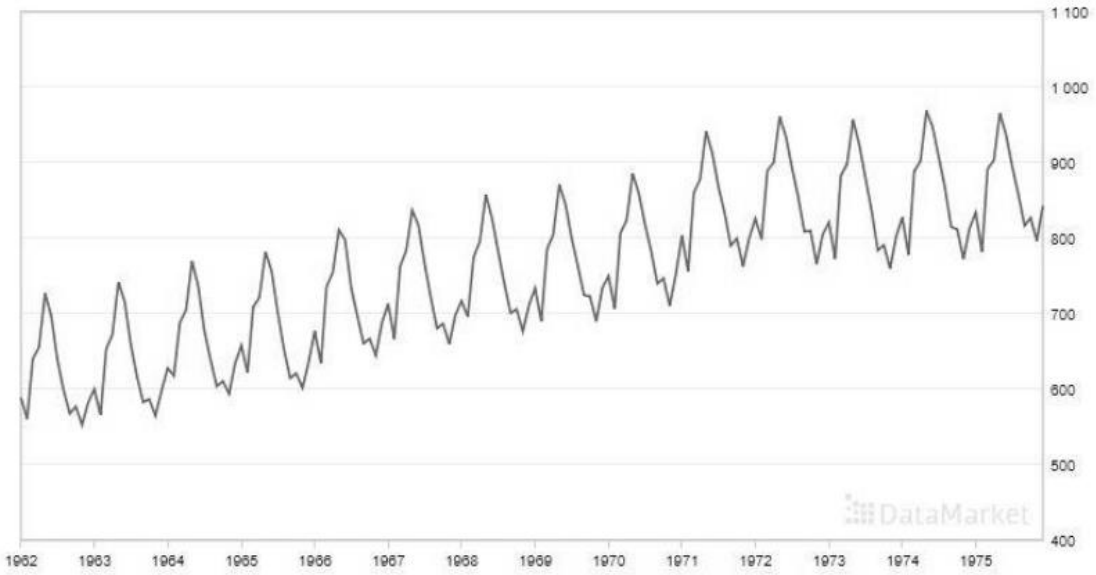
1.2.1 Το προσθετικό μοντέλο

Το προσθετικό μοντέλο είναι κατάλληλο αν το μέγεθος των εποχιακών διακυμάνσεων ή η μεταβολή γύρω από τη τάση και τις κυκλικές κινήσεις δεν διαφέρει ανάλογα το επίπεδο της χρονοσειράς. [26]

Η σχέση που συνδέει τις συνιστώσες στο προσθετικό μοντέλο γράφεται ως εξής [17]:

$$Y_t = T_t + S_t + C_t + I_t$$

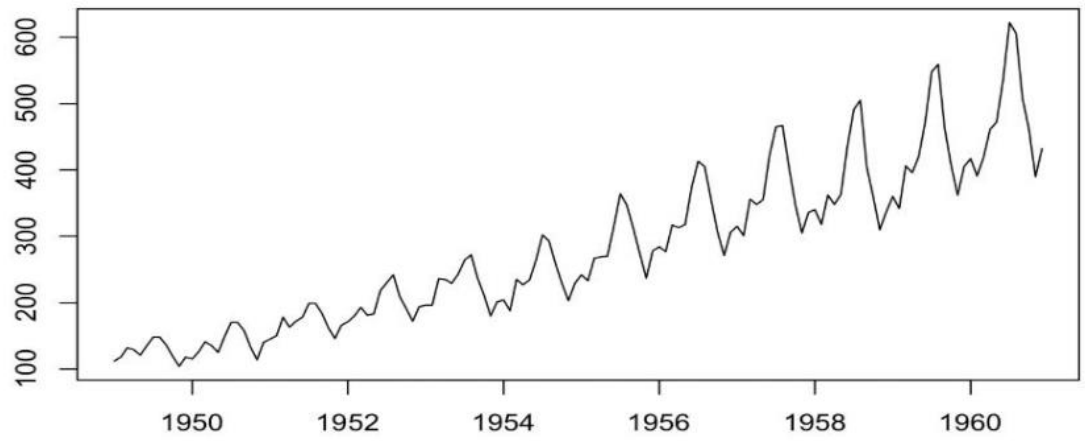
Ένα τέτοιο μοντέλο είναι κατάλληλο για τη χρονοσειρά της παρακάτω εικόνας.



Εικόνα 1: Παράδειγμα χρονοσειράς για προσθετικό μοντέλο (Μηνιαία παραγωγή γάλακτος Ιανουάριος 1962 – Δεκέμβριος 1975) [26]

1.2.2 Το πολλαπλασιαστικό μοντέλο

Όταν η μεταβολή του εποχικού προτύπου ή η μεταβολή γύρω από τη τάση και τις κυκλικές κινήσεις φαίνεται να είναι ανάλογη με το επίπεδο των χρονοσειρών, τότε καταλληλότερο είναι ένα πολλαπλασιαστικό μοντέλο. [17] Στην παρακάτω εικόνα φαίνεται μια χρονοσειρά κατάλληλη για το συγκεκριμένο μοντέλο. [27]



Εικόνα 2: Παράδειγμα χρονοσειράς για πολλαπλασιαστικό μοντέλο [27]

Η σχέση που συνδέει τις συνιστώσες στο πολλαπλασιαστικό μοντέλο γράφεται ως εξής:

$$Y_t = T_t * S_t * C_t * I_t$$

όπου Y_t η πραγματική τιμή της χρονοσειράς για περίοδο t, T_t η τάση, S_t η εποχικότητα, C_t η κυκλικότητα και I_t οι ακανόνιστες διακυμάνσεις. [17]

1.3 Αιτιοκρατία και Στοχαστικότητα

Όλες οι χρονολογικές σειρές από πραγματικά μεγέθη παρουσιάζουν θόρυβο, αυτές είναι και οι στοχαστικές χρονοσειρές. Ο εντοπισμός αυτού του αιτιοκρατικού μέρους της σειράς είναι πρόκληση να πραγματοποιηθεί όταν αυτό εμπεριέχεται μέσα στον θόρυβο και τότε θεωρούμε πως το σύστημα είναι στοχαστικό.[17]

1.4 Στασιμότητα

Στάσιμη (stationary) θεωρείται μια χρονοσειρά της οποίας ο μέσος και οι διακυμάνσεις παραμένουν σταθερές στον χρόνο. Η εμφάνιση τάσης ή περιοδικότητας σε μια χρονοσειρά υποδηλώνει τη μη στασιμότητά της. Πριν την ανάλυση θα πρέπει να ουδετεροποιηθεί η όποια επίδραση των δυο αυτών συνιστωσών σε μια χρονοσειρά. Η μη στασιμότητα αποτελεί σοβαρό πρόβλημα στην ανάλυση χρονοσειρών και ιδιαίτερα στη πρόβλεψη μελλοντικών τιμών. [17]

2. ΓΡΑΜΜΙΚΑ ΜΟΝΤΕΛΑ ΠΡΟΒΛΕΨΗΣ

Για την πρόβλεψη χρονοσειρών χρησιμοποιούνται ευρέως δύο μοντέλα, το Autoregressive – AR (αυτοπαλίνδρομο) και το Moving Average - MA (κινητού μέσου όρου) [18]. Συνδυάζοντας αυτά τα δύο προκύπτει το Autoregressive Moving Average - ARMA και το Autoregressive Integrated Moving Average - ARIMA. Για πρόβλεψη χρονοσειρών που παρουσιάζουν εποχικότητα χρησιμοποιείται μια παραλλαγή του ARIMA, το Seasonal Autoregressive Integrated Moving Average - SARIMA. Το μοντέλο ARIMA και οι παραλλαγές του βασίζονται στην αρχή Box-Jenkins και είναι γνωστά ως μοντέλα Box-Jenkins. [19] Λόγω της ευκολίας στην κατανόηση της διαδικασίας και στην ερμηνεία των αποτελεσμάτων τους, η χρήση αυτών των μοντέλων είναι συνηθισμένη. Ωστόσο ο γραμμικός χαρακτήρας των παραπάνω μοντέλων περιορίζει τη πρόβλεψη μη-γραμμικών προβλημάτων, τα οποία αποτελούν και την πλειοψηφία των χρονοσειρών. Η μη-γραμμική δομή των Νευρωνικών Δικτύων έδωσε τη λύση, καθώς, δίνει τη δυνατότητα να αντιλαμβάνονται τόσο τις γραμμικές όσο και τις μη-γραμμικές συσχετίσεις των δεδομένων. Έτσι ξεπεράστηκαν οι περιορισμοί αυτοί, με την ανάπτυξη της Τεχνητής Νοημοσύνης (Artificial Intelligence - AI), και την εφαρμογή όλο και περισσότερο των Νευρωνικών Δικτύων στη πρόβλεψη χρονοσειρών. [18]

2.1 Autoregressive (AR) Model

Σε ένα autoregressive μοντέλο, η μεταβλητή εξόδου y_t εξαρτάται γραμμικά από τις προηγούμενες τιμές της (y_{t-1}, ... , y_{t-p}) και κάποιου λευκού θορύβου ε_t. [18] Εξορισμού ένα υπόδειγμα $\{y_t\}$ λέγεται ότι είναι autoregressive p τάξης συμβολίζεται με AR(p),εάν το y_t περιγράφεται από την εξής σχέση:

$$y_t = a_1 y_{t-1} + a_2 y_{t-2} + \cdots + a_p y_{t-p} + \varepsilon_t$$

όπου

ε_t: ο λευκός θόρυβος (white noise) με μηδενικό μέσο (zero mean) και σταθερή διακύμανση σ_z^2 .

a_1, ..., a_p: οι παράμετροι του υποδείγματος

Η τάξη p του υποδείγματος καθορίζει τον αριθμό των προηγούμενων παρατηρήσεων που χρησιμοποιούνται για την πρόβλεψη της τρέχουσας τιμής.

Υπενθυμίζεται ότι η διακύμανση μετράει κατά πόσο απέχει κάθε τιμή από την μέση τιμή. Ο υπολογισμός της γίνεται αφαιρώντας τον κάθε αριθμό στα δεδομένα μας με την μέση τιμή και στην συνέχεια τετραγωνίζοντας το αποτέλεσμα ώστε να είναι θετικό και στην συνέχεια διαιρούμε το άθροισμα των τετραγώνων με τον αριθμό των τιμών των δεδομένων μας.

$$\sigma^2 = \frac{\sum(X-\mu)^2}{N}$$

όπου N το πλήθος των δεδομένων, X η εκάστοτε παρατήρηση και μ η μέση τιμή.

Επιπρόσθετα, αν θεωρήσουμε ως τυχαία τα διαδοχικά στοιχεία μιας χρονοσειράς, τότε αυτή η χρονοσειρά λέγεται ότι αποτελείται από ανεξάρτητες τυχαίες μεταβλητές με ίδια κατανομή, για $x_t, x_{t+1}, x_{t+\tau}$, για τ > 1. Μια iid χρονολογική σειρά είναι εντελώς τυχαία και δεν περιέχει αυτό-συσχετίσεις, γραμμικές ή μη-γραμμικές. Μια τέτοια χρονολογική σειρά, iid, ονομάζεται και λευκός θόρυβος (white noise) και η κατανομή της συμβολίζεται ως WN(0,σ_ε^2), με μέση τιμή 0 και διασπορά σ_ε^2, αν αυτός ο θόρυβος επίσης, ακολουθεί γκαουσιανή κατανομή, τότε λέγεται γκαουσιανός θόρυβος. [18]

2.2 Moving Average (MA) Model

Υποθέτοντας ότι $\{\varepsilon_t\}$ είναι μια καθαρά τυχαία διαδικασία με μηδενικό μέσο και διακύμανση σ_z^2 τότε ένα υπόδειγμα $\{y_y\}$ λέγεται κινητού μέσου τάξης q MA(q) εάν το y_t περιγράφεται από την εξής σχέση [18]:

$$y_t = \varepsilon_t + \beta_1\varepsilon_{t-1} + \beta_2\varepsilon_{t-2} + \cdots + \beta_p\varepsilon_{t-p}$$

Όπου,

ε_t: ο λευκός θόρυβος (white noise)

$\beta_1, \ldots, \beta_p$: οι παράμετροι του υποδείγματος

Στη διαδικασία κινητού μέσου το υπόδειγμα $\{y_t\}$ θεωρείται ότι δημιουργείται ως ένας σταθμικός μέσος (weighted average) τυχαίων σφαλμάτων των q προηγούμενων περιόδων.

2.3 Auto Regressive Moving Average (ARMA) Model

Το μοντέλο ARMA είναι ένα από τα πιο ευρέως χρησιμοποιούμενα μοντέλα καθώς συνδυάζει τα πλεονεκτήματα των δυο προαναφερθέντων. [18] Το συγκεκριμένο υπόδειγμα τάξης (p,q) ορίζεται ως εξής:

$$y_t = a_1y_{t-1} + a_2y_{t-2} + \cdots + a_py_{t-p} + \varepsilon_t + \beta_1\varepsilon_{t-1} + \beta_2\varepsilon_{t-2} + \cdots + \beta_p\varepsilon_{t-p}$$

Όπου,

ε_t: ο λευκός θόρυβος (white noise)

$a_1, \ldots, a_p$: οι παράμετροι του υποδείγματος για το AR

$\beta_1, \ldots, \beta_p$: οι παράμετροι του υποδείγματος για το MA

2.4 Auto Regressive Integrated Moving Average (ARIMA) Model

Τα μοντέλα που παρουσιάστηκαν προηγουμένως, AR – AM – ARMA, χρησιμοποιούνται σε στάσιμες διαδικασίες. Δηλαδή, ο μέσος και η διακύμανση δεν εξαρτώνται από τον χρόνο t αλλά παραμένουν σταθερά. Στην πράξη οι περισσότερες χρονοσειρές είναι μη-στάσιμες οπότε για να εφαρμοστεί ένα στάσιμο υπόδειγμα θα πρέπει να αφαιρεθούν τα μη-στάσιμα χαρακτηριστικά. [19] Μια λύση σε αυτό παρουσιάστηκε με το μοντέλο ARIMA τάξεως (p,d,q). Η ικανότητα του ARIMA μοντέλου να ανταπεξέρχεται σε μη-στάσιμες διαδικασίες, το καθιστά μια από τις πιο χρησιμοποιούμενες προσεγγίσεις στη πρόβλεψη

χρονοσειρών. Υπερνικά αυτόν τον περιορισμό με τη χρήση διαφορών και αυτό επιτυγχάνεται αφαιρώντας την παρατήρηση της τρέχουσας περιόδου από την προηγούμενη. Η διαφόριση αυτή μπορεί να είναι διαφόρων τάξεων. [18] Για παράδειγμα η πρώτη τάξης διαφόριση πραγματοποιείται αντικαθιστώντας το y_t με την εξής σχέση:

$$y'_t = y_t - y_{t-1}$$

Η γενική μορφή του υποδείγματος ARIMA (p,d,q) περιγράφεται από τη παρακάτω σχέση:

$$y'_t = a_1 y'_{t-1} + a_2 y'_{t-2} + \cdots + a_p y'_{t-p} + \varepsilon_t + \beta_1 \varepsilon_{t-1} + \beta_2 \varepsilon_{t-2} + \cdots + \beta_p \varepsilon_{t-p}$$

Όπου,

p: οι παράμετροι της αυτοπαλίνδρομης διαδικασίας

d: ο αριθμός των διαφορών προκειμένου η χρονοσειρά να γίνει στάσιμη

q: οι παράμετροι της διαδικασίας του κινητού μέσου

2.5 Seasonal ARIMA (SARIMA) Model

Το μοντέλο SARIMA είναι μια επέκταση του ARIMA και χρησιμοποιείται για δεδομένα όπου παρουσιάζουν περιοδικότητα. [18] Το SARIMA διατυπώνεται ως SARIMA(p,d,q)(P,D,Q)m όπου P το πλήθος των εποχικών autoregressive όρων, D το πλήθος των εποχικών διαφορών και Q το πλήθος των εποχικών όρων κινητού μέσου, m ο αριθμός των χρονικών βημάτων στη χρονική περίοδο μια εποχής. [19]

3. ΜΗΧΑΝΙΚΗ ΜΑΘΗΣΗ

Ο Arthur Samuel ορίζει ως μηχανική μάθηση «Το πεδίο μελέτης που δίνει στους υπολογιστές την ικανότητα να μαθαίνουν, χωρίς να έχουν ρητά προγραμματιστεί». [28] Η μηχανική μάθηση επικεντρώνεται στην ανάπτυξη αλγορίθμων που μπορούν να έχουν πρόσβαση σε δεδομένα και να τα χρησιμοποιούν για να μάθουν και να βελτιώνονται. Η μηχανική μάθηση χωρίζεται σε τρεις κύριες κατηγορίες:

- *Επιβλεπόμενη Μάθηση (Supervised Learning).*
  Ο αλγόριθμος εκπαιδεύεται σε ένα σύνολο παραδειγμάτων με ζευγάρια εισόδων και επιθυμητών εξόδων. Μερικοί από τους αλγορίθμους που χρησιμοποιούνται είναι:

- Μηχανές Διανυσμάτων Υποστήριξης (Support Vector Machines - SVM)
- Γραμμική Παλινδρόμηση (Linear Regression)
- Λογιστική Παλινδρόμηση (Logistic Regression)
- Δέντρα Αποφάσεων (Decision Trees)
- Νευρωνικά Δίκτυα (Neural Networks)

➢ *Μη Επιβλεπόμενη Μάθηση (Unsupervised Learning)*.
Ο αλγόριθμος εκπαιδεύεται χρησιμοποιώντας πληροφορίες που δεν είναι ούτε ταξινομημένες αλλά ούτε επισημασμένες και επιτρέπουν το αλγόριθμο να ενεργεί χωρίς καθοδήγηση. Αλγόριθμοι για αυτή τη κατηγορία είναι:
- Συσταδοποίηση k-means
- Ιεραρχική Συσταδοποίηση (Hierarchical Clustering)
- Ανάλυση Κύριων Συνιστωσών (Principle Component Analysis - PCA)

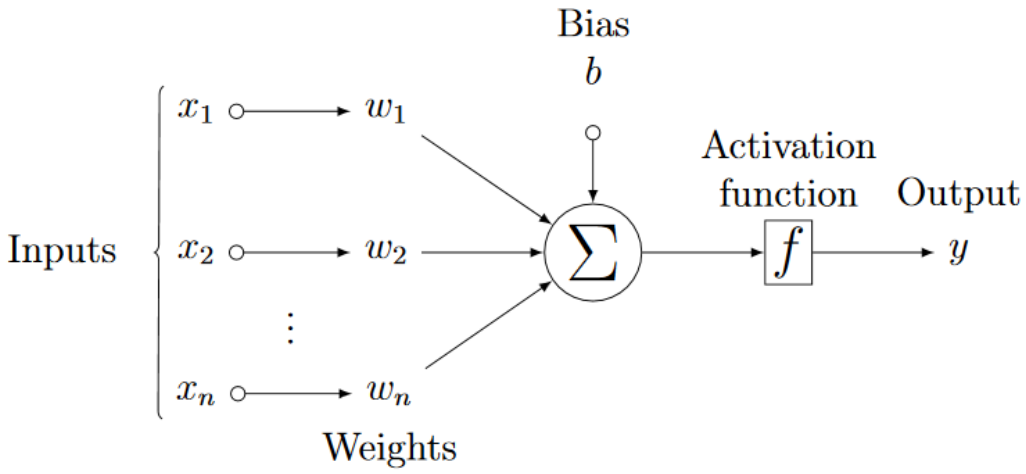
➢ *Ενισχυτική Μάθηση (Reinforcement Learning)*.
Ο αλγόριθμος μαθαίνει πως να συμπεριφέρεται σε ένα περιβάλλον εκτελώντας ενέργειες και βλέποντας αποτελέσματα.

4. ΝΕΥΡΩΝΙΚΑ ΔΙΚΤΥΑ

Τα τεχνητά νευρωνικά δίκτυα (Artificial Neural Networks - ANN) ή σε συντομία Νευρωνικά Δίκτυα είναι από τα κύρια εργαλεία στη μηχανική μάθηση. [30] Είναι υπολογιστικά συστήματα εμπνευσμένα από τα βιολογικά νευρωνικά δίκτυα που αποτελούν τον εγκέφαλο. Παρόλο που τα ΤΝΔ ανακαλύφθηκαν από τη δεκαετία του 1940, μόνο κατά τα τελευταία χρόνια έπαιξαν σημαντικό ρόλο στην τεχνητή νοημοσύνη. Οι κύριοι λόγοι για τους οποίους άργησαν να χρησιμοποιούνται είναι η έλλειψη υπολογιστικής ισχύς και η έλλειψη δεδομένων για την εκπαίδευση ενός μοντέλου.

4.1 Perceptron

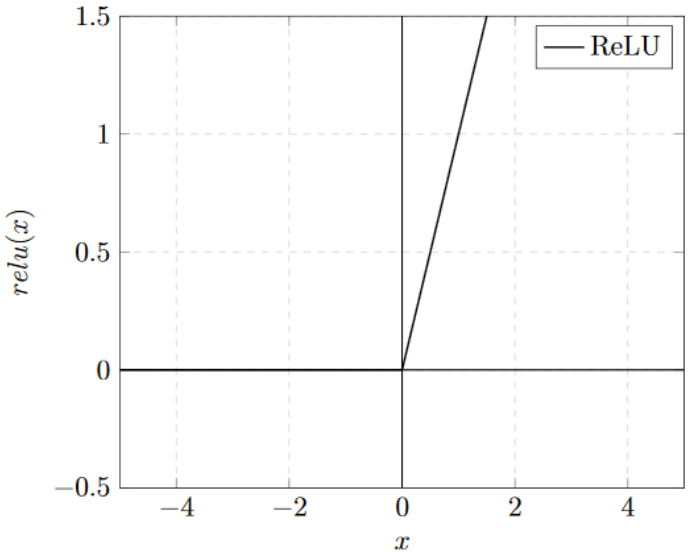
Η βασική μονάδα υπολογισμού σε ένα ΤΝΔ είναι ο νευρώνας (Neuron) , συχνά ονομαζόμενος και ως κόμβος. [30] Λαμβάνει εισόδους από άλλους κόμβους ή από μια εξωτερική πηγή και υπολογίζει μια έξοδο. Κάθε είσοδος πολλαπλασιάζεται με το αντίστοιχο βάρος (Weight) και υπολογίζεται τα ολικό άθροισμα των γινομένων. Ο κόμβος εφαρμόζει μια συνάρτηση ενεργοποίησης σε αυτό το άθροισμα και υπολογίζεται η έξοδος του νευρώνα. Παρακάτω φαίνεται η αναπαράσταση ενός νευρώνα με τρεις εισόδους καθώς και η εξίσωση της εξόδου.



Εικόνα 3: Αναπαράσταση ενός νευρώνα με τρεις εισόδους [30]

$$y = f(\sum_{i=1}^{n} w_i * x_i + b_i)$$

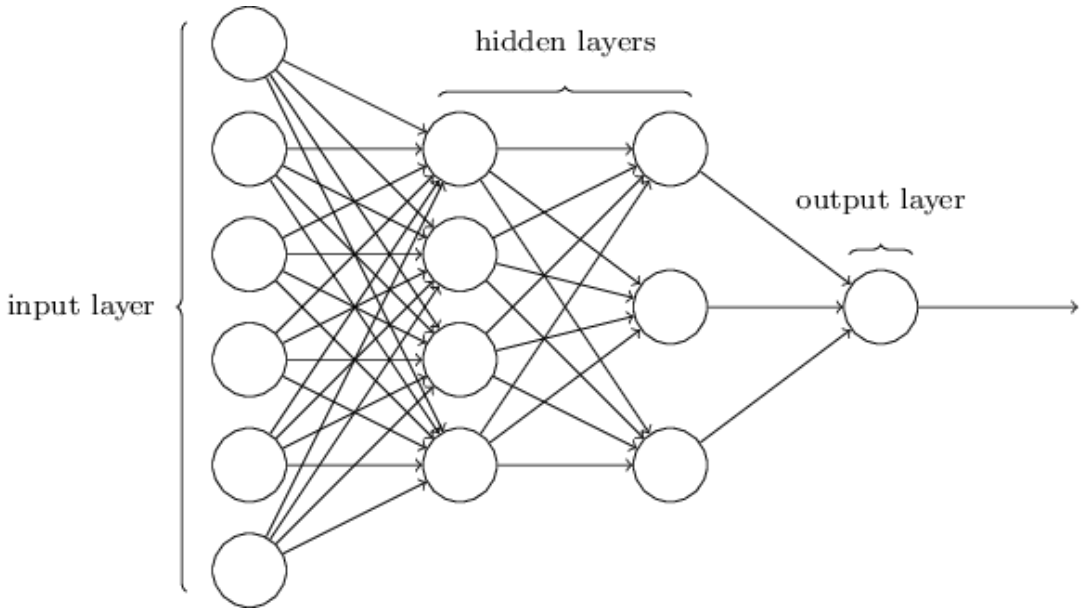
Το παραπάνω δίκτυο παίρνει ως εισόδους τα X_1, X_2 έως X_n που έχουν για βάρη τα W_1, W_2 έως W_n αντίστοιχα. Επιπλέον υπάρχει ακόμα μια είσοδος 1 με βάρος b η οποία ονομάζεται πόλωση (bias). Η συνάρτηση f είναι μη γραμμική και ονομάζεται συνάρτηση ενεργοποίησης (activation function). Ο σκοπός της συνάρτησης ενεργοποίησης είναι να εισάγει μη γραμμικότητα στην έξοδο ενός νευρώνα. [30] Αυτό είναι σημαντικό καθώς σχεδόν όλα τα πραγματικά δεδομένα είναι μη γραμμικά. Παραδείγματα τέτοιων συναρτήσεων είναι η σιγμοειδής συνάρτηση, η υπερβολική εφαπτομένη και η ReLU την οποία και θα χρησιμοποιήσουμε στις εφαρμογές αυτής τη εργασίας. Η ReLu αποδίδει συχνά καλύτερα από άλλες συναρτήσεις ενεργοποίησης για κρυφά επίπεδα. Ο βασικός λόγος της αυξημένης απόδοσης οφείλεται στο γεγονός ότι η ReLU είναι μια γραμμική συνάρτηση μη κορεσμού. Ο κορεσμός είναι το μεγαλύτερο πρόβλημα των δυο προηγούμενων σιγμοειδών συναρτήσεων. Σε αντίθεση λοιπόν με την logistic ή tanh, η ReLU δεν κορέζεται στο -1, 0 ή 1. Οι πιο πρόσφατες έρευνες αναφέρουν ότι τα κρυμμένα επίπεδα του ΤΝΔ πρέπει να χρησιμοποιούν την ενεργοποίηση του ReLu.



Εικόνα 4: Γραφική αναπαράσταση της ReLu συνάρτησης [30]

$$f(x) = \max(0, x) = \begin{cases} 0, για\ x < 0 \\ x, για\ x \geq 0 \end{cases}$$

Η βασικότερη μορφή ενός νευρωνικού δικτύου είναι τα Πολυεπίπεδα Perceptrons (Multilayer Perceptrons - MLP). Οι νευρώνες στα MLP είναι οργανωμένοι σε επίπεδα (layers) και δεν υπάρχουν συνδέσεις μεταξύ νευρώνων του ιδίου επιπέδου. [29] Το πρώτο από αυτά τα επίπεδα ονομάζεται επίπεδο εισόδου (input layer) και χρησιμοποιείται για την εισαγωγή των δεδομένων. Τα στοιχεία αυτού του επιπέδου δεν αποτελούν νευρώνες καθώς δεν εκτελούν κάποιον υπολογισμό. Ακολουθούν ένα ή περισσότερα κρυφά επίπεδα (hidden layers) και τέλος υπάρχει το επίπεδο εξόδου (output layer). Στην παρακάτω εικόνα φαίνεται ένα παράδειγμα ενός MLP με 6 εισόδους, 2 κρυφά επίπεδα με 4 και 3 νευρώνες αντίστοιχα για το κάθε επίπεδο και ένα επίπεδο εξόδου.



Εικόνα 5: MLP με 6 εισόδους, 2 κρυφά επίπεδα με 4 και 3 νευρώνες αντίστοιχα για το κάθε επίπεδο και ένα επίπεδο εξόδου [29]

Ένα MLP νευρωνικό δίκτυο έχει τις εξής ιδιότητες:

- Feedforward αποκλειστικά, καμία ανάδραση.
- Πλήρως συνδεδεμένα επίπεδα, δηλαδή κάθε νευρώνας ενός επιπέδου συνδέεται με όλους τους νευρώνες του επομένου. Οι νευρώνες του ίδιου επιπέδου δε συνδέονται μεταξύ τους.
- Μη-γραμμική συνάρτηση ενεργοποίησης (activation function).

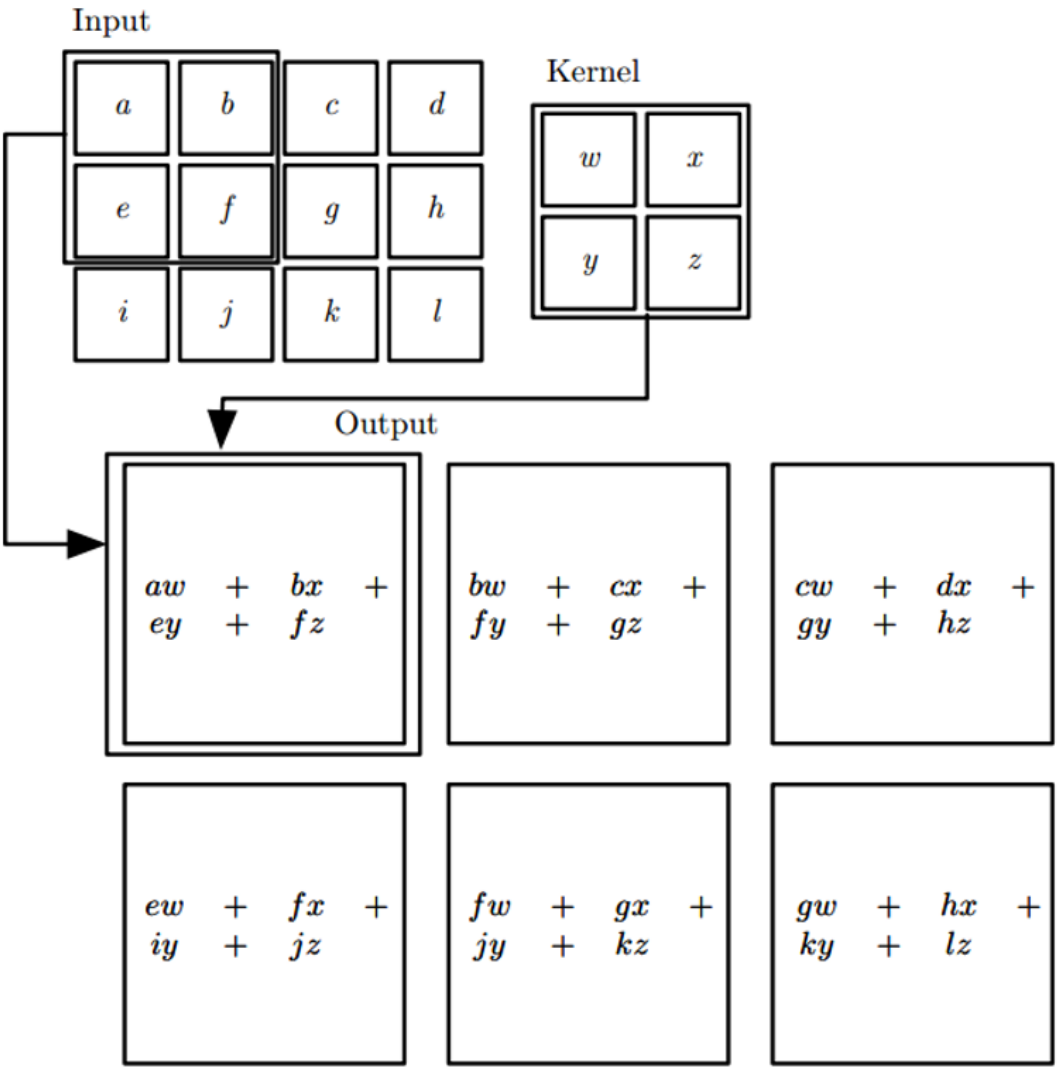
Χαρακτηριστικά:

- Ένα Multilayer Perceptron δίκτυο, είναι σε θέση να υλοποιήσει γραμμικές και μη-γραμμικές συναρτήσεις.
- Σύμφωνα με το [16], τα Multilayer Feedforward Networks μπορούν να προσεγγίσουν οποιαδήποτε συνάρτηση με οσηδήποτε ακρίβεια.
- Εκπαίδευση με back-propagation.

4.2 Convolutional Neural Network - CNN (Συνελικτικό Νευρωνικό Δίκτυο)

Κάθε νευρώνας στα κλασικά Νευρωνικά Δίκτυα ενός επιπέδου μεταφέρει την πληροφορία του σε όλους τους νευρώνες του επομένου επιπέδου. Και αντιστρόφως, κάθε νευρώνας αποκτά μία τιμή που προκύπτει και επηρεάζεται από τις τιμές όλων των νευρώνων του προηγουμένου επιπέδου.

Βασική ιδέα ενός Συνελικτικού Νευρωνικού Δικτύου (CNN) είναι η χρήση μίας μάσκας-πυρήνα (kernel) μεγέθους συνήθως μικρότερού από την είσοδο. [9] Όπως φαίνεται και στην εικόνα 6, ο πυρήνας αυτός εφαρμόζεται επανειλημμένα πάνω στην είσοδο. Με αυτή την τεχνική διευκολύνεται η επεξεργασία μεγάλων εισόδων και γίνεται εφικτή, όπως θα δούμε, η επεξεργασία εισόδων μεταβλητού μήκους.



Εικόνα 6: Η ιδέα της Συνέλιξης σε ένα CNN [9]

4.3 Recurrent Neural Network – RNN (Αναδρομικό Νευρωνικό Δίκτυο)

Τα RNN είναι νευρωνικά δίκτυα που επεξεργάζονται ακολουθίες εισόδων. [8] Τα CNN είναι κατάλληλα για περιπτώσεις όπου τα δεδομένα είναι οργανωμένα σε χωρικές διαστάσεις (π.χ. εικόνες σε 2-διάστατο πλέγμα). Τα RNN προσφέρουν σημαντικά πλεονεκτήματα για δεδομένα που σχετίζονται με κάποια χρονική εξέλιξη (time series).

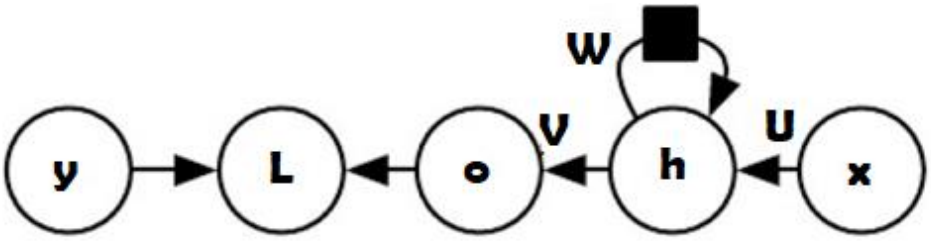
4.3.1 Διαμοιρασμός παραμέτρων (Parameter Sharing) στα RNN

Η τεχνική του Parameter Sharing χρησιμεύει στην επεξεργασία εισόδων διαφορετικού και μη- καθορισμένου μήκους. [8] Εφαρμόζουμε τον ίδιο πίνακα πάνω σε όλες τις εισόδους. Οι είσοδοι όμως μπορούν να είναι κομμάτια μιας μεγαλύτερης ακολουθίας.

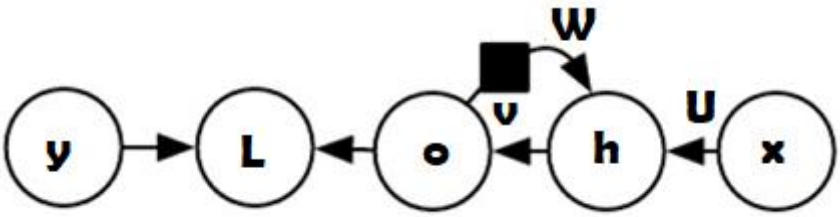
4.3.2 Είδος Ανάδρασης

Στα RNN έχουμε δύο βασικές προσεγγίσεις ως προς τον τρόπο με τον οποίο γίνεται η ανάδραση.

1. Από κρυφό (Hidden επίπεδο σε κρυφό επίπεδο). (Εικόνα 7)
2. Από το επίπεδο εξόδου (Output layer σε κάποιο κρυφό επίπεδο). (Εικόνα 8)
3. Συνδυασμός των δύο.



Εικόνα 7: RNN δίκτυο με ανάδραση τύπου 1, από κρυφό επίπεδο προς κρυφό επίπεδο. [8]



Εικόνα 8: RNN δίκτυο με ανάδραση τύπου 2, από το επίπεδο εξόδου προς κρυφό επίπεδο. [8]

Η πρώτη προσέγγιση είναι πιο ισχυρή αλλά η 2η μπορεί να εκπαιδευτεί πιο εύκολα.

4.4 Deep RN (Βαθιά Αναδρομικά Νευρωνικά Δίκτυα)

Ένα Νευρωνικό Δίκτυο που περιέχει περισσότερα από ένα κρυφά επίπεδα ονομάζεται βαθύ νευρωνικό δίκτυο. [8]

Με βάση τη ροή της πληροφορίας και τον υπολογισμό (computation) σε ένα RNN έχουμε τις ακόλουθες τρεις κατηγορίες:

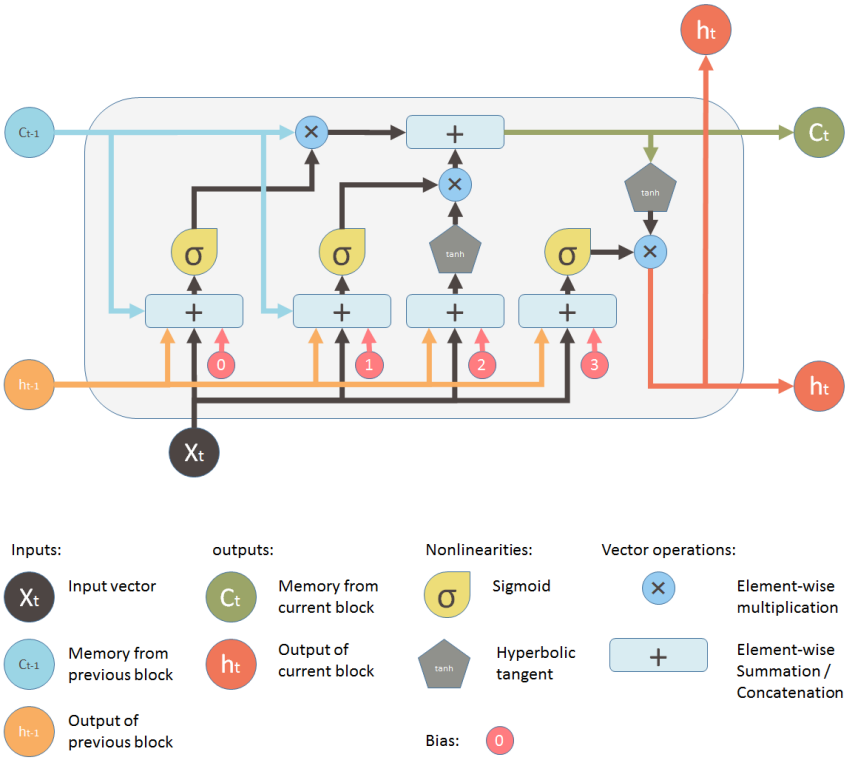
1. Είσοδο στη κατάσταση ενός κρυφού επιπέδου(hidden state - αφού έχουμε ανατροφοδότηση στα κρυφά επίπεδα μπορούμε πλέον να μιλάμε και για κατάσταση των κρυφών επιπέδων).
2. Μία κρυφή κατάσταση στην επόμενη
3. Η κρυφή κατάσταση στην έξοδο.

Η προσθήκη βάθους σε καθένα από αυτά τα στάδια οδηγεί πιθανώς σε βελτίωση [9, 10]. Απαιτείται ένα Deep RNN έτσι ώστε να μπορέσουν να αποτυπωθούν και να αποκωδικοποιηθούν οι εξαρτήσεις των δεδομένων εισόδου.

Η παράγωγος, όταν τροφοδοτείται συνεχόμενα σε πολλά επίπεδα ενός δικτύου, έχει την τάση να παίρνει είτε πολύ μικρές είτε πολύ μεγάλες τιμές. Ωστόσο, εξαιτίας των ανατροφοδοτήσεων που υπάρχουν σε ένα RNN, το παραπάνω γεγονός μπορεί να αποτελέσει πρόβλημα στην εκπαίδευση ενός RNN. Σημειώνεται ότι τις περισσότερες φορές η παράγωγος τείνει προς μικρές τιμές (vanished – gradient problem). Αυτό σημαίνει ότι μία κατάσταση ή ένα πρότυπο (pattern) που έχει παρατηρηθεί δε θα μεταδοθεί (propagation) παρά μόνο στις αμέσως επόμενες καταστάσεις.

Επομένως, είναι δύσκολο να κρατήσει το δίκτυο πληροφορίες σχετικά με μακροχρόνιες εξαρτήσεις. Υπογραμμίζεται ότι στα RNN ένας πίνακας που συμμετέχει στο βρόχο ανάδρασης πολλαπλασιάζεται πολλές φορές με τον εαυτό του και έτσι οι τιμές του τείνουν προς το 0 (ή σπανιότερα προς το άπειρο) [8]. Από τα δύο προαναφερθέντα φαίνεται ότι η έκφραση των Long-Term Dependencies σε ένα νευρωνικό δίκτυο αποτελεί ένα σημαντικό ζήτημα και αντικείμενο μελέτης. Προς την επίλυσή τους έχουν γίνει διάφορες προτάσεις, όπως η χρήση συναρτήσεων ενεργοποίησης (activation functions) με τιμές που δεν τείνουν στο άπειρο, όπως π.χ. η ReLU ή η softmax.

Επομένως η κύρια διαφορά του RNN με το MLP είναι ότι περιέχει τουλάχιστον ένα βρόγχο ανατροφοδότησης. Για αυτό τον λόγο τα RNN είναι πιο κατάλληλα για διαδοχικά δεδομένα όπως οι χρονοσειρές. Ωστόσο ένα σημαντικό τους μειονέκτημα είναι ο μηδενισμός (vanishing gradient) και η έκρηξη παραγώγου (exploding gradient). [31] Για να αντιμετωπιστεί αυτό το πρόβλημα ανακαλύφθηκαν τα Επίπεδα Μακράς Βραχυπρόθεσμης Μνήμης (Long Short Term Memories - LSTM). Τα LSTM αποτελούνται από 3 πύλες όπως φαίνονται και στην παρακάτω εικόνα .



Εικόνα 9: Αναπαράσταση της λειτουργικότητας του LSTM [31]

4.5 Προσθέτοντας Μνήμη στα RNN: Το Δίκτυο LSTM

4.5.1 Η τεχνική των Leaky Units

Μία προσέγγιση στο πρόβλημα της εξασθένισης των βαρών (Weight Decaying) είναι να θέσουμε αναδράσεις μόνο από κάθε νευρώνα προς τον εαυτό του και να τους δώσουμε ένα σταθερό βάρος. [8] Συμβολίζουμε αυτό το βάρος με α.

Έτσι, κάθε μονάδα θα δίνει τη χρονική στιγμή t στην έξοδό της την τιμή

$$\mu^{(t)} = a\mu^{(t-1)} + (1 - a)v^{(t)}$$

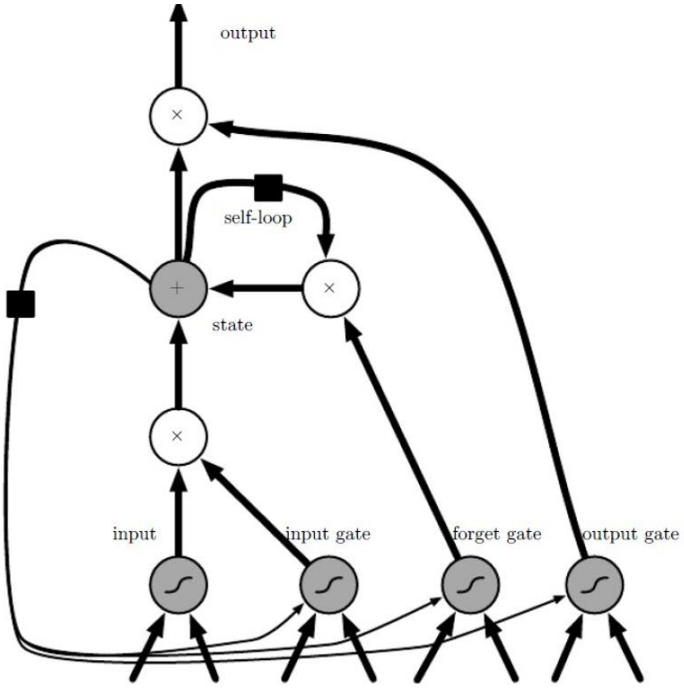
Αν το α είναι κοντά στη τιμή 1, τότε ο νευρώνας θα θυμάται πληροφορίες του παρελθόντος, ενώ αν είναι κοντά στο 0, οι πληροφορίες του παρελθόντος θα ξεχνιούνται γρήγορα. Η προσέγγιση αυτή αποτελεί την τεχνική των Leaky Units. Ωστόσο, το μειονέκτημά της είναι ότι το α θα είναι σταθερό. Είτε θέτουμε εξ' αρχής την τιμή του είτε αφήνουμε το σύστημα να το μάθει.

4.5.2 Η τεχνική των Skip Connections

Μία άλλη τεχνική αφορά τις λεγόμενες Skip Connections. Αυτές προσθέτουν στο σύστημα ανάδραση από τη χρονική στιγμή t στην t + d, όπου το d είναι και πάλι μία σταθερά. [8] Σε περιπτώσεις όπου το d > 1 το σύστημα συμβάλει στο να θυμάται πληροφορίες από το παρελθόν. Υπογραμμίζεται ωστόσο, αφού το d είναι σταθερά, το σύστημα δε έχει τη δυνατότητα να προσαρμοστεί στις ιδιαιτερότητες της εισόδου.

4.6 Τα δίκτυα LSTM

Τα Νευρωνικά Δίκτυα με πύλες (gated RNNs) προς το παρόν δίνουν την πιο αποτελεσματική λύση στην πράξη. Στην κατηγορία αυτή ανήκουν και τα LSTM RNNs (Long Short – Term Memory RNNs). [8] Σε αυτά, το κλασικό cell αντικαθίσταται με ένα gated cell. Αυτό εξωτερικά παραμένει ίδιο με το κλασικό, αλλά εσωτερικά χρησιμοποιεί πύλες, με βάρη που μαθαίνει, για να ελέγξει την ανάδραση. [32]



Εικόνα 10: Η εσωτερική δομή ενός LSTM κυττάρου [32]

Κάθε LSTM κύτταρο έχει εσωτερικά ένα cell στο οποία διατηρεί την κατάστασή του. [8] Και οι τρεις πύλες του σχήματος δέχονται την ίδια είσοδο από το εξωτερικό του LSTM κυττάρου και παράγουν έξοδό στο διάστημα 0 έως 1, με βάρη που βάρη που μαθαίνουν. Η πρώτη πύλη (input gate) καθορίζει το συντελεστή με τον οποίο θα πολλαπλασιαστεί η εξωτερική είσοδος του LSTM κυττάρου. Η δεύτερη πύλη (forget gate)

καθορίζει το συντελεστή με τον οποίο θα πολλαπλασιαστεί η εσωτερική ανάδραση του LSTM κυττάρου. Η επόμενη κατάσταση καθορίζεται από την τρέχουσα κατάσταση και από την ανάδραση. Η Τρίτη πύλη (output gate) καθορίζει το συντελεστή με τον οποίο η εσωτερική κατάσταση του κυττάρου θα περάσει στην έξοδο. Αυτή επομένως μπορεί να απενεργοποιήσει συνολικά το LSTM cell.

4.6.1 Εξισώσεις

Ακολουθούν οι εξισώσεις που διέπουν τη λειτουργία του LSTM κυττάρου. [8]

- Με U συμβολίζουμε τον πίνακα με τα βάρη που θα πολλαπλασιάσουν την εξωτερική είσοδο.
- Με W συμβολίζουμε τον πίνακα με τα βάρη που θα πολλαπλασιάσουν την εξωτερική ανάδραση. Υπενθυμίζεται εδώ ότι εξωτερικά ένα LSTM δίκτυο δε διαφέρει σε τίποτα από ένα RNN , άρα κάθε νευρώνας μπορεί να δέχεται ανάδραση τύπου 1 ή τύπου 2.
- Με b συμβολίζεται η σταθερά (bias).
- Τα b, U και W θα έχουν δείκτη g όταν αφορούν στην input gate, δείκτη f όταν αφορούν στην forget gate και δείκτη o όταν αφορούν στην output gate.

Για την input gate:

$$g_i^{(t)}=\sigma(b_i^g + \sum_j U_{i,j}^g x_j^{(t)} + W_{i,j}^g h_j^{(t-1)}$$

Για τη forget gate:

$$f_i^{(t)}=\sigma(b_i^f + \sum_j U_{i,j}^f x_j^{(t)} + W_{i,j}^f h_j^{(t-1)}$$

Για την output gate:

$$o_i^{(t)}=\sigma(b_i^o + \sum_j U_{i,j}^o x_j^{(t)} + W_{i,j}^o h_j^{(t-1)}$$

Η τιμή s της κατάστασης του κυττάρου τη χρονική στιγμή τ καθορίζεται τόσο από την s(t − 1), πολλαπλασιασμένη με την τιμή της forget gate, όσο και από την εξωτερική είσοδο (εξωτερική είσοδος είναι ο όρος μέσα στην παρένθεση), πολλαπλασιασμένη με την τιμή της input gate:

$$s_i(t) = f_i\,(\mathrm{t}) s_i(t-1) + g_i(t)\sigma(b_i + \sum_j U_{i,j}\, x_j^{(t)} + \sum_j W_{i,j}\, h_j^{(t-1)})$$

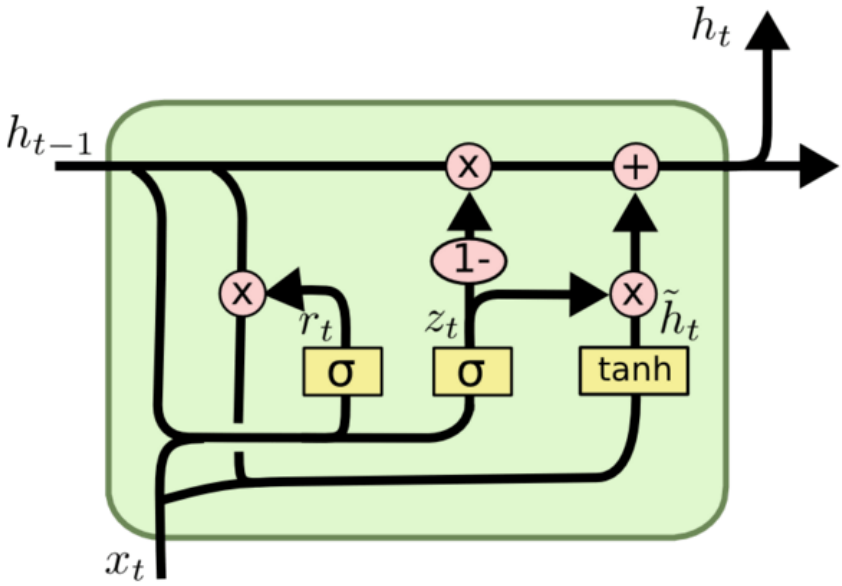
Σε όλες τις προηγούμενες εξισώσεις, ο δείκτης i υποδεικνύει τον i-οστό νευρώνα του επιπέδου στο οποίο βρισκόμαστε και ο δείκτης j, πάνω στον οποίο γίνονται όλα τα αθροίσματα, αναφέρεται σε όλους τους νευρώνες του προηγουμένου επιπέδου.

Το κύτταρο LSTM συνολικά χρησιμοποιεί τετραπλάσιες παραμέτρους από τον κλασικό νευρώνα:

- Τα b, U και W , που είναι το bias, ο πίνακας για τις εξωτερικές εισόδους και ο πίνακας για τις εξωτερικές αναδράσεις.
- Μία τριάδα b, U και W για κάθε πύλη

4.7 Τα GRU RNN δίκτυα

Οι Επανατροφοδοτούμενες Μονάδες με Πύλες (Gated Recurrent Units ή GRUs) αποτελούν μια άλλη μορφή επανατροφοδοτούμενων δικτύων τα οποία έδωσαν λύση στο πρόβλημα των εξαφανιζόμενων. [33] Ένα GRU είναι τυπικά ένα LSTM χωρίς την πύλη εξόδου Αυτό σημαίνει ότι καταγράφονται όλα τα περιεχόμενα από το κελί της μνήμης στο ευρύτερο δίκτυο σε χρονικό βήμα.



Εικόνα 11: Αναπαράσταση της λειτουργικότητας του GRU [33]

4.7.1 Η Διαδικασία της Μάθησης Μέσω Παραγώγων (Gradient Based Learning)

Οι περισσότεροι σύγχρονοι αλγόριθμοι Βαθιάς Μάθησης (Deep Learning) περιλαμβάνουν κάποιου είδους βελτιστοποίησης (optimization).[33] Πιο συγκεκριμένα περιλαμβάνουν την ελαχιστοποίηση (minimization) κάποιας Συνάρτησης Κόστους (Cost Function ή Loss Function ή Error Function ή Objective Function, όλοι οι όροι χρησιμοποιούνται με την ίδια έννοια στα πλαίσια της Μηχανικής Μάθησης).

Με τον όρο ελαχιστοποίηση εννοούμε την εύρεση εκείνου του $x*$ για το οποίο η Συνάρτηση κόστους f(x) παίρνει την ελάχιστη δυνατή τιμή της, είναι δηλαδή $x* = \text{argmin} f(x)$.

Η Συνάρτηση Κόστους f(x), όπου το x είναι παράμετρος ή διάνυσμα παραμέτρων, εκφράζει, στα πλαίσια της Μηχανικής Μάθησης, κάποιο σφάλμα. Η ελαχιστοποίηση αυτού του σφάλματος θα δώσει τις τελικές τιμές των παραμέτρων. Επομένως όταν το σύστημα «μαθαίνει» τις τιμές του x, στην ουσία βρίσκει το x εκείνο για το οποίο κάποια Συνάρτηση Κόστους ελαχιστοποιείται.

Δύο είναι τα βασικά στοιχεία ενός αλγορίθμου μάθησης:

1. η Συνάρτηση Κόστους: αξιολογεί τις εξόδους του αλγορίθμου μάθησης και αναθέτει έναν αριθμό-κόστος (Penalty) σε κάθε έξοδο. Έχουν προταθεί και χρησιμοποιηθεί πολλές Συναρτήσεις Κόστους.
2. η Μέθοδος Βελτιστοποίησης (Optimizer): μέθοδος η οποία προτείνει νέες τιμές για τις παραμέτρους, έχοντας ως είσοδο τις παραμέτρους του αλγορίθμου μάθησης και το κόστος που ανατέθηκε στην έξοδο του αλγορίθμου με σκοπό να μειωθεί η τιμή του κόστους.

4.8 Συναρτήσεις Κόστους (Loss Functions)

Προκειμένου να γίνει η εκπαίδευση ενός Νευρωνικού Δικτύου, απαιτείται ένας τρόπος αξιολόγησης της εξόδου του σε σχέση με την είσοδο και την αναμενόμενη έξοδο. Για το σκοπό αυτό χρησιμοποιούνται οι Συναρτήσεις Κόστους (Loss Functions). [9], [15], [16]

Στα ακόλουθα, τα y_{true}, y_{pred} είναι τα διανύσματα της πραγματικής, αναμενόμενης τιμής και της εξόδου του δικτύου αντίστοιχα. Όπου χρησιμοποιείται ο δείκτης j, δηλώνει το j-οστό στοιχείο του αντιστοίχου διανύσματος. Το σ δηλώνει εκτίμηση πιθανότητας (probability estimate).

1. *L1 ή MAE (Minimum Absolute Error)*

$$| y_{true} - y_{pred} |$$

2. *L2 ή MSE (Minimum Squared Error)*

$$| y_{true} - y_{pred} |$$

Είναι μία ειδική περίπτωση Maximum Likelihood Estimator όπου θεωρούμε ότι τα δεδομένα που έχουμε προς εκπαίδευση (το y ως συνάρτηση της εισόδου x) ακολουθούν κανονική κατανομή.

Συνηθίζεται περισσότερο σε προβλήματα Παλινδρόμησης (Regression). Είναι η προεπιλεγμένη (default) Συνάρτηση Κόστους στη Γραμμική Παλινδρόμηση (Linear Regression), αλλά χρησιμοποιείται αρκετά και στην εκπαίδευση νευρωνικών δικτύων.

3. *Hinge Loss*

$$\sum_j \max\left(0, \frac{1}{2} - y_{pred}^{j} * y_{true}^{j}\right)$$

όπου το y_{pred} πρέπει να μετατραπεί σε +- 1 μορφή. Χρησιμοποιείται σε ταξινόμηση (classification) με SVM (Support Vector Machines).

4. *Log Loss ή Maximum Likelihood ή Cross-Entropy*

$$-\sum_j y_{true}^{j} * \log\sigma(o)^{(j)}$$

Συνηθίζεται περισσότερο σε προβλήματα Ταξινόμησης (Classification). Χρησιμοποιείται σε Βαθιά Νευρωνικά Δίκτυα, ενώ έχει εκτενώς χρησιμοποιηθεί σε Αναγνώριση Φωνής [12] και Αναγνώριση Γραφής [13].

5. ΠΡΟΕΠΕΞΕΡΓΑΣΙΑ ΔΕΔΟΜΕΝΩΝ

5.1 Κανονικοποίηση (Normalization)

Πριν την έναρξη της εκπαίδευσης των δεδομένων είναι απαραίτητη η προεπεξεργασία τους.[14] Με την κανονικοποίηση, όλα τα χαρακτηριστικά (features) των δεδομένων εκπαίδευσης (training data) ανάγονται στο ίδιο εύρος, πράγμα που είναι αναγκαίο για τη σωστή εκπαίδευση του δικτύου. Ακολουθούν οι σημαντικότερες τεχνικές κανονικοποίησης.

5.2 Generalization, Overfitting - Underfitting

Γενίκευση (generalization) είναι η ικανότητα του εκπαιδευόμενου να προσαρμόζεται σε νέα δεδομένα, σε δεδομένα που δεν έχει προηγουμένως ξαναδεί. [15] Για να την μετρήσουμε, χωρίζουμε τα διαθέσιμα δεδομένα σε δύο μέρη, σε σετ εκπαίδευσης (training set) και σε σετ ελέγχου (test set).

Χρησιμοποιώντας το training set το δίκτυο εκπαιδεύεται και αποδίδει τιμές σε όλα τα βάρη του νευρωνικού δικτύου. Ως αποτέλεσμα δίνει την τιμή του

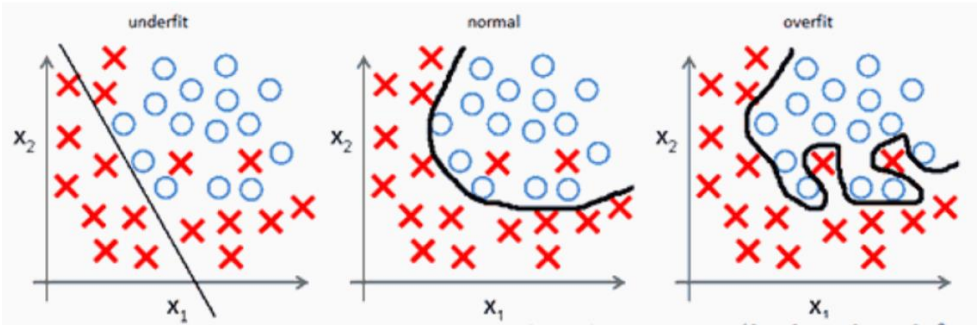
σφάλματος εκπαίδευσης (training error), που είναι η τιμή της συνάρτησης κόστους (Loss function) μετά το πέρας της εκπαίδευσης, χρησιμοποιώντας δηλαδή τις τελικές τιμές των βαρών (τα learned weights).

Από το test set μας παίρνουμε το σφάλμα ελέγχου (test error), που είναι η τιμή της συνάρτησης κόστους με τις τελικές τιμές των βαρών που το δίκτυο έχει μάθει. Οι δύο στόχοι είναι [34]:

1. Η ελαχιστοποίηση του training error.
2. Η ελαχιστοποίηση της διαφοράς μεταξύ training error και test error.

Η αδυναμία ελαχιστοποίησης του training error ονομάζεται Underfitting. Underfitting έχουμε όταν το δίκτυο δε μπορεί να προσεγγίσει καθόλου καλά τις αναμενόμενες εξόδους. Αυτό μπορεί να συμβαίνει για διάφορους λόγους, όπως η χαμηλή χωρητικότητα του δικτύου (έχουμε π.χ. χρησιμοποιήσει λίγα επίπεδα ή μικρό αριθμό νευρώνων ανά επίπεδο), η χρήση μόνο γραμμικών συναρτήσεων ενεργοποίησης (activation functions) ενώ τα δεδομένα δεν παρουσιάζουν γραμμική εξάρτηση κ.α.

Η αδυναμία ελαχιστοποίηση της διαφοράς μεταξύ training error και test error ονομάζεται Overfitting. Ο κυριότερος λόγος για τους οποίους αυτό μπορεί να συμβαίνει είναι το μικρό πλήθος δεδομένων εκπαίδευσης, που έχει ως αποτέλεσμα να μάθει το δίκτυο τιμές για τα βάρη που ναι μεν ελαχιστοποιούν, ή και μηδενίζουν, το training error, αλλά δε δίνουν σωστές προβλέψεις στο test set. Μία ακόμη αιτία είναι η πολύ μεγάλη χωρητικότητα του δικτύου ή η υπερβολική εκπαίδευση, δηλαδή εκπαίδευση με πάρα πολλές επαναλήψεις, πάρα πολλές εποχές. Ως αποτέλεσμα αυτού είναι η καλή εκμάθηση του training set, χωρίς όμως αυτό να συνεπάγεται δυνατότητα γενίκευσης στο test set.



Εικόνα 12: Underfitting και Overfitting [34]

Στην πρώτη εικόνα φαίνεται η αδυναμία του συστήματος να διακρίνει επαρκώς μεταξύ θετικών και αρνητικών δειγμάτων. Στη συγκεκριμένη περίπτωση οφείλεται στη γραμμικότητα του δικτύου, που δε μπορεί να προσομοιώσει τα μη-γραμμικά δεδομένα. Στην τρίτη εικόνα, αντίθετα,

βλέπουμε ένα δίκτυο που έχει «μάθει» έναν ιδιαίτερο και εξαρτώμενο από τα training data τρόπο διαχωρισμού των δειγμάτων σε θετικά και αρνητικά. Παρότι το σφάλμα εκπαίδευσης εδώ να είναι μηδενικό, το δίκτυο αυτό δε θα μπορεί κατά πάσα πιθανότητα να γενικευτεί, αφού η διαχωριστική γραμμή έχει προσαρμοστεί τέλεια στα συγκεκριμένα δεδομένα. [35] Έχουν προταθεί αρκετές μέθοδοι για regularization (Κανονικοποίηση). Οι σημαντικότερες είναι η L2 regularization, η Dropout και η Early Stopping.

5.2.1 L2 Regularization

Η μέθοδος συνίσταται ουσιαστικά στην προσθήκη ενός επιπλέον όρου στη συνάρτηση κόστους. [8] Έστω

$$J(w; Q, y)$$

η συνάρτηση κόστους, όπου θ είναι το διάνυσμα των βαρών, Q το διάνυσμα με τα χαρακτηριστικά (features) των εισόδων και οι y το διάνυσμα των (αναμενόμενων) εξόδων.

Η συνάρτηση κόστους μετά την προσθήκη του όρου κανονικοποίησης γίνεται:

$$J'(w; Q, y) = J(w; Q, y) + \lambda\Omega(w)$$

όπου λ είναι μία παράμετρος που καθορίζει τη βαρύτητα που θα έχει ο όρος κανονικοποίησης. Βλέπουμε ότι ο όρος κανονικοποίησης Ω εξαρτάται μόνο από τις τιμές των βαρών θ και όχι από τις εισόδους και τις εξόδους.

Στην περίπτωση που ο όρος Ω είναι δευτέρου βαθμού, έχουμε

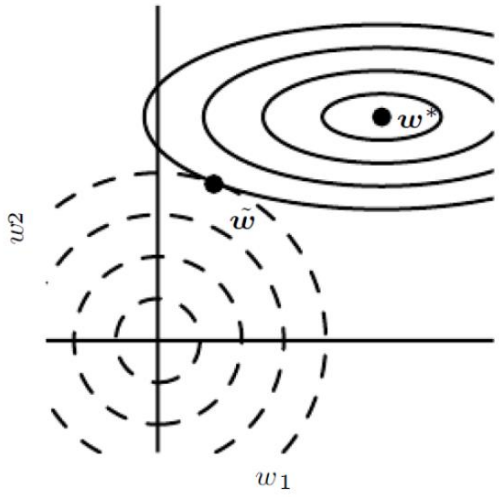
$$\Omega(w) = \frac{1}{2} W^T * W$$

και

$$J'(w; Q, y) = J(w; Q, y) + \frac{\lambda}{2} W^T * W$$

Η επίδραση που έχει ο όρος κανονικοποίησης είναι ότι δεν αφήνει τα βάρη ω να πάρουν την τιμή στην οποία ελαχιστοποιείται η Συνάρτηση Κόστους αλλά μία τιμή αρκετά κοντά σε αυτή. [36] Ως αποτέλεσμα, αποφεύγεται σε μεγάλο βαθμό το overfitting.

Η επίδραση του όρου κανονικοποίησης είναι περισσότερο έντονη στις διαστάσεις στις οποίες οι μεταβολές επηρεάζουν σημαντικά την τιμή της συνάρτηση κόστους (βλ. Εικόνα 13).



Εικόνα 13: Η επίδραση της κανονικοποίησης regularization στην ελαχιστοποίηση της συνάρτησης κόστους. [36]

Γραφική απεικόνιση της επίδρασης της L2 Regularization στη βέλτιστη τιμή των βαρών w*, σε ένα μοντέλο με 2 βάρη w1, w2. Οι ελλείψεις με τη συνεχή γραμμή είναι οι ισοδυναμικές γραμμές της συνάρτησης κόστους – χωρίς να λαμβάνεται υπόψη ο όρος κανονικοποίησης, ενώ οι διακεκομμένες γραμμές είναι οι ισοδυναμικές του όρου L2. Στην οριζόντια διάσταση, η τιμή της συνάρτησης κόστους δε μεταβάλλεται έντονα όταν μετακινούμαστε κοντά στο w*. Βλέπουμε ότι στη διάσταση αυτή, η επίδραση της κανονικοποίησης είναι έντονη, καθώς το w απέχει οριζόντια πολύ από το w*. Αντίθετα, στη δεύτερη διάσταση, η συνάρτηση κόστους είναι αρκετά πιο ευαίσθητη. Εδώ, η επίδραση της κανονικοποίησης είναι πιο μικρή. Έτσι, η L2 Regularization εμποδίζει μεν το overfitting, αλλά δε στερεί μεγάλη ποσότητα πληροφορίας από το δίκτυο. [8]

5.2.2 Η μέθοδος Early stopping

Μία απλή αλλά αποτελεσματική τεχνική αποφυγής του Overfitting είναι το Early stopping. [22]

Κατά το Overfitting, αυτό που παρατηρείται είναι από τη μία πλευρά να μειώνεται συνεχώς το σφάλμα εκπαίδευσης (training error) και από την άλλη να μη μειώνεται και από ένα σημείο και μετά να αυξάνεται το σφάλμα ελέγχου (test error).

Ελέγχοντας την τιμή του test error, η τεχνική του Early stopping μας λέει να σταματήσουμε την εκπαίδευση όταν αυτή αρχίσει να αυξάνεται.

Μία παράμετρος που εισάγεται στο σύστημα είναι ο «χρόνος» εκπαίδευσης – δηλαδή το πλήθος των επαναλήψεων ή οι εποχές - μέχρι να γίνει ο έλεγχος του test error. Αν T αυτό το πλήθος, τότε το υπολογιστικό

κόστος της τεχνικής αυτής είναι μικρό, συνίσταται μόνο στο τρέξιμο του test set και την παραγωγή προβλέψεων μία φορά ανά Τ επαναλήψεις.

5.2.3 Η τεχνική του Dropout

Η Dropout τεχνική εισάγει μία σταθερά p στο διάστημα [0, 1], η οποία καθορίζει την πιθανότητα με την οποία η έξοδος κάθε νευρώνα θα τίθεται από το σύστημα σε 0. Καθορίζει έτσι την πιθανότητα με την οποία ένας νευρώνας θα αποκόπτεται από το δίκτυο. [22]

Σε μία μαθηματική διατύπωση, μπορούμε να θεωρήσουμε μία μάσκα μ, μήκους ίσου με το πλήθος των βαρών του νευρωνικού δικτύου μας. Έτσι, η συνάρτηση κόστους γίνεται J(θ, μ), εξαρτάται δηλαδή από την τιμή των βαρών και από τη μάσκα που θα επιλέξουμε.

Η εκπαίδευση του δικτύου συνίσταται τώρα στην ελαχιστοποίηση της μέσης τιμής E[J(θ, μ)]. Ο μέσος όρος περιέχει εκθετικά πολλούς όρους, μπορούμε όμως να υπολογίσουμε μία καλή προσέγγισή του δειγματολειπτώντας το μ.

6. ΑΞΙΟΛΟΓΗΣΗ ΜΟΝΤΕΛΩΝ

Μετά την επιλογή του κατάλληλου μοντέλου και των παραμέτρων αυτού, ακολουθεί η αξιολόγηση της ακρίβειας της πρόβλεψης που προκύπτει. Αυτό επιτυγχάνεται με το διαχωρισμό των δεδομένων σε σετ εκπαίδευσης (training set) και σετ δοκιμών (test set). Το training set χρησιμοποιείται για το την εκπαίδευση του μοντέλου και τον υπολογισμό των προβλεπόμενων τιμών και το test set για την αξιολόγηση της ακρίβειας. [37] Το μέγεθος του training και του test set είναι τυπικά περίπου 80% και 20% αντιστοίχως. Σημειώνεται ότι οι τιμές αυτές εξαρτώνται και από το μέγεθος των δεδομένων αλλά και από το πόσο θέλουμε να προβλέψουμε. Ιδανικά το test set θα πρέπει να είναι τουλάχιστον όσο μεγάλο όσο και η πρόβλεψη που απαιτείται. Για να προσδιοριστεί η ακρίβεια μιας πρόβλεψης χρησιμοποιούνται διάφορα μέτρα αξιολόγησης. Γενικά το σφάλμα πρόβλεψης θεωρείται η διαφορά μεταξύ της πραγματικής και της προβλεπόμενης τιμής και εκφράζεται ως:

$$e_t = Y_t - \hat{Y}_t$$

όπου Y_t η πραγματική τιμή και $\hat{Y}_t$ η προβλεπόμενη για τη χρονική στιγμή t. Παρακάτω θα κάνουμε ανάλυση των πιο γνωστών σφαλμάτων.

6.1 Μέσο Τετραγωνικό Σφάλμα (Mean Squared Error - MSE)

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(Y_i - \hat{Y}_i)^2$$

Το μέσο τετραγωνικό σφάλμα μετράει την απόκλιση του σφάλματος υψωμένο στο τετράγωνο για όλο το σύνολο της πρόβλεψης. [37] Εξαιτίας του τετραγώνου τα σφάλματα αντίθετου πρόσημου δεν ακυρώνονται μεταξύ τους. Ένα άλλο χαρακτηριστικό του MSE είναι ότι «τιμωρεί» τα μεγάλα σφάλματα περισσότερο από τα μικρά. Σημειώνεται ωστόσο, ότι δεν είναι εύκολα ερμηνεύσιμο, αλλά θεωρείται καλό μέτρο όσον αναφορά τα σφάλματα.

6.2 Ρίζα Μέσου Τετραγωνικού Σφάλματος (Root Mean Squared Error -RMSE)

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(Y_i - \hat{Y}_i)^2}$$

Η ρίζα του μέσου τετραγωνικού σφάλματος παρουσιάζει τις ίδιες ιδιότητες με αυτές του μέσου τετραγωνικού σφάλματος. [37] Το πλεονέκτημα αυτού σε σχέση με το MSE είναι ότι βρίσκεται στην ίδια κλίμακα με τα δεδομένα οπότε είναι και πιο εύκολα να το κατανοήσει κάποιος μη-γνώστης.

6.3 Μέσο Απόλυτο Σφάλμα (Mean Absolute Error -MAE)

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|Y_i - \hat{Y}_i|$$

Το μέσο τετραγωνικό σφάλμα μετράει τη μέση απόλυτη απόκλιση των προβλεπόμενων τιμών από τις πραγματικές για όλο το σύνολο της πρόβλεψης. [37] Λόγω της απόλυτης τιμής του οι θετικές τιμές δεν αναιρούνται από τις αρνητικές. Επίσης το σφάλμα είναι στην ίδια κλίμακα με τα δεδομένα.

6.4 Μέσο Απόλυτο τετραγωνικό Σφάλμα (Mean Absolute Percentage Error -MAPE)

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}\frac{|Y_i - \hat{Y}_i|}{Y_i} * 100\%$$

Το MAPE εκφράζεται σε ποσοστό και γίνεται εύκολα κατανοητό στο ευρύ κοινό. [37] Λόγω του ποσοστού έχει την δυνατότητα να συγκρίνει υποδείγματα σε διαφορετικής κλίμακας δεδομένα. Ωστόσο, μπορεί να

χρησιμοποιηθεί μόνο σε χρονοσειρές με τιμές αρκετά μεγαλύτερες του 1, καθώς αν η πραγματική τιμή είναι κοντά ή ίση με το μηδέν, το σφάλμα τείνει στο άπειρο.

6.5 Συντελεστής προσδιορισμού (R^2)

$$R^2 = 1 - \frac{\frac{1}{N}\Sigma_{i=1}^{N}(Y_i - \hat{Y}_i)^2}{\frac{1}{N}\Sigma_{i=1}^{N}(Y_i - \bar{Y}_i)^2} = 1 - \frac{MSE}{\frac{1}{N}\Sigma_{i=1}^{N}(Y_i - \bar{Y}_i)^2}$$

Όπου,

$$\bar{Y} = \frac{1}{N}\sum_{\iota=1}^{N} Y_i$$

Είναι στην ουσία μια κανονικοποιημένη έκδοση του MSE με κλίμακα 0 και 1.

7. ΔΕΔΟΜΕΝΑ

Για την παρούσα εργασία εξετάστηκαν 3 πακέτα δεδομένων.

1. Individual household electric power consumption Data Set – Πηγή UCI [24]
2. Beijing PM2.5 Data Data Set – Πηγή UCI [23]
3. EMC Data Science Global Hackathon (Air Quality Prediction) - Πηγή Kaggle [25]

7.1 Δεδομένα Οικιακής Κατανάλωσης Ενέργειας

Το σύνολο δεδομένων Κατανάλωσης Ενέργειας Οικιακής Χρήσης (Individual Household Electric Power Consumption) είναι ένα σύνολο δεδομένων πολυδιάστατης χρονοσειράς που περιγράφει την κατανάλωση ηλεκτρικής ενέργειας για ένα νοικοκυριό για περίοδο τεσσάρων χρόνων, από το Δεκέμβριο 2006 μέχρι και το Νοέμβριο 2010, το οποίο παρέχεται από το UCI Machine Learning repository. Επίσης, οι παρατηρήσεις αυτές συλλέχθηκαν ανά λεπτό.

Πρόκειται για μια πολυδιάστατη (multivariate) χρονοσειρά που αποτελείται από επτά μεταβλητές (πέρα από την ημερομηνία και την ώρα). Αυτές είναι:

- *global_active_power*: Η συνολική ενεργή ισχύς που καταναλώνεται από το νοικοκυριό (kilowatts).
- *global_reactive_power*: Η συνολική ισχύς που επιστρέφεται από το νοικοκυριό (kilowatts).
- *voltage*: Μέση τάση (volts).
- *global_intensity*: Μέση ένταση ρεύματος (amps).
- *sub_metering_1*: ενεργή ενέργεια για την κουζίνα (watt-hours ενεργούς ενέργειας).
- *sub_metering_2*: ενεργή ενέργεια για ρούχα (watt-hours ενεργούς ενέργειας).
- *sub_metering_3*: ενεργή ενέργεια για συστήματα ελέγχου του κλίματος (watt-hours ενεργούς ενέργειας).

Σημειώνεται ότι η ενεργός ενέργεια (active energy) είναι η πραγματική ενέργεια που καταναλώνεται από το νοικοκυριό, ενώ η επανενεργός ενέργεια (reactive energy) είναι η αχρησιμοποίητη που βρίσκεται στο κύκλωμα του νοικοκυριού.

Το σύνολο των δεδομένων παρέχει την ενεργώς ισχύ καθώς και κάποια μέρη της ενεργούς ισχύος από το κύριο κύκλωμα στο σπίτι, συγκεκριμένα την κουζίνα, το πλυντήριο και τον έλεγχο του κλίματος. Αυτά δεν είναι όλα τα κυκλώματα του νοικοκυριού. Οι υπόλοιπες watt-hours μπορούν να υπολογιστούν από την ενεργό ενέργεια μετατρέποντας κατ' αρχάς την ενεργό ενέργεια σε ώρες ρεύματος και κατόπιν αφαιρώντας την άλλη υπο-μετρική ενεργό ενέργεια σε watt-hours, ως εξής:

$$sub\_metering\_remainder = (global\_active\_power*1000/60) - (sub\_metering\_1+sub\_metering\_2 + sub\_metering\_3)$$

7.1.1 Προετοιμασία των δεδομένων

Παρακάτω είναι οι πρώτες γραμμές δεδομένων (και η κεφαλίδα) από το ακατέργαστο αρχείο.

```
Date;Time;Global_active_power;Global_reactive_power;Voltage;Global_intensity;Sub_metering_1;Sub_metering_2;Sub_metering_3
16/12/2006;17:24:00;4.216;0.418;234.840;18.400;0.000;1.000;17.000
16/12/2006;17:25:00;5.360;0.436;233.630;23.000;0.000;1.000;16.000
16/12/2006;17:26:00;5.374;0.498;233.290;23.000;0.000;2.000;17.000
16/12/2006;17:27:00;5.388;0.502;233.740;23.000;0.000;1.000;17.000
16/12/2006;17:28:00;3.666;0.528;235.680;15.800;0.000;1.000;17.000
```

Εικόνα 14: Οι πρώτες γραμμές του ακατέργαστου dataset

Όπως φαίνεται παραπάνω, οι στήλες δεδομένων χωρίζονται με ερωτηματικά (';'). Κάθε γραμμή περιέχει τα ημερήσια δεδομένα για τη χρονική περίοδο που παρέχεται το dataset. Τα δεδομένα έχουν και κενές τιμές. Για παράδειγμα, μπορούμε να δούμε δεδομένα που λείπουν για 2-3 ημέρες γύρω στις 28/4/2007.

```
28/4/2007;00:19:00;0.490;0.202;235.020;2.200;0.000;0.000;0.000
28/4/2007;00:20:00;0.492;0.208;236.240;2.200;0.000;0.000;0.000
28/4/2007;00:21:00;?;?;?;?;?;?;?
28/4/2007;00:22:00;?;?;?;?;?;?;?
28/4/2007;00:23:00;?;?;?;?;?;?;?
28/4/2007;00:24:00;?;?;?;?;?;?;?
28/4/2007;00:25:00;?;?;?;?;?;?;?
28/4/2007;00:26:00;?;?;?;?;?;?;?
```

Εικόνα 15: Παράδειγμα σειρών με ελλείπουσες τιμές στο ακατέργαστο dataset

Αρχικά φορτώνουμε το αρχείο δεδομένων ως Pandas DataFrame, κάνοντας χρήση της συνάρτησης read_csv().

Για την σωστή εισαγωγή του καθορίζεται:

- το διαχωριστικό μεταξύ των στηλών το ερωτηματικό (sep = ';')
- η γραμμή 0 ότι περιέχει τα ονόματα των στηλών (header = 0)
- η φόρτωση των δεδομένων ως ένα πίνακα αντικειμένων αντί για πίνακα αριθμών, λόγω των τιμών '?' για τα ελλείποντα δεδομένα (low_memory = False)
- αυτόματα από το Pandas η μορφή ημερομηνίας-ώρας κατά την ανάλυση δεδομένων, η οποία είναι αρκετά πιο γρήγορη (infer_datetime_format = True)
- μια νέα στήλη που ονομάζεται 'datetime' ώστε να αναλυθούν μαζί οι στήλες ημερομηνίας και ώρας (parse_dates = {'datetime': [0,1]})
- και τέλος η νέα αυτή στήλη να είναι ο δείκτης για το DataFrame (index_col = ['datetime']).

Στη συνέχεια, επισημαίνουμε όλες τις τιμές που λείπουν και υποδεικνύονται με χαρακτήρα '?' με τη τιμή NaN, η οποία είναι type float ώστε να μπορέσουμε να δουλέψουμε με τα δεδομένα ως έναν πίνακα από float τιμές αντί για string.

Επιπλέον, δημιουργούμε μια νέα στήλη που να περιέχει το υπόλοιπο της κατανάλωσης ρεύματος, χρησιμοποιώντας τον υπολογισμό που αναλύθηκε προηγουμένως (*sub_metering_remainder*).

Τέλος, αποθηκεύουμε την τροποποιημένη έκδοση του dataset σε ένα νέο αρχείο, αλλάζοντας την επέκταση αρχείου σε .csv

```
datetime,Global_active_power,Global_reactive_power,Voltage,Global_intensity,Sub_metering_1,Sub_metering_2,Sub_metering_3,sub_metering_4
2006-12-16 17:24:00,4.216,0.418,234.840,18.400,0.000,1.000,17.0,52.26667
2006-12-16 17:25:00,5.360,0.436,233.630,23.000,0.000,1.000,16.0,72.333336
2006-12-16 17:26:00,5.374,0.498,233.290,23.000,0.000,2.000,17.0,70.566666
2006-12-16 17:27:00,5.388,0.502,233.740,23.000,0.000,1.000,17.0,71.8
2006-12-16 17:28:00,3.666,0.528,235.680,15.800,0.000,1.000,17.0,43.1
```

Εικόνα 16: Οι πρώτες γραμμές του τροποποιημένου dataset

Έτσι, στο νέο αρχείο 'household_power_consumption.csv' έχει προστεθεί μια νέα στήλη στο τέλος, η ημερομηνία και η ώρα έχουν συγχωνευτεί σε

μια στήλη την αρχή και οι παρατηρήσεις που λείπουν σημειώνονται με κενό, όπου το Pandas το διαβάζει ως NaN. Για παράδειγμα:

```
2008-07-13 20:25:00,0.440,0.000,239.190,2.000,0.000,0.000,0.0,7.3333335
2008-07-13 20:26:00,,,,,,,,
2008-07-13 20:27:00,,,,,,,,
2008-07-13 20:28:00,0.440,0.072,240.020,1.800,0.000,0.000,0.0,7.3333335
```

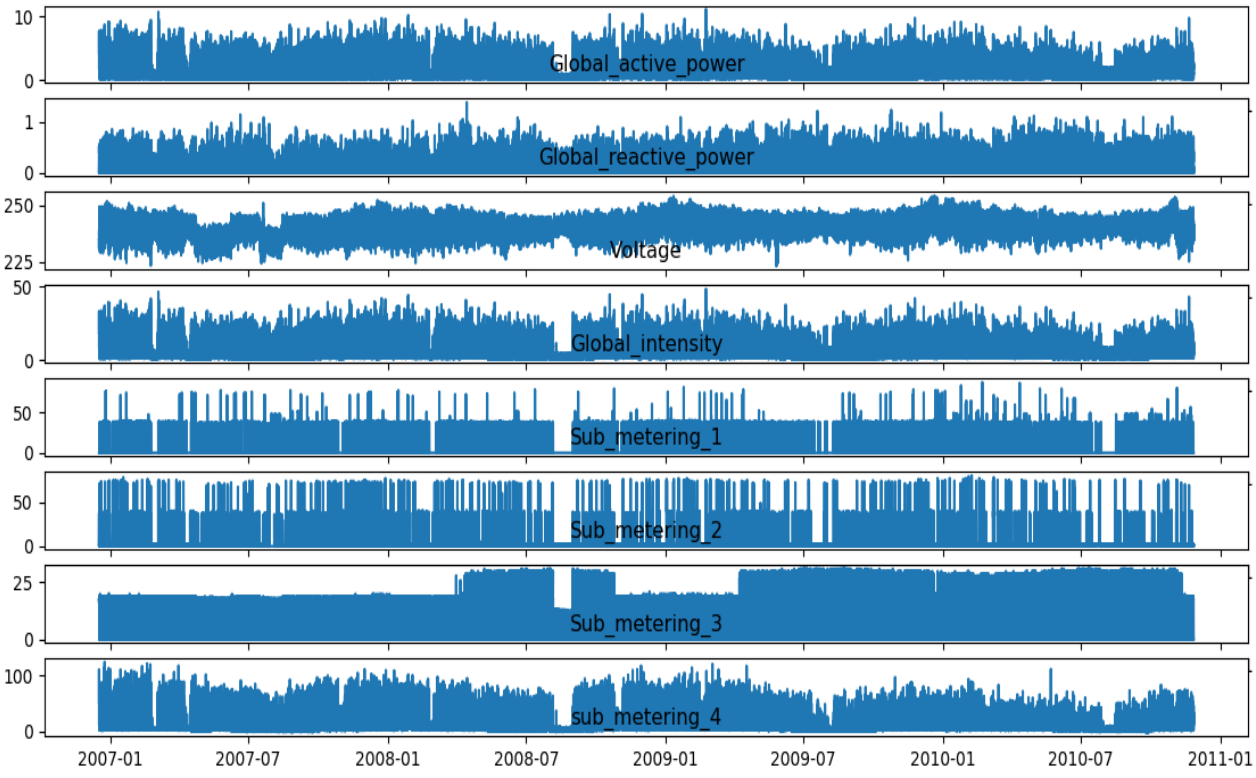
Εικόνα 17: Παράδειγμα σειρών με ελλείπουσες τιμές στο τροποποιημένο dataset

Η τροποποιημένη έκδοση του dataset, χρησιμοποιήθηκε για τις οπτικοποιήσεις με σκοπό περαιτέρω διερεύνησή του.

7.1.1.1 Οπτικοποίηση των δεδομένων

Ο καλύτερο τρόπος για την κατανόηση των δεδομένων είναι η δημιουργία γραφημάτων.

Ξεκινάμε με τη δημιουργία ξεχωριστών γραφημάτων για κάθε μία από τις οκτώ μεταβλητές. Αρχικά, δημιουργείται ένα σχήμα όπου περιέχει οκτώ γραφήματα, ένα για κάθε μεταβλητή.



Εικόνα 18: Η γραφική αναπαράσταση των οχτώ μεταβλητών του dataset για όλη την περιεχόμενη περίοδο

Μια πρώτη παρατήρηση είναι για το sub_metering_3. Για αυτή τη μεταβλητή έχουμε κάθετες μεταβολές μεγάλου χρονικού διαστήματος που

μπορεί να οφείλεται σε αλλαγή του τρόπου θέρμανσης/ψύξης για το συγκεκριμένο νοικοκυριό.

Ενδιαφέρον παρουσιάζει η μεταβλητή sub_metering_4 που φαίνεται να μειώνεται με την πάροδο του χρόνου ή να δείχνει πτωτική τάση. Πιθανώς να υπάρχει μια αντιστρόφως ανάλογη σχέση με τη μεταβλητή sub_metering_3 καθώς παρατηρείται σταθερή αύξηση προς το τέλος της σειράς.

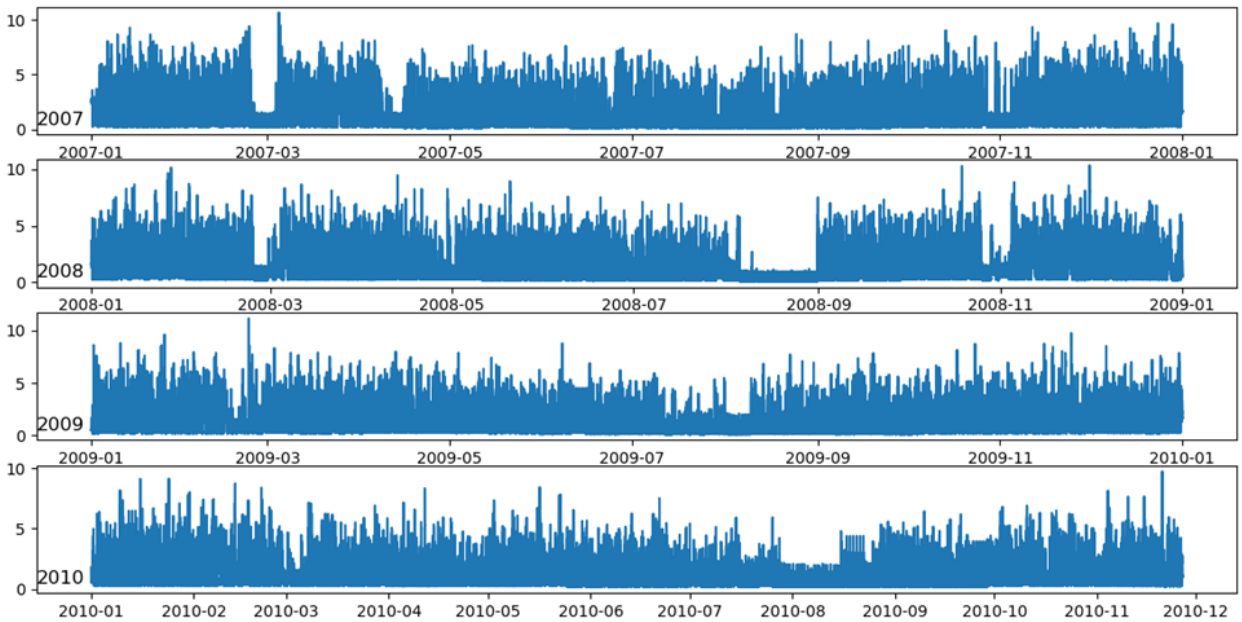
Οι παρατηρήσεις αυτές ενισχύουν την ανάγκη να τηρηθεί η χρονική διάταξη των ακολουθιών αυτών των δεδομένων κατά την τοποθέτηση και αξιολόγηση κάθε μοντέλου.

Επίσης, παρατηρούμε εποχικότητα στο «Global_active_power» και κάποιες άλλες παραλλαγές.

Υπάρχουν κάποιες κορυφές στη κατανάλωση που μπορεί να συμπίπτουν με συγκεκριμένες περιόδους, όπως τα Σαββατοκύριακα, όπου αναμένουμε να υπάρχει μεγαλύτερη κατανάλωση ενέργειας από ένα νοικοκυριό.

Εστιάζουμε στη μεταβλητή "Global active power" ή "active power" εν συντομία.

Σε πρώτο βήμα, δημιουργούμε ένα νέο γράφημα για την active power για κάθε έτος για να δούμε αν υπάρχουν κοινά πρότυπα κατά τη διάρκεια των χρόνων. Το πρώτο έτος, το 2006, έχει λιγότερο από ένα μήνα δεδομένων, έτσι αφαιρέθηκε από το γράφημα.



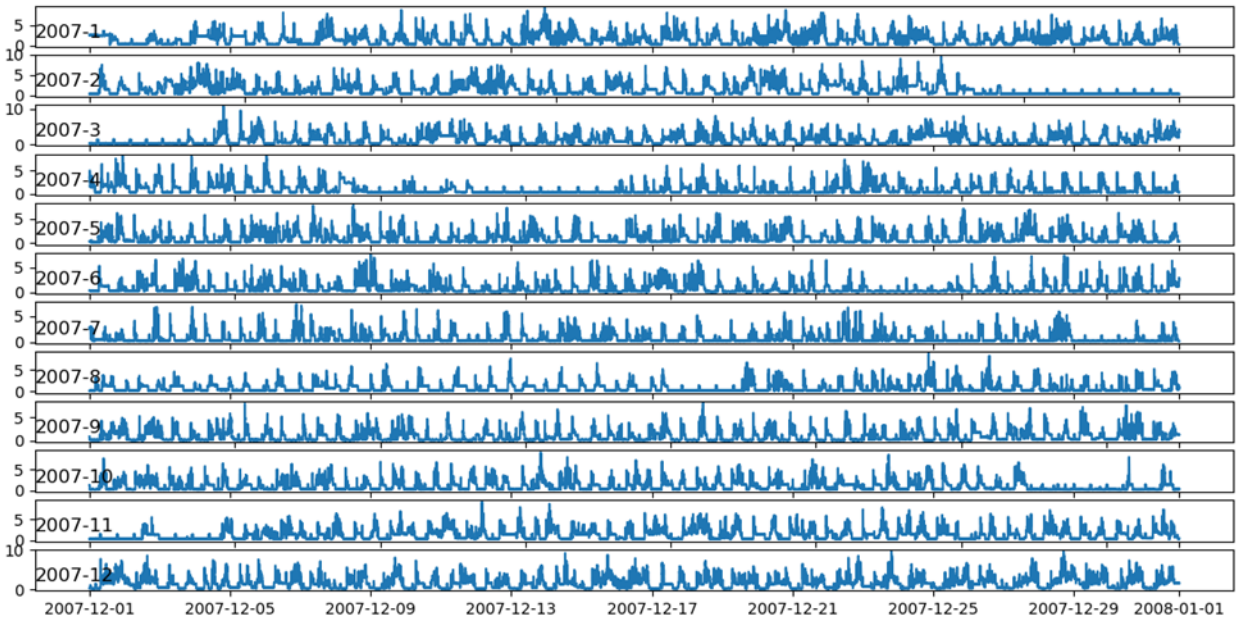
Εικόνα 19: Η γραφική αναπαράσταση της μεταβλητής global active power του dataset για τα έτη 2007, 2008, 2009 και 2010

Παρατηρούμε ότι εμφανίζονται παρόμοιες συμπεριφορές για τη περίοδο Φλεβάρη-Μάρτιο και τη περίοδο Αύγουστο-Σεπτέμβριο, όπου παρουσιάζεται σημαντική μείωση στην κατανάλωση.

Παρατηρούμε επίσης πτωτική τάση κατά τους καλοκαιρινούς μήνες και ίσως μεγαλύτερη κατανάλωση τους χειμερινούς μήνες προς το τέλος των γραφημάτων. Αυτή η συμπεριφορά μπορεί να παρουσιάζει ένα ετήσιο εποχιακό μοντέλο κατανάλωσης.

Τέλος, μπορούμε να δούμε μερικά κομμάτια των δεδομένων που λείπουν τουλάχιστον στο πρώτο, το τρίτο και το τέταρτο γράφημα.

Αν εστιάσουμε στη κατανάλωση για ένα δεδομένο έτος π.χ. 2007 μπορούμε να μελετήσουμε την active power για κάθε μήνα αυτού του έτους.



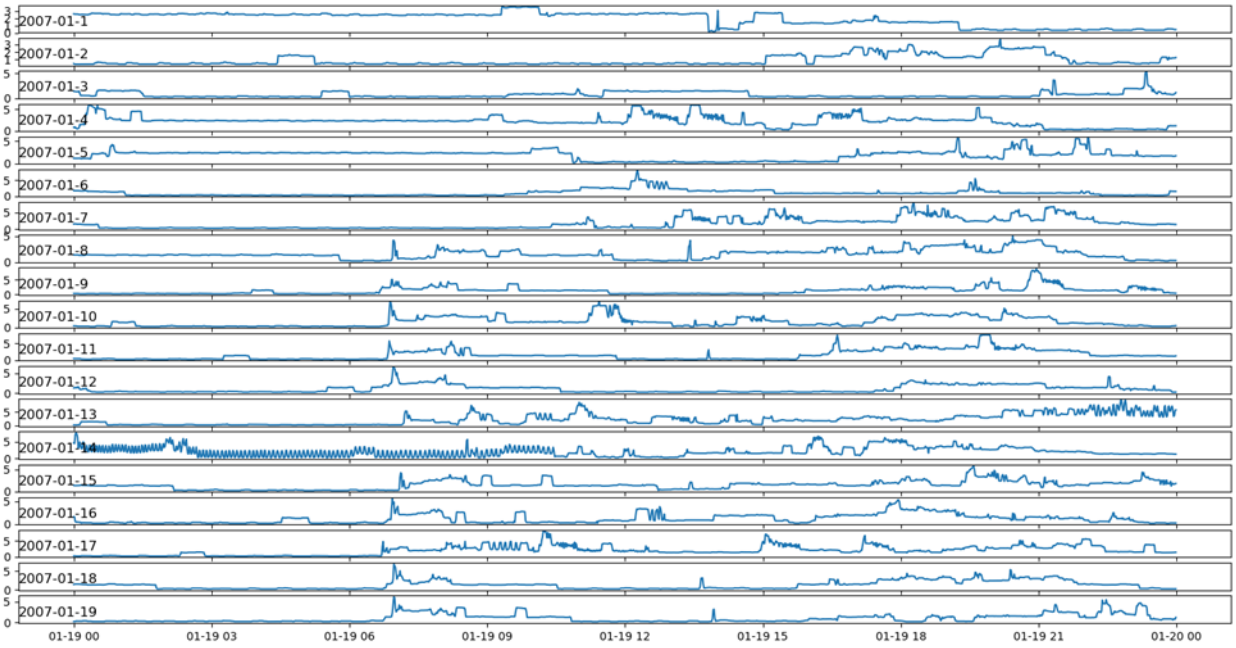
Εικόνα 20: Η γραφική αναπαράσταση της μεταβλητής global active power του dataset για όλους τους μήνες του έτους 2007

Παρατηρούμε ότι η ημερήσια κατανάλωση παρουσιάζει εποχικότητα μέσα σε κάθε μήνα.

Διαπιστώνουμε ότι υπάρχουν περιόδους ημερών με ελάχιστη κατανάλωση, όπως τον Αύγουστο και τον Απρίλιο. Αυτές πιθανώς να αντιπροσωπεύουν περιόδους διακοπών όπου το σπίτι ήταν ελεύθερο και η κατανάλωση ενέργειας ήταν ελάχιστη.

Τέλος, μπορούμε να εστιάσουμε ακόμα περισσότερο στα δεδομένα και να εξετάσουμε την ημερήσια συμπεριφορά της κατανάλωσης.

Δημιουργείται ένα σχήμα με 20 γραφήματα, ένα για κάθε ημέρα, για τις πρώτες ημέρες του μήνα Γενάρη του έτους 2007.



Εικόνα 21: Η γραφική αναπαράσταση της μεταβλητής global active power του dataset για τις πρώτες 20 μέρες τον Ιανουάριο του 2007

Υπάρχει μια κοινή συμπεριφορά και σε επίπεδο ημερήσιας κατανάλωσης ενέργειας. Για παράδειγμα, η κατανάλωση πολλών ημερών αρχίζει από νωρίς το πρωί, περίπου 6-7 πμ.

Ορισμένες μέρες δείχνουν πτώση της κατανάλωσης στη μέση της ημέρας, η οποία μπορεί να έχει νόημα αν οι περισσότεροι ένοικοι βρίσκονται εκτός σπιτιού.

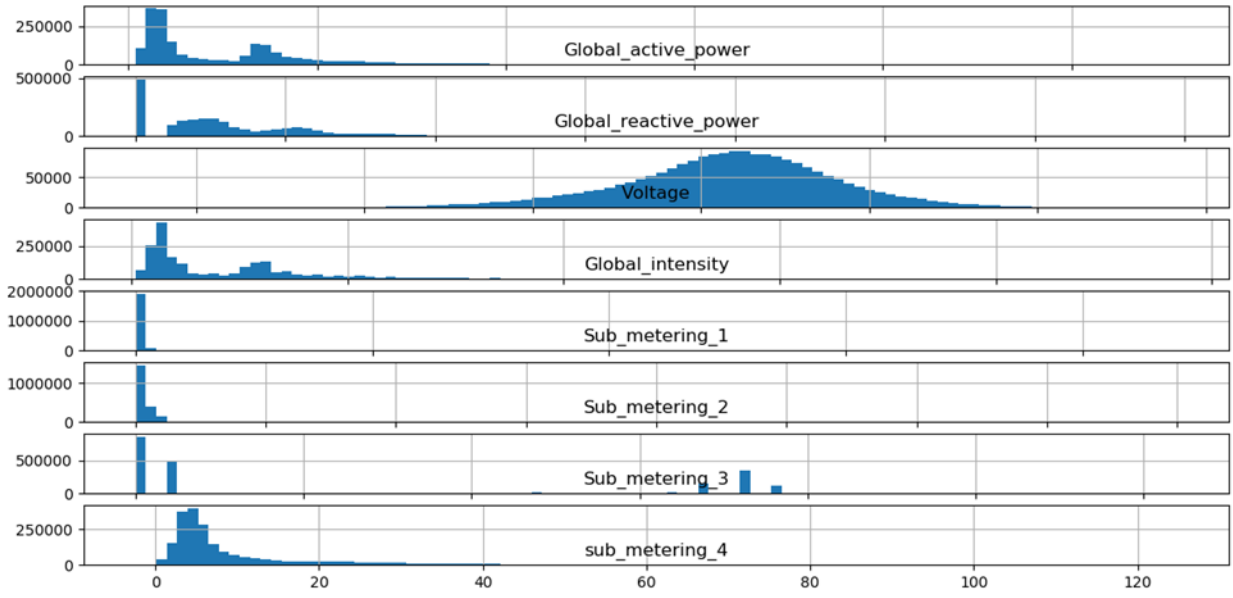
Βλέπουμε κάποια ισχυρή ολονύκτια κατανάλωση σε μερικές ημέρες, που για το μήνα Ιανουάριο μπορεί να ταιριάξει με ένα σύστημα θέρμανσης που χρησιμοποιείται.

Η χρονική περίοδος, ειδικά η εποχή και ο καιρός που τη χαρακτηρίζουν, είναι ένας σημαντικός παράγοντας στη μοντελοποίηση αυτών των δεδομένων.

7.1.1.2 Κατανομή μεταβλητών

Ένας άλλος σημαντικός τομέας που πρέπει να εξεταστεί είναι η κατανομή των μεταβλητών, δηλαδή αν οι κατανομές παρατηρήσεων ακολουθούν γκαουσιανή ή κάποια άλλη κατανομή. Μπορούμε να διερευνήσουμε τις κατανομές των δεδομένων ελέγχοντας ιστογράμματα.

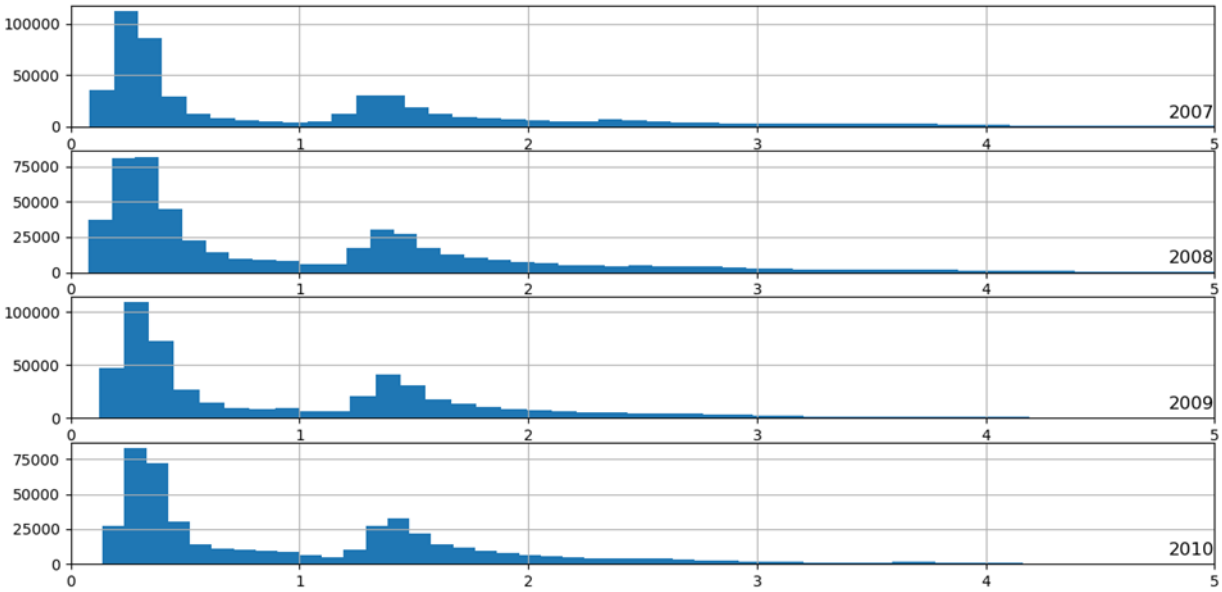
Αρχικά, δημιουργούμε ένα ιστόγραμμα για κάθε μεταβλητή από το σύνολο των δεδομένων.



Εικόνα 22: Η γραφική αναπαράσταση της κατανομής των οχτώ μεταβλητών του dataset

Παρατηρούμε ότι η active και η reactive power, η intensity, καθώς και οι sub_metered powers είναι όλες λοξές κατανομές προς τις μικρές watt-hours ή kilowatt τιμές. Επίσης, βλέπουμε ότι η κατανομή των δεδομένων τάσης (voltage) είναι έντονα γκαουσιανή.

Η κατανομή της active power φαίνεται να είναι bimodal, πράγμα που σημαίνει ότι έχει δύο μέσες ομάδες παρατηρήσεων. Διερευνούμε περαιτέρω αυτό εξετάζοντας τη κατανομή της active power για τα τέσσερα ολόκληρα χρόνια των δεδομένων.



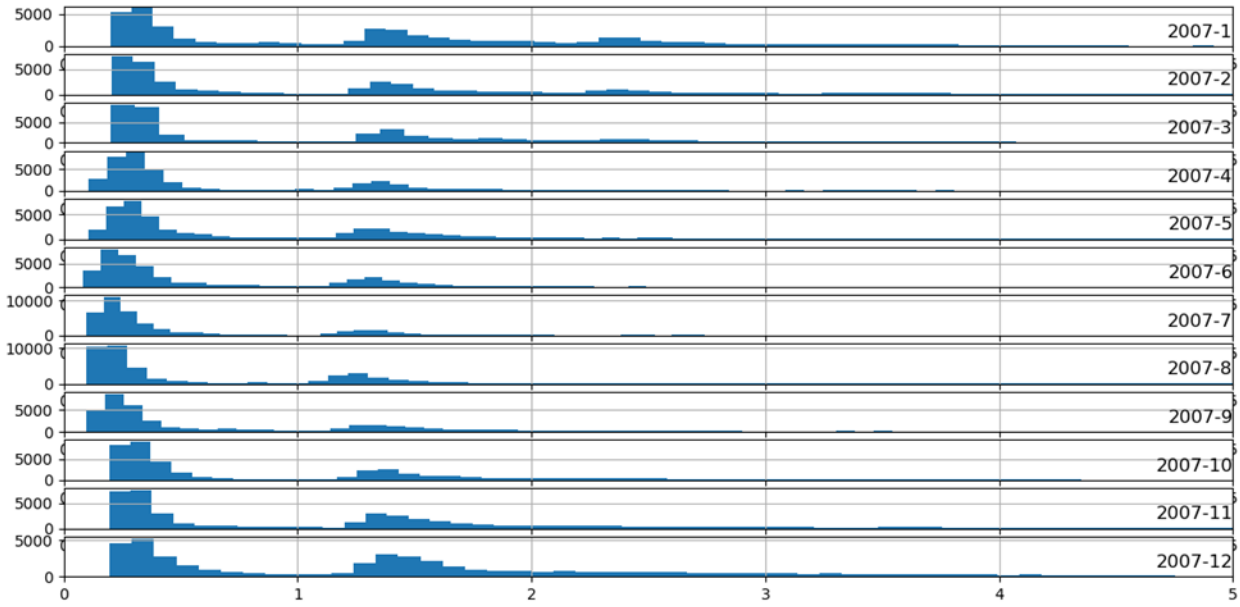
Εικόνα 23: Η γραφική αναπαράσταση της κατανομής της μεταβλητής global active power του dataset για τα έτη 2007, 2008, 2009 και 2010

Διαπιστώνεται ότι η κατανομή της active power κατά τη διάρκεια αυτών των ετών ακολουθεί όμοιο μοτίβο. Η κατανομή είναι bimodal, με μια κορυφή γύρω από 0,3 KW και ίσως άλλη περίπου 1,3 KW.

Θα μπορούσε να γίνει μια διάκριση στην ανάλυση των δεδομένων, διαχωρίζοντάς τα σε αυτά με κορυφή 1, κορυφή 2 ή μακρά ουρά. Αυτές οι ομάδες ή συστάδες όσον αφορά την ημερήσια ή την ωριαία κατανάλωση, μπορεί να είναι χρήσιμες για την ανάπτυξη ενός προγνωστικού μοντέλου.

Είναι πιθανό οι ομαδοποιημένες ομάδες να διαφέρουν κατά τη διάρκεια των εποχών του έτους.

Το διερευνούμε αυτό εξετάζοντας τη κατανομή για την active power για κάθε μήνα σε ένα χρόνο.



Εικόνα 24: Η γραφική αναπαράσταση της κατανομής της μεταβλητής global active power του dataset για όλους τους μήνες του έτους 2007

Παρατηρούμε την ίδια κατανομή των δεδομένων κάθε μήνα. Οι άξονες για τα γραφήματα φαίνονται να ευθυγραμμίζονται (δεδομένης της παρόμοιας κλίμακας) και μπορούμε να δούμε ότι οι κορυφές μετατοπίζονται προς τα αριστερά τους θερμότερους μήνες και δεξιότερα τους ψυχρότερους μήνες.

Επίσης βλέπουμε μια παχύτερη ή πιο προεξέχουσα ουρά προς μεγαλύτερες τιμές κιλοβάτ για τους ψυχρότερους μήνες, από Νοέμβριο μέχρι και Μάρτιο.

7.1.1.3 Περαιτέρω προετοιμασία δεδομένων

Ένα σημαντικό βήμα της προετοιμασίας των δεδομένων είναι η συμπλήρωση των κενών τιμών.

Έχοντας μετατρέψει τα '?' σε NaN τιμές, μπορούμε να αντιγράψουμε την παρατήρηση από την ίδια ώρα την προηγούμενη ημέρα. Η υλοποίηση αυτής της σκέψης γίνεται από τη συνάρτηση fill_missing() όπου λαμβάνει τον NumPy πίνακα των δεδομένων και συμπληρώνει τις κενές τιμές με αυτές ακριβώς 24 ώρες πριν.

7.1.1.4 Ορισμός προβλήματος πρόβλεψης

Εξετάζουμε το εξής πρόβλημα πρόβλεψης: ποια είναι η αναμενόμενη κατανάλωση ενέργειας για το νοικοκυριό για την επόμενη εβδομάδα με δεδομένη την πρόσφατη κατανάλωση ενέργειας. Αυτό το ερώτημα απαιτεί ένα μοντέλο πρόβλεψης όπου θα προβλέπει τη συνολική active power για κάθε ημέρα, τις επόμενες επτά ημέρες.

Ένα μοντέλο αυτού του τύπου θα μπορούσε να είναι χρήσιμο για τους ένοικους κατά τον προγραμματισμό των δαπανών. Ωφέλιμο είναι και από την πλευρά της προσφοράς, για τον προγραμματισμό της ζήτησης ηλεκτρικής ενέργειας για ένα συγκεκριμένο νοικοκυριό.

Αυτή η διαμόρφωση του συνόλου δεδομένων υποδεικνύει επίσης ότι θα ήταν χρήσιμο να συγχωνευτούν οι παρατηρήσεις από ανά λεπτό της κατανάλωσης ισχύος, σε ημερήσια σύνολα, δεδομένου ότι μας ενδιαφέρει η συνολική ισχύς ανά ημέρα.

Αυτό επιτυγχάνεται χρησιμοποιώντας τη συνάρτηση resample() στο Pandas DataFrame. Η κλήση αυτής της συνάρτησης με το όρισμα 'D' επιτρέπει την ομαδοποίηση των φορτωμένων δεδομένων που ταξινομούνται ανά ημερομηνία και ώρα. Στη συνέχεια, υπολογίζουμε το άθροισμα όλων των παρατηρήσεων για κάθε ημέρα και να δημιουργούμε ένα νέο σύνολο δεδομένων ημερήσιας κατανάλωσης ενέργειας για κάθε μία από τις οκτώ μεταβλητές.

Δημιουργούμε έτσι, ένα νέο σύνολο ημερήσιας συνολικής κατανάλωσης ενέργειας και το αποτέλεσμα αποθηκεύεται σε ένα ξεχωριστό αρχείο με την ονομασία 'household_power_consumption_days.csv'.

7.1.1.5 Τρόπος αξιολόγησης

Μια πρόβλεψη θα αποτελείται από επτά τιμές, μία για κάθε ημέρα της επόμενης εβδομάδας.

Οι μονάδες μέτρησης της συνολικής ισχύος είναι kilowatts και είναι χρήσιμο να υπάρχει μια μέτρηση σφάλματος με τις ίδιες μονάδες μέτρησης. Και το Root Mean Squared (RMSE) και το Mean Absolut Error (MAE) ταιριάζουν στο πρόβλημά μας. Σε αντίθεση με το MAE, το RMSE είναι περισσότερο αυστηρό στην απόδοση σφαλμάτων πρόβλεψης και αυτό θα χρησιμοποιηθεί στην ανάλυσή μας.

Η μέτρηση απόδοσης για αυτό το πρόβλημα θα είναι το RMSE για κάθε επόμενη ώρα από την ημέρα 1 έως την ημέρα 7. Επιπλέον, θα συνοψίσουμε την απόδοση του μοντέλου χρησιμοποιώντας ένα

μοναδικό σκορ προκειμένου να βοηθηθούμε στην επιλογή του καλύτερου μοντέλου.

Η συνάρτηση evaluation_forecasts() εφαρμόζει αυτήν τη συμπεριφορά και θα επιστρέφει την απόδοση ενός μοντέλου βάσει πολλαπλών προβλέψεων για τις επτά ημέρες. Αυτή επιστρέφει πρώτα το συνολικό RMSE ανεξάρτητα από την ημέρα και κατόπιν ένα πίνακα με RMSE τιμές για κάθε ημέρα που προβλέφθηκε.

7.1.1.6 Train και Test Datasets

Θα χρησιμοποιήσουμε τα πρώτα τρία χρόνια δεδομένων για την εκπαίδευση των προγνωστικών μοντέλων (train dataset) και το τελευταίο έτος για την αξιολόγηση των μοντέλων αυτών (test dataset).

Το σύνολο των δεδομένων θα διαιρεθεί σε τυπικές εβδομάδες, όπου ξεκινούν την Κυριακή και τελειώνουν το Σάββατο.

Αρχικά, χωρίζουμε τα δεδομένα σε ολόκληρες εβδομάδες, δουλεύοντας προς τα πίσω από το test dataset.

Το τελευταίο έτος των δεδομένων είναι το 2010 και η πρώτη Κυριακή για το 2010 ήταν η 3η Ιανουαρίου. Τα δεδομένα τελειώνουν στα μέσα Νοεμβρίου του 2010 και το τελευταίο Σάββατο στα πιο πρόσφατα δεδομένα είναι η 20η Νοεμβρίου. Αυτό δίνει 46 εβδομάδες στο test dataset.

Η πρώτη και τελευταία σειρά ημερήσιων δεδομένων για το test dataset επομένως είναι:

2010-01-03, 2083.4539999999984, 191.61000000000055, 350992.12000000034, 8703.600000000033, 3842.0, 4920.0, 10074.0, 15888.233355799992

...

2010-11-20, 2197.006000000004, 153.76800000000028, 346475.9999999998, 9320.20000000002, 4367.0, 2947.0, 11433.0, 17869.76663959999

Τα δεδομένα ξεκινάνε στα τέλη του 2006. Η πρώτη Κυριακή στο σύνολο δεδομένων είναι η 17η Δεκεμβρίου, η οποία είναι η δεύτερη σειρά δεδομένων.

Η οργάνωση των δεδομένων σε ολόκληρες εβδομάδες δίνει 159 πλήρεις εβδομάδες για το training ενός προγνωστικού μοντέλου.

2006-12-17, 3390.46,226.0059999999994, 345725.32000000024, 14398.59999999998, 2033.0, 4187.0, 13341.0, 36946.66673200004

...

2010-01-02, 1309.2679999999998, 199.54600000000016, 352332.8399999997, 5489.7999999999865, 801.0, 298.0, 6425.0, 14297.133406600002

Η συνάρτηση split_dataset() χωρίζει τα ημερήσια δεδομένα σε training και testing datasets οργανώνοντάς τα σε ολόκληρες εβδομάδες.

Η εκτέλεσή του πλήρη κώδικα, δείχνει ότι πράγματι το training dataset έχει 159 εβδομάδες δεδομένων, ενώ το testing έχει 46 εβδομάδες.

(159, 7, 8)

3390.46 1309.2679999999998

(46, 7, 8)

2083.4539999999984 2197.006000000004

7.1.1.7 Walk-forward Validation

Τα μοντέλα θα αξιολογηθούν χρησιμοποιώντας ένα σχήμα που ονομάζεται walk-forward validation.

Συγκεκριμένα, όταν ένα μοντέλο απαιτείται για να κάνει μια πρόβλεψη μιας εβδομάδας, στη συνέχεια, τα πραγματικά δεδομένα για εκείνη την εβδομάδα είναι διαθέσιμα στο μοντέλο, έτσι ώστε να μπορεί να χρησιμοποιηθεί ως βάση για την πρόβλεψη της επόμενης εβδομάδας.

Μπορούμε να το καταδείξουμε παρακάτω με διαχωρισμό των δεδομένων εισόδου και των δεδομένων εξόδου/ των προβλεπόμενων δεδομένων.

Είσοδος,	Πρόβλεψη
[Εβδομάδα1]	Εβδομάδα2
[Εβδομάδα1 + Εβδομάδα2]	Εβδομάδα3
[Εβδομάδα1 + Εβδομάδα2 + Εβδομάδα3]	Εβδομάδα4

...

Η walk-forward αξιολόγηση των προγνωστικών μοντέλων για αυτό το σύνολο δεδομένων γίνεται μέσω της συνάρτησης evaluate_model().

Το όνομα μιας συνάρτησης παρέχεται στο μοντέλο ως το όρισμα "model_func". Αυτή η συνάρτηση είναι υπεύθυνη για τον καθορισμό του

μοντέλου, το fitting του μοντέλου στο training dataset και για τη πρόβλεψη μιας εβδομάδας.

Οι προβλέψεις που έγιναν από το μοντέλο στη συνέχεια αξιολογούνται με βάση το testing dataset χρησιμοποιώντας τη συνάρτηση evaluated_forecasts() που ορίστηκε προηγουμένως.

Τέλος μόλις έχουμε την αξιολόγηση για ένα μοντέλο, μπορούμε να συνοψίσουμε την απόδοση. Η συνάρτηση summarize_scores() τυπώνει την απόδοση ενός μοντέλου σε μία γραμμή για εύκολη σύγκριση με άλλα μοντέλα.

Τώρα έχουμε όλα τα στοιχεία για να ξεκινήσουμε την αξιολόγηση προγνωστικών μοντέλων στο σύνολο δεδομένων.

7.1.2 ARIMA

7.1.2.1 Autocorrelation Analysis

Η στατιστική συσχέτιση συνοψίζει τη δύναμη της σχέσης μεταξύ δύο μεταβλητών. Υποθέτουμε ότι η κατανομή κάθε μεταβλητής ταιριάζει είναι μια γκαουσιανή κατανομή. Αν συμβαίνει αυτό, μπορούμε να χρησιμοποιήσουμε τον συντελεστή συσχέτισης του Pearson για να βρούμε τη συσχέτιση μεταξύ των μεταβλητών μας.

Ο συντελεστής συσχέτισης του Pearson είναι ένας αριθμός μεταξύ -1 και 1 ο οποίος περιγράφει έναν αρνητικό ή θετικό συσχετισμό αντίστοιχα. Η τιμή μηδέν δείχνει μηδενική συσχέτιση.

Μπορούμε να υπολογίσουμε τη συσχέτιση για τις παρατηρήσεις των χρονοσειρών με παρατηρήσεις σε προηγούμενα βήματα χρόνου, που ονομάζονται υστερήσεις. Επειδή η συσχέτιση των παρατηρήσεων της χρονοσειράς υπολογίζεται με τιμές της ίδιας σειράς σε προηγούμενους χρόνους, αυτή ονομάζεται σειριακή συσχέτιση ή αυτοσυσχέτιση.

Η γραφική παράσταση της αυτοσυσχέτισης μιας χρονικής σειράς με χρονική υστέρηση ονομάζεται AutoCorrelation Function (ACF). Το γράφημα αυτό ονομάζεται μερικές φορές correlogram ή ένα autocorrelation γράφημα.

Η Partial AutoCorrelation Function (PACF) είναι μια σύνοψη της σχέσης μεταξύ μιας παρατήρησης σε μια χρονοσειρά με παρατηρήσεις σε προηγούμενα βήματα, με τις σχέσεις των παρεμβαλλόμενων παρατηρήσεων να αφαιρούνται.

Η αυτοσυσχέτιση για μια παρατήρηση και μια παρατήρηση σε ένα βήμα προγενέστερου χρόνου καθορίζεται τόσο από την άμεση συσχέτιση όσο και από τις έμμεσες συσχετίσεις. Αυτές οι έμμεσες συσχετίσεις είναι μια γραμμική συνάρτηση της συσχέτισης της παρατήρησης, με παρατηρήσεις στα παρεμβαλλόμενα χρονικά βήματα. Είναι αυτοί οι έμμεσοι συσχετισμοί που επιδιώκει να αφαιρέσει η PACF.

Μπορούμε να υπολογίσουμε τις ACF και PACF χρησιμοποιώντας τις συναρτήσεις plot_acf() και plot_pacf() αντίστοιχα της βιβλιοθήκης statsmodels.

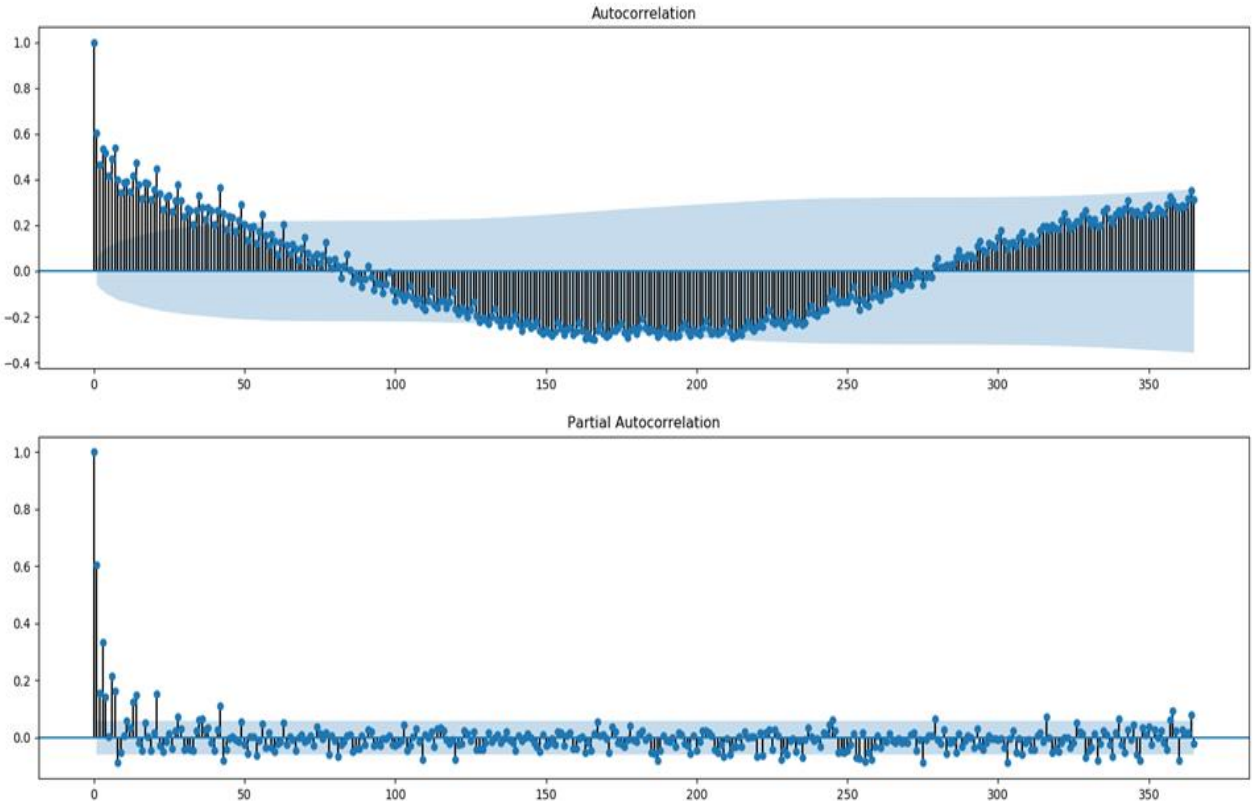
Για να υπολογίσουμε και να σχεδιάσουμε την αυτοσυσχέτιση, πρέπει να μετατρέψουμε τα δεδομένα σε μια μόνο μεταβλητή/χρονοσειρά. Συγκεκριμένα, τη συνολική ημερήσια ισχύς που καταναλώνεται.

Η συνάρτηση to_series() θα λάβει τα δεδομένα πολλών μεταβλητών χωρισμένα σε εβδομαδιαία παράθυρα και θα επιστρέψει μια ενιαία χρονοσειρά.

Μπορούμε να καλέσουμε αυτή τη συνάρτηση για το training dataset. Μια απλή χρονοσειρά της καθημερινής κατανάλωσης ισχύος μπορεί στη συνέχεια να εξαχθεί από το training dataset.

Στη συνέχεια, δημιουργούμε ένα ενιαίο σχήμα που περιέχει τόσο ένα ACF γράφημα όσο και ένα γράφημα PACF. Καθορίζεται ο αριθμός των χρονικών βημάτων υστέρησης για ένα έτος καθημερινών παρατηρήσεων ή 365 ημερών.

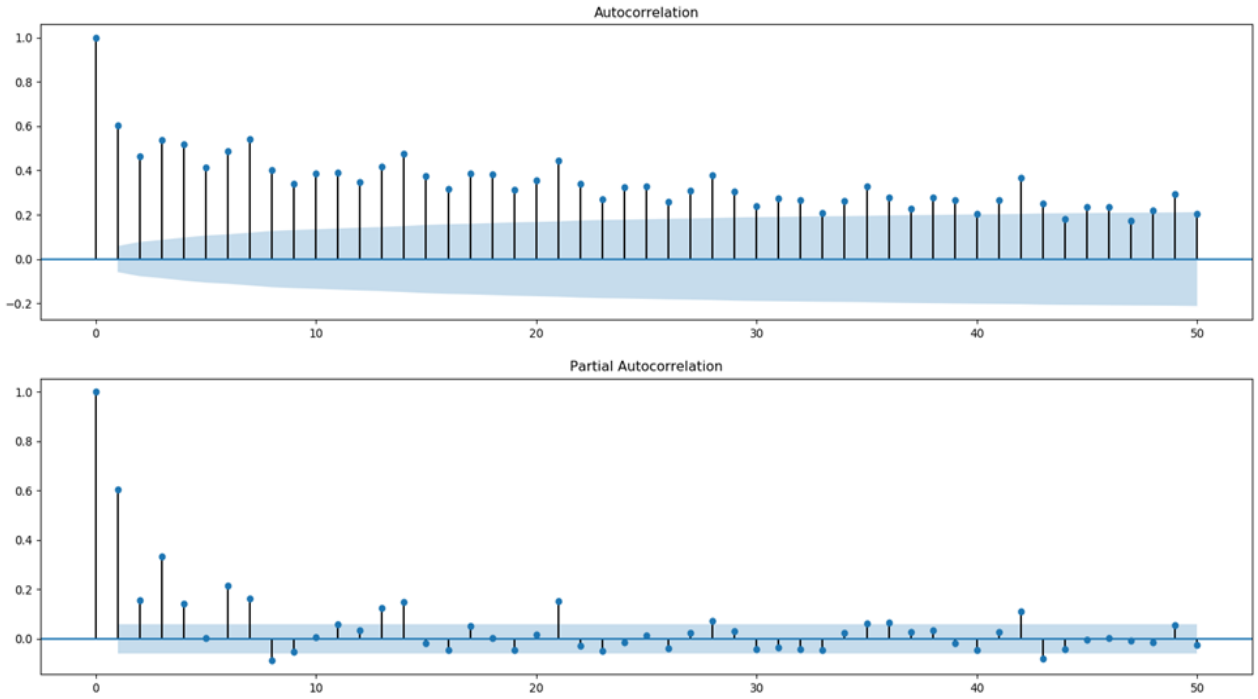
Αναμένουμε ότι η ενέργεια που καταναλώνεται αύριο και την προσεχή εβδομάδα θα εξαρτηθεί από την ισχύ που καταναλώνεται τις προηγούμενες ημέρες. Ως εκ τούτου, αναμένουμε να δούμε ένα ισχυρό σήμα αυτοσυσχέτισης στα γραφήματα ACF και PACF.



Εικόνα 25: Οι γραφικές παραστάσεις των ACF και PACF με lag 365

Τα γραφήματα είναι πυκνά και η ανάγνωσή τους παραμένει δύσκολη. Ωστόσο θα μπορούσαμε να εντοπίσουμε ένα μοτίβο αυτοσυσχέτισης και στα δύο.

Μπορούμε επίσης να δούμε ορισμένες σημαντικές υστερήσεις στις παρατηρήσεις σε ένα ολόκληρο χρόνο. Σίγουρα θα έχουμε και εποχιακές συσχετίσεις. Μπορούμε να μεγεθύνουμε το γράφημα και να αλλάξουμε το lag των παρατηρήσεων από 365 σε 50.



Εικόνα 26: Οι γραφικές παραστάσεις των ACF και PACF με lag 50

Η εκ νέου εκτέλεση του κώδικα με αυτή την αλλαγή είναι μια μεγεθυμένη έκδοση των γραφημάτων με πολύ λιγότερη ακαταστασία.

Βλέπουμε ένα οικείο μοτίβο αυτοσυσχέτισης και στα δύο γραφήματα. Αυτό το μοτίβο αποτελείται από δύο στοιχεία:

ACF: Ένας μεγάλος αριθμός σημαντικών παρατηρήσεων υστέρησης που υποβαθμίζονται αργά καθώς αυξάνεται η υστέρηση.

PACF: Μερικές σημαντικές παρατηρήσεις υστέρησης που πέφτουν απότομα καθώς αυξάνεται η υστέρηση.

Η γραφική παράσταση ACF υποδεικνύει ότι υπάρχει ένα ισχυρό συστατικό αυτοσυσχέτισης, ενώ η γραφική παράσταση PACF υποδεικνύει ότι αυτό το συστατικό είναι διακριτό για τις πρώτες περίπου επτά παρατηρήσεις υστερήσεων.

7.1.2.2 Ανάπτυξη Autoregression Μοντέλου

Μπορούμε να αναπτύξουμε ένα autoregression μοντέλο για μια ενιαία χρονοσειρά καθημερινής κατανάλωσης ρεύματος.

Η βιβλιοθήκη Statsmodels παρέχει πολλούς τρόπους για την ανάπτυξη ενός μοντέλου AR, όπως η χρήση των κλάσεων AR, ARMA, ARIMA και SARIMAX.

Θα χρησιμοποιήσουμε την κλάση ARIMA, καθώς επιτρέπει την εύκολη επεκτασιμότητα στη μέθοδο των διαφορών (differencing) και κινητών μέσων (moving average).

Αρχικά, το σύνολο των δεδομένων οργανώνεται σε εβδομάδες προηγούμενων παρατηρήσεων. Αυτό πρέπει να μετατραπεί σε χρονοσειρά μιας μόνο μεταβλητής που θα αφορά την ημερήσια κατανάλωση ενέργειας. Για αυτό το σκοπό, χρησιμοποιούμε τη συνάρτηση to_series() που αναπτύχθηκε προηγουμένως.

Στη συνέχεια, ένα μοντέλο ARIMA μπορεί να οριστεί εισάγοντας arguments στον constructor της κλάσης ARIMA. Θα προσδιορίσουμε ένα μοντέλο AR (7), το οποίο σημειώνεται να είναι ARIMA (7,0,0).

Στη συνέχεια, το μοντέλο μπορεί να προσαρμοστεί στο training dataset. Θα χρησιμοποιήσουμε τις προεπιλογές και θα απενεργοποιήσουμε όλες τις πληροφορίες εντοπισμού σφαλμάτων κατά τη διάρκεια της προσαρμογής ρυθμίζοντας το disp = False.

Μια πρόβλεψη μπορεί να γίνει καλώντας τη συνάρτηση predict() και μεταφέροντάς της είτε ένα διάστημα ημερομηνιών είτε δεικτών σε σχέση με το training dataset. Θα χρησιμοποιήσουμε δείκτες που ξεκινούν με πρώτο βήμα παραπάνω από το training dataset και θα το επεκτείνουν για έξι ακόμη ημέρες, δίνοντας συνολικά μια περίοδο προβλέψεων διάρκειας επτά ημερών.

Όλα αυτά τα πραγματοποιούμε σε μια συνάρτηση με όνομα arima_forecast() που παίρνει τη χρονοσειρά και επιστρέφει μια πρόβλεψη μιας εβδομάδας.

Αναπτύχθηκαν και άλλα ARIMA μοντέλα με τα τελικά αποτελέσματα να παρουσιάζονται στον παρακάτω πίνακα:

Πίνακας 1. Απόδοση ARIMA μοντέλων

Model	RMSE
ARIMA (3,0,0)	394.464
ARIMA (4,0,0)	386.186
ARIMA (5,0,0)	385.849
ARIMA (6,0,0)	379.765
ARIMA (7,0,0)	381.614

Η εκτέλεση του προγράμματος εκτυπώνει την απόδοση του AR (6) στο σύνολο δεδομένων δοκιμής.

Βλέπουμε ότι το μοντέλο επιτυγχάνει συνολικό RMSE 379.765, το καλύτερο σκορ από τα εξεταζόμενα μοντέλα.

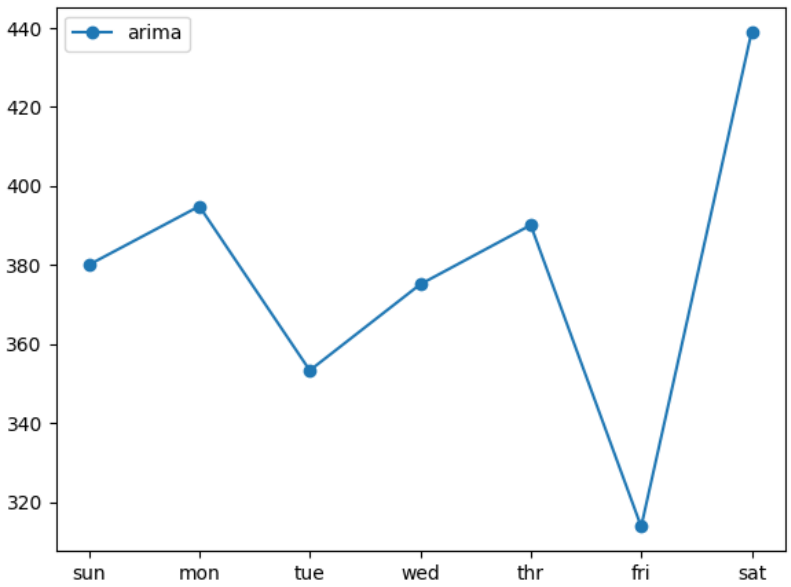
Πίνακας 2. Απόδοση ARIMA(6,0,0)

Mean RMSE	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
379.765	380.1	394.9	353.4	375.2	390.2	313.9	439.1

Επίσης δημιουργείται μια γραφική παράσταση της πρόβλεψης, η οποία δείχνει το RMSE σε kilowatt για καθέναν από τους επτά χρόνους παράδοσης της πρόβλεψης.

Αναμένεται οι αρχικοί χρόνοι πρόβλεψης είναι ευκολότεροι να προβλεφθούν από τους μεταγενέστερους , καθώς το σφάλμα σε κάθε διαδοχική χρονική περίοδο συνθέτει.

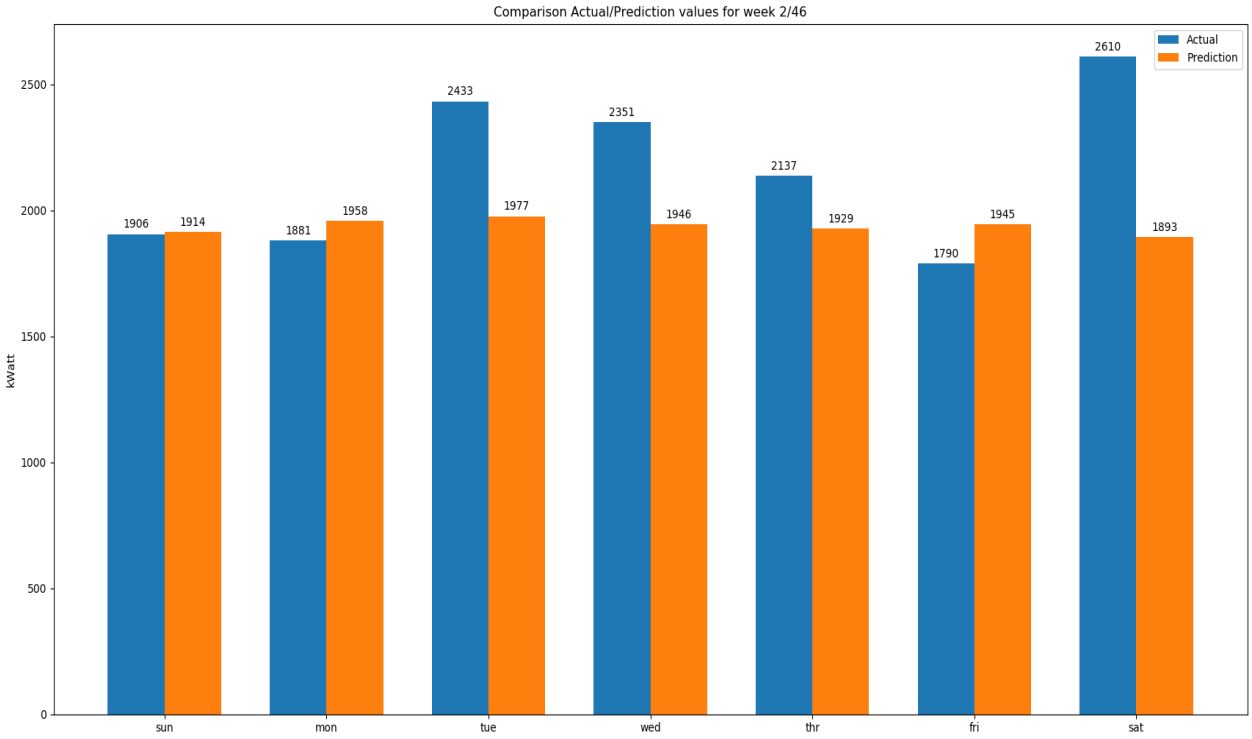
Αντίθετα, βλέπουμε ότι η Παρασκευή (lead time +6) είναι η ευκολότερη πρόβλεψη και το Σάββατο (lead time +7) είναι η πιο δύσκολη πρόβλεψη. Μπορούμε επίσης να διαπιστώσουμε ότι οι υπόλοιποι χρόνοι πρόβλεψης έχουν παρόμοιο σφάλμα στην περιοχή των 300 - 440 kilowatt.



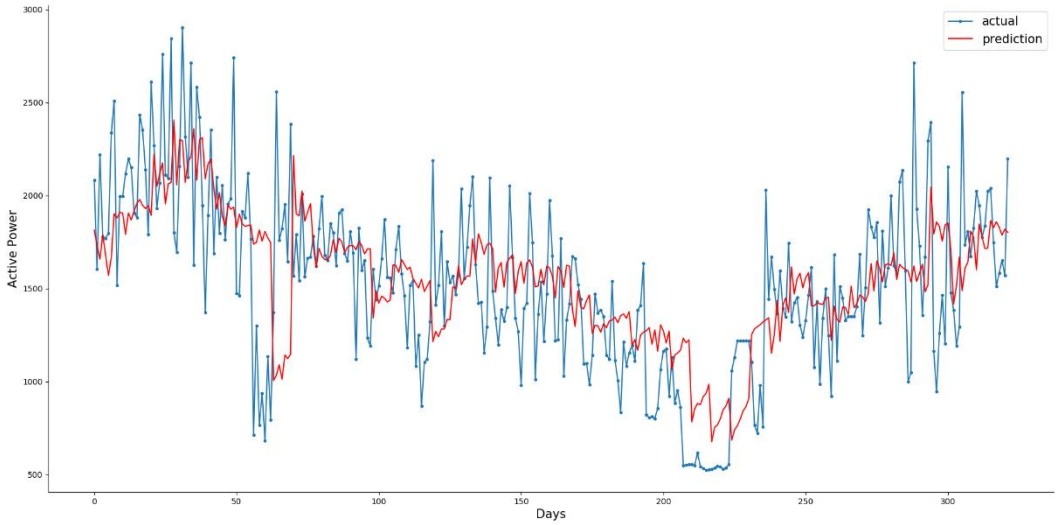
Εικόνα 27: Γραφική παράσταση των επιδόσεων του βέλτιστου ARIMA μοντέλου για το dataset

Ακολουθεί μια σύγκριση των προβλεπόμενων τιμών με τις πραγματικές για μια ενδεικτική εβδομάδα. Οι διαφορές τους ποικίλουν από τρέξιμο σε

τρέξιμο και από εβδομάδα σε εβδομάδα. Μια πιο συνολική εικόνα έχουμε στην Εικόνα 29 όπου παρουσιάζεται μια σύγκριση των προβλεπόμενων τιμών με τις πραγματικές για ολόκληρο το testing dataset.

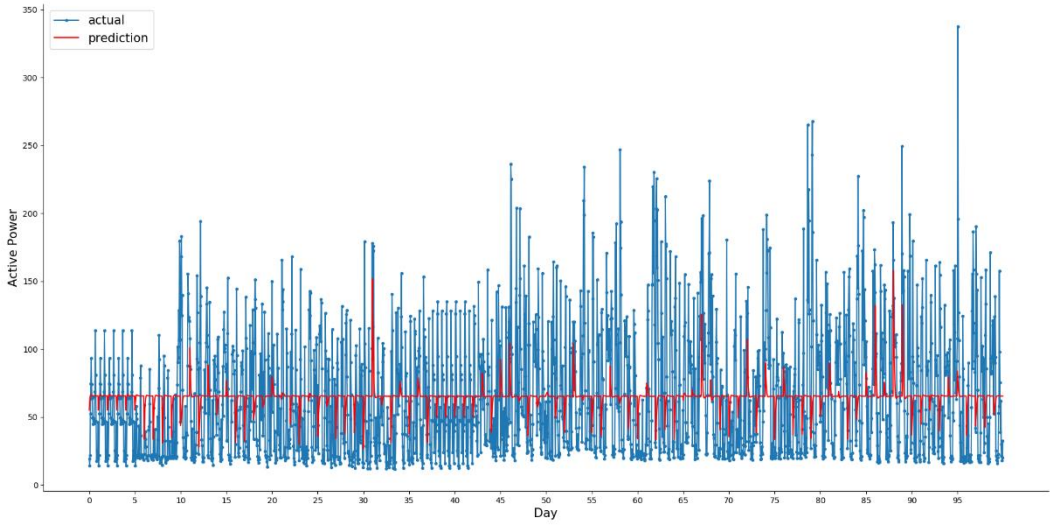


Εικόνα 28: Σύγκριση των πραγματικών και των προβλεπόμενων τιμών του βέλτιστου ARIMA μοντέλου για το dataset, για τη δεύτερη από τις 46 εβδομάδες του testing dataset



Εικόνα 29: Σύγκριση των πραγματικών και των προβλεπόμενων τιμών του βέλτιστου ARIMA μοντέλου για ολόκληρο το testing dataset

Το αρχικό dataset ήταν σε λεπτά. Το τροποποιήσαμε σε ημέρες παίρνοντας το μέσο όρο αυτών των τιμών για να εξετάσουμε σε μια βδομάδα την επίδοση του μοντέλου. Θα μπορούσαμε να εξετάσουμε την επίδοση του μοντέλου τροποποιώντας το αρχικό dataset σε ώρες αντί για μέρες. Παρακάτω φαίνεται η σύγκριση των προβλεπόμενων και των πραγματικών τιμών του dataset.



Εικόνα 30: Σύγκριση των τιμών των πραγματικών και των προβλεπόμενων τιμών του βέλτιστου ARIMA μοντέλου για ολόκληρο το testing dataset τροποποιημένου σε ώρες

7.1.3 LSTMs

Τα RNNs, όπως έχουμε αναλύσει προηγουμένως, είναι ειδικά σχεδιασμένα για να δουλεύουν, να μαθαίνουν και να προβλέπουν δεδομένα αλληλουχίας.

Υπενθυμίζουμε, ένα RNN είναι ένα νευρικό δίκτυο όπου η έξοδος του δικτύου από ένα χρονικό βήμα παρέχεται ως είσοδος στο επόμενο βήμα του χρόνου. Αυτό επιτρέπει στο μοντέλο να αποφασίζει για το τι να προβλέψει με βάση τόσο την είσοδο για το τρέχον βήμα του χρόνου όσο και την άμεση γνώση του τι ήταν το αποτέλεσμα στο βήμα του προηγούμενου χρόνου. Το πιο επιτυχημένο και ευρέως χρησιμοποιούμενο RNN είναι το Long-Short Memory Network - LSTM. Είναι επιτυχές καθώς υπερνικά τις προκλήσεις που συνεπάγεται η εκπαίδευση ενός RNN. Εκτός από την αξιοποίηση της επαναλαμβανόμενης σύνδεσης των εξόδων από το προηγούμενο χρονικό βήμα, τα LSTMs έχουν επίσης μια εσωτερική μνήμη που λειτουργεί σαν μια τοπική μεταβλητή,

επιτρέποντάς τους να συσσωρεύουν κατάσταση πάνω από την ακολουθία εισόδου.

Τα LSTM προσφέρουν πολλά οφέλη όσον αφορά την πρόβλεψη χρονοσειρών πολλαπλών σταδίων. Αυτά είναι:

- Μητρική υποστήριξη για τις ακολουθίες. Τα LSTMs είναι ένας τύπος επαναλαμβανόμενου δικτύου και επομένως έχουν σχεδιαστεί για να λαμβάνουν δεδομένα αλληλουχίας ως εισροές, σε αντίθεση με άλλα μοντέλα όπου παρατηρήσεις υστέρησης πρέπει να παρουσιάζονται ως χαρακτηριστικά εισόδου.
- Εισροές πολλών μεταβλητών. Τα LSTMs υποστηρίζουν απευθείας πολλές παράλληλες ακολουθίες εισόδου για εισόδους πολλών μεταβλητών, σε αντίθεση με άλλα μοντέλα όπου οι είσοδοι πολλών μεταβλητών παρουσιάζονται σε επίπεδη δομή.
- Έξοδος διάνυσμα. Όπως και άλλα νευρωνικά δίκτυα, τα LSTMs είναι σε θέση να αντιστοιχίζουν δεδομένα εισόδου απευθείας σε ένα διάνυσμα εξόδου που μπορεί να αντιπροσωπεύει πολλά βήματα χρόνου εξόδου.

Επιπλέον, έχουν αναπτυχθεί εξειδικευμένες αρχιτεκτονικές που έχουν σχεδιαστεί ειδικά για sequence-to-sequence προβλέψεις. Η μέθοδος seq2seq είναι σχεδιασμένη να εκπαιδεύεται έτσι ώστε να παράγει ακολουθίες εξόδου, βάσει των ακολουθιών εισόδου που δέχεται.

Ένα παράδειγμα RNN αρχιτεκτονικής για προβλήματα seq2seq είναι ο κωδικοποιητής-αποκωδικοποιητής LSTM (encoder-decoder LSTM).

Ένας encoder-decoder LSTM είναι ένα μοντέλο που αποτελείται από δύο τμήματα: ο ένας αποκαλείται κωδικοποιητής που διαβάζει τις ακολουθίες εισόδου και τον συμπιέζει σε μια εσωτερική αναπαράσταση σταθερού μήκους και ένα μοντέλο εξόδου αποκαλούμενο αποκωδικοποιητής που ερμηνεύει την εσωτερική αναπαράσταση και το χρησιμοποιεί για την πρόβλεψη της ακολουθίας εξόδου.

Η προσέγγιση encoder-decoder στην πρόβλεψη ακολουθίας έχει αποδειχθεί αρκετά πιο αποτελεσματική από την άμεση παραγωγή ενός vector και είναι η προτιμώμενη προσέγγιση.

Γενικά, τα LSTMs έχουν βρεθεί ότι δεν είναι πιο αποτελεσματικά σε προβλήματα τύπου autoregression. Πρόκειται για προβλήματα όπου η πρόβλεψη του επόμενου βήματος είναι συνάρτηση των πρόσφατων βημάτων του χρόνου.

Τα μονοδιάστατα CNN έχουν αποδειχθεί αποτελεσματικά στην αυτόματη εκμάθηση χαρακτηριστικών από τις ακολουθίες εισόδου.

Μια δημοφιλής προσέγγιση συνδυάζει τα CNN με τα LSTM, όπου το CNN είναι ως ένας encoder που μάθει χαρακτηριστικά από τις υπο-ακολουθίες δεδομένων εισόδου, τα οποίες παρέχονται ως βήματα χρόνου σε ένα LSTM. Αυτή η αρχιτεκτονική ονομάζεται CNN-LSTM.

Μια παραλλαγή στην αρχιτεκτονική του CNN-LSTM είναι το ConvLSTM που χρησιμοποιεί την συνελικτική ανάγνωση των ακολουθιών εισόδου απευθείας μέσα στις μονάδες του LSTM. Αυτή η προσέγγιση έχει αποδειχθεί πολύ αποτελεσματική για την ταξινόμηση των χρονοσειρών και μπορεί να προσαρμοστεί για χρήση σε προβλέψεις χρονοσειρών πολλαπλών σταδίων.

Τα μοντέλα θα αναπτυχθούν και θα παρουσιαστούν στο πρόβλημα πρόβλεψης κατανάλωσης ηλεκτρικής ενέργειας του νοικοκυριού.

7.1.3.1 Univariate LSTM

Αναπτύσσουμε ένα απλό LSTM μοντέλο το οποίο διαβάζει σε μια ακολουθία τη συνολική ημερήσια κατανάλωση ενέργειας και προβλέπει μια έξοδο διανύσματος την ημερήσια κατανάλωση ενέργειας της επόμενης τυπικής εβδομάδας.

Αυτό το μοντέλο θα αποτελέσει τη βάση για τα πιο περίτεχνα μοντέλα που αναπτυχθούν σε επόμενα βήματα.

Ο αριθμός των προηγούμενων ημερών που χρησιμοποιήθηκαν ως είσοδος ορίζει την μονοδιάστατη (1D) υποακολουθία των δεδομένων που το LSTM θα διαβάσει και θα μάθει να εξάγει χαρακτηριστικά. Ως σημείο εκκίνησης θα χρησιμοποιήσουμε τις προηγούμενες επτά ημέρες.

Ένα μοντέλο LSTM αναμένει ότι τα δεδομένα έχουν το σχήμα:

[samples, timesteps, features]

Ένα δείγμα θα αποτελείται από επτά χρονικά βήματα με ένα χαρακτηριστικό για τις επτά ημέρες, αυτό της συνολικής ημερήσιας κατανάλωσης ενέργειας.

Το training dataset έχει 159 εβδομάδες δεδομένων, οπότε το σχήμα του training dataset θα είναι: *[159, 7, 1]*

Τα δεδομένα σε αυτή τη μορφή θα χρησιμοποιούσαν την προηγούμενη τυποποιημένη εβδομάδα για να προβλέψουν την επόμενη κανονική εβδομάδα. Ένα πρόβλημα είναι ότι 159 περιπτώσεις δεν είναι πολλές για την εκπαίδευση ενός νευρικού δικτύου.

Ένας τρόπος για να δημιουργήσουμε πολλά περισσότερα δεδομένα εκπαίδευσης είναι να αλλάξουμε το πρόβλημα κατά τη διάρκεια της

εκπαίδευσης για να προβλέψουμε τις επόμενες επτά ημέρες, λαμβάνοντας υπόψη τις προηγούμενες επτά ημέρες, ανεξάρτητα από την κανονική εβδομάδα.

Αυτό επηρεάζει μόνο τα training data και το πρόβλημα της δοκιμής παραμένει το ίδιο: να προβλέψουμε την ημερήσια κατανάλωση ενέργειας για την επόμενη κανονική εβδομάδα, δεδομένης της προηγούμενης εβδομάδας.

Αυτό θα απαιτήσει μικρή προετοιμασία των δεδομένων εκπαίδευσης.

Το training dataset δίνεται σε κανονικές εβδομάδες με οκτώ μεταβλητές, συγκεκριμένα στο σχήμα [159, 7, 8]. Το πρώτο βήμα είναι να αναδιοργανώσουμε τα δεδομένα έτσι ώστε να έχουμε οκτώ ακολουθίες χρονοσειρών.

Στη συνέχεια, πρέπει να επαναλάβουμε τα βήματα χρόνου και να διαιρέσουμε τα δεδομένα σε επικαλυπτόμενα παράθυρα. Κάθε επανάληψη κινείται κατά μήκος ενός χρονικού βήματος και προβλέπει τις επόμενες επτά ημέρες. Δηλαδή:

Input, Output

[d01, d02, d03, d04, d05, d06, d07], [d08, d09, d10, d11, d12, d13, d14]

[d02, d03, d04, d05, d06, d07, d08], [d09, d10, d11, d12, d13, d14, d15]

...

Μπορούμε να το κάνουμε αυτό παρακολουθώντας τους δείκτες έναρξης και τέλους για τις εισόδους και τις εξόδους καθώς επαναλαμβάνουμε σε όλο το μήκος των πεπλατυσμένων δεδομένων από την άποψη των χρονικών βημάτων.

Μπορούμε επίσης να το κάνουμε αυτό με τρόπο που ο αριθμός των εισόδων και εξόδων να παραμετροποιείται (π.χ. n_input, n_out).

Ορίζουμε μια συνάρτηση που ονομάζεται to_supervised(), η οποία λαμβάνει μια λίστα με εβδομάδες (history) και τον αριθμό των χρονικών βημάτων που χρησιμοποιούνται ως είσοδοι και έξοδοι και επιστρέφει τα δεδομένα στην μορφή αλληλεπικαλυπτόμενου κινούμενου παραθύρου.

Όταν τρέχουμε αυτή τη συνάρτηση σε ολόκληρο το training dataset, μετατρέπουμε τα 159 δείγματα σε 1099. συγκεκριμένα, το μετασχηματισμένο σύνολο δεδομένων έχει τα σχήματα: X=[1099,7,1] και y=[1099,7]. Στη συνέχεια, μπορούμε να ορίσουμε και να προσαρμόσουμε το LSTM μοντέλο στο training dataset.

Αυτό το πρόβλημα πρόβλεψης χρονοσειρών πολλών βημάτων είναι

autoregression. Δηλαδή, είναι πιθανόν να μοντελοποιείται καλύτερα εκεί όπου οι επόμενες επτά ημέρες έχουν κάποια συνάρτηση με τις παρατηρήσεις σε προηγούμενα βήματα. Επιπλέον, το γεγονός ότι έχουμε μια σχετικά μικρή ποσότητα δεδομένων σημαίνει ότι απαιτείται ένα μικρό μοντέλο.

Αναπτύσσουμε ένα μοντέλο με ένα μόνο κρυφό στρώμα LSTM με 200 units. Ο αριθμός των μονάδων στο κρυφό επίπεδο δεν σχετίζεται με τον αριθμό των χρονικών βημάτων στις ακολουθίες εισόδου. Το επίπεδο LSTM ακολουθείται από ένα πλήρως συνδεδεμένο στρώμα με 200 κόμβους που θα ερμηνεύσει τις λειτουργίες που έχουν αποκτηθεί από το LSTM στρώμα. Τέλος, ένα στρώμα εξόδου θα προβλέψει άμεσα ένα διάνυσμα με επτά στοιχεία, ένα για κάθε ημέρα στην ακολουθία εξόδου.

Επίσης, θα χρησιμοποιήσουμε τη mean squared error loss function, καθώς είναι μια καλή αντιστοιχία για την επιλεγμένη μετρική του RMSE. Θα χρησιμοποιήσουμε την αποδοτική εφαρμογή του Adam και θα εφαρμόσουμε το μοντέλο για 70 epochs με batch size 16.

Το μικρό batch size και η στοχαστική φύση του αλγορίθμου σημαίνουν ότι το ίδιο μοντέλο θα μάθει μια ελαφρώς διαφορετική αντιστοίχιση των εισροών στις εξόδους κάθε φορά που εκπαιδεύεται. Αυτό σημαίνει ότι τα αποτελέσματα μπορεί να διαφέρουν όταν αξιολογείται το μοντέλο. Μπορούμε να δοκιμάσουμε να τρέξουμε το μοντέλο πολλές φορές και να υπολογίσουμε έναν μέσο όρο απόδοσης του μοντέλου.

Ορίζουμε τη συνάρτηση build_model() που προετοιμάζει τα δεδομένα εκπαίδευσης, καθορίζει το μοντέλο και προσαρμόζει το μοντέλο στα training data, επιστρέφοντας το μοντέλο αυτό πλέον έτοιμο για πρόβλεψη.

Τώρα που γνωρίζουμε πώς να προσαρμόσουμε το μοντέλο, μπορούμε να δούμε πώς μπορεί να χρησιμοποιηθεί το μοντέλο για να κάνει μια πρόβλεψη.

Γενικά, το μοντέλο αναμένει ότι τα δεδομένα θα έχουν την ίδια τρισδιάστατη μορφή όταν κάνουν μια πρόβλεψη.

Στην περίπτωση αυτή, το αναμενόμενο σχήμα ενός μοτίβου εισόδου είναι ένα δείγμα, επτά ημέρες ενός χαρακτηριστικού για την ημερήσια κατανάλωση ενέργειας:

[1, 7, 1]

Τα δεδομένα πρέπει να έχουν αυτό το σχήμα όταν κάνουν προβλέψεις για το test dataset και όταν χρησιμοποιείται ένα τελικό μοντέλο για να κάνουν

προβλέψεις στο μέλλον. Εάν αλλάξουμε τον αριθμό των ημερών εισόδου σε 14, τότε το σχήμα του training dataset και το σχήμα των νέων δειγμάτων κατά τη διενέργεια των προβλέψεων πρέπει να τροποποιηθούν αναλόγως ώστε να έχουν 14 βήματα. Πρόκειται για μια επιλογή μοντέλου που πρέπει να μεταφέρουμε μπροστά όταν χρησιμοποιούμε το μοντέλο.

Χρησιμοποιούμε τη walk-forward validation για την αξιολόγηση του μοντέλου όπως περιγράφεται στην προηγούμενη ενότητα.

Αυτό σημαίνει ότι έχουμε τις διαθέσιμες παρατηρήσεις για την προηγούμενη εβδομάδα προκειμένου να προβλέψουμε την ερχόμενη εβδομάδα. Αυτά συγκεντρώνονται σε μια σειρά τυπικών εβδομάδων που ονομάζονται history.

Για να προβλέψουμε την επόμενη κανονική εβδομάδα, πρέπει να ανακτήσουμε τις τελευταίες ημέρες παρατηρήσεων. Όπως και με τα δεδομένα εκπαίδευσης, πρέπει πρώτα να τροποποιήσουμε τα history data για να αφαιρέσουμε την εβδομαδιαία δομή έτσι ώστε να καταλήξουμε σε οκτώ παράλληλες χρονοσειρές.

Στη συνέχεια, πρέπει να ανακτήσουμε τις τελευταίες επτά ημέρες της ημερήσιας συνολικής κατανάλωσης ενέργειας (δείκτης χαρακτηριστικών 0).

Θα το παραμετροποιήσουμε όπως κάναμε για τα training data, έτσι ώστε ο αριθμός των προηγούμενων ημερών που χρησιμοποιήθηκαν ως εισροές από το μοντέλο, να μπορεί να τροποποιηθεί στο μέλλον.

Στη συνέχεια, αναδιαμορφώνουμε την είσοδο στην αναμενόμενη τρισδιάστατη δομή.

Ορίζουμε τη συνάρτηση forecast() που υλοποιεί αυτό και παίρνει ως arguments το προσαρμοσμένο μοντέλο στο training dataset, το ιστορικό δεδομένων που παρατηρείται μέχρι τώρα και τον αριθμό των βημάτων χρόνου εισαγωγής που αναμένεται από το μοντέλο.

Η εκτέλεση του προγράμματος προσαρμόζει και αξιολογεί το μοντέλο, εκτυπώνει το συνολικό RMSE σε όλες τις επτά ημέρες και το RMSE ανά ημέρα για κάθε χρόνο παράδοσης.

Τα συγκεκριμένα αποτελέσματα μπορεί να διαφέρουν ανάλογα με την στοχαστική φύση του αλγορίθμου.

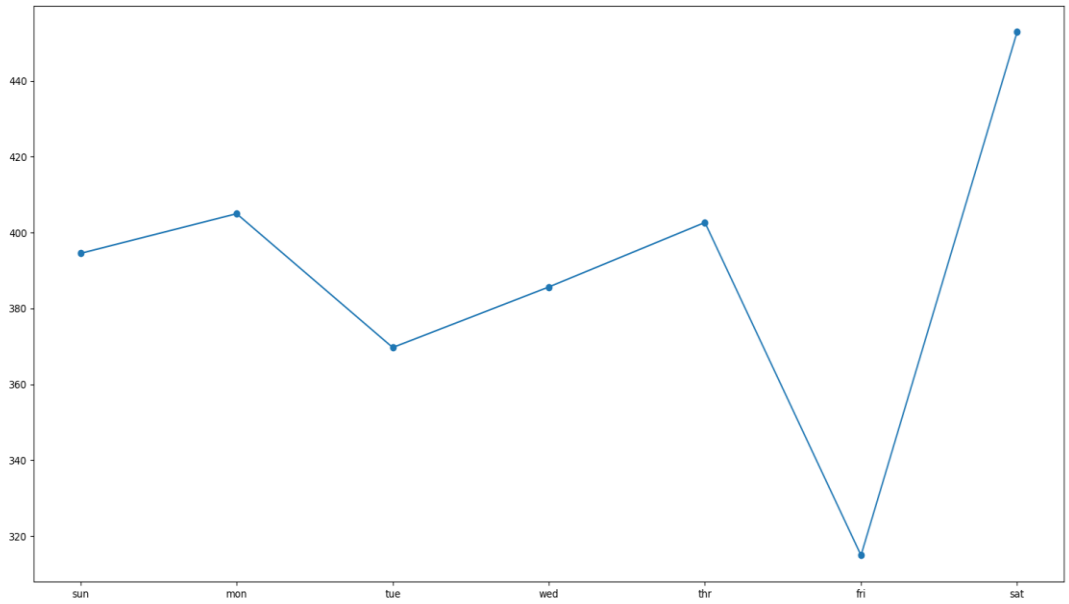
Κατά την εκτέλεση του μοντέλου αυτού βλέπουμε ότι το μοντέλο ήταν αποδοτικό με συνολικό RMSE περίπου 391 kilowatt.

Πίνακας 3. Επίδοση του univariate LSTM με είσοδο 7 ημέρες

Mean RMSE	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
391.256	394.5	405.0	369.7	385.7	402.7	314.9	452.9

Ένα διάγραμμα της ημερήσιας RMSE δημιουργείται επίσης.

Το διάγραμμα δείχνει ότι ίσως οι Τρίτες και οι Παρασκευές είναι πιο εύκολες ημέρες για πρόβλεψη από τις άλλες μέρες και ίσως το Σάββατο στο τέλος της κανονικής εβδομάδας είναι η πιο δύσκολη ημέρα για πρόβλεψη.



Εικόνα 31: Γραφική παράσταση των επιδόσεων του univariate LSTM μοντέλου για το dataset με είσοδο 7 ημερών

Μπορούμε να αυξήσουμε τον αριθμό των προηγούμενων ημερών για να χρησιμοποιήσουμε ως είσοδο 14 αντί για 7 με την αλλαγή της μεταβλητής n_input.

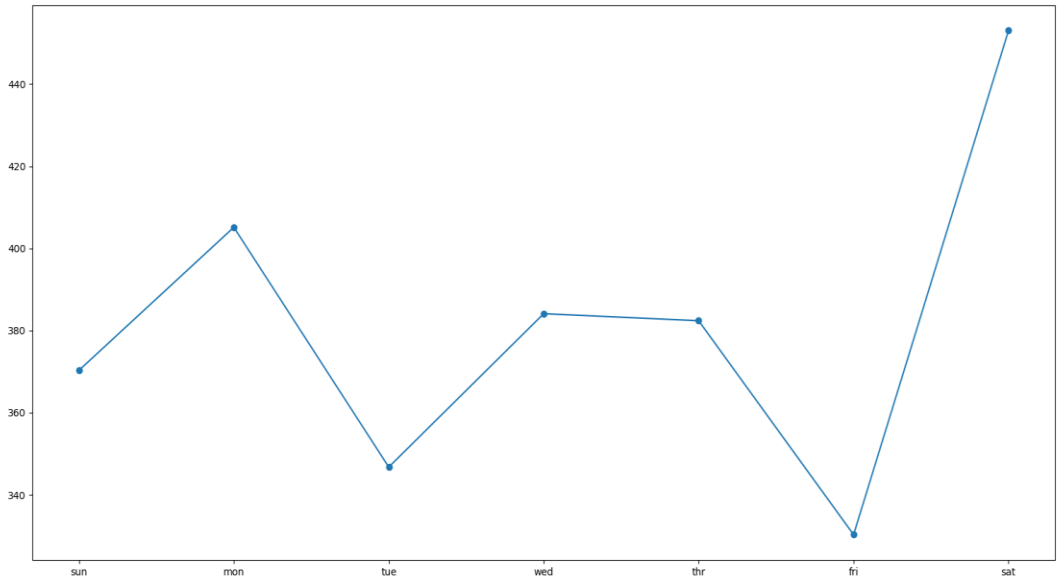
Σε αυτήν την περίπτωση, με την εκ νέου εκτέλεση του προγράμματος μπορούμε να δούμε μια περαιτέρω πτώση του συνολικού RMSE σε περίπου 383 kilowatt, υποδηλώνοντας ότι η περαιτέρω ρύθμιση του μεγέθους εισόδου και ίσως ο αριθμός κόμβων στο μοντέλο μπορεί να οδηγήσει σε καλύτερη απόδοση.

Πίνακας 4. Απόδοση univariate LSTM με είσοδο 14 ημέρες

Mean RMSE	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
383.512	370.3	405.1	346.8	384.1	382.4	330.3	453.0

Συγκρίνοντας τις τιμές RMSE ανά ημέρα, βλέπουμε ότι κάποιοι είναι καλύτεροι και μερικοί είναι χειρότεροι όταν χρησιμοποιούν εισροές επτά ημερών.

Αυτό μπορεί να υποδηλώνει όφελος στη χρήση των δύο διαφορετικών μεγεθών εισόδου με κάποιο τρόπο, όπως ένα σύνολο των δύο προσεγγίσεων ή ίσως ένα μόνο μοντέλο (π.χ. ένα μοντέλο πολλαπλών κεφαλών) που διαβάζει τα training data με διάφορους τρόπους.



Εικόνα 32: Γραφική παράσταση των επιδόσεων του univariate LSTM μοντέλου για το dataset με είσοδο 14 ημερών

Επαναλαμβάνουμε τη διαδικασία κάνοντας μικροαλλαγές στη δομή του μοντέλου. Τα αποτελέσματα παρουσιάζονται στον ακόλουθο πίνακα.

Πίνακας 5. Απόδοση univariate LSTM μοντέλων

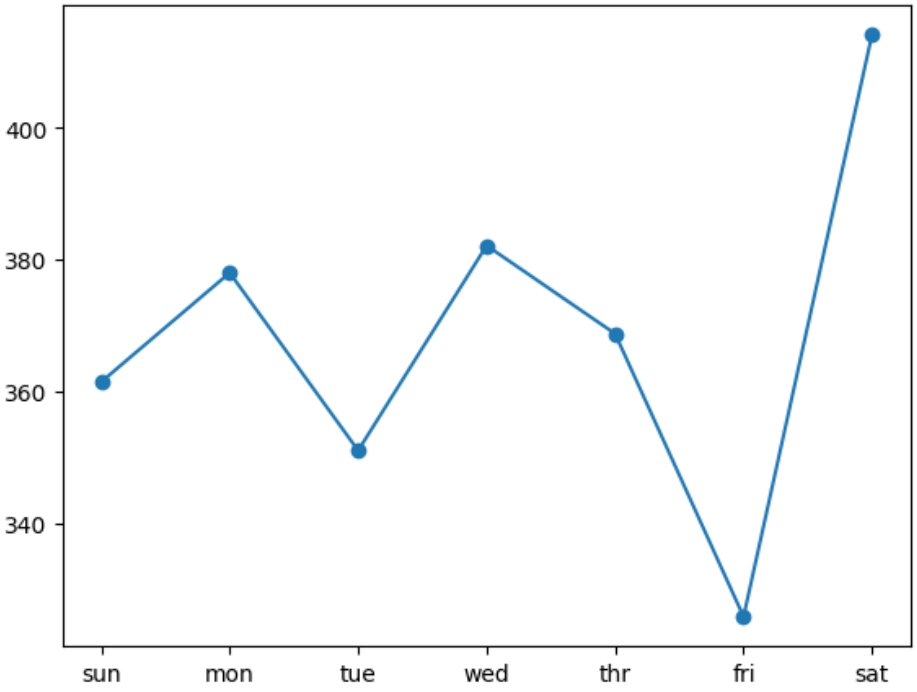
LSTM MODEL RESULTS							
inputs	7	14	14	14	14	14	14
Neurons	200	200	200	200	200	100	300
Epochs	70	70	70	70	70	70	70

Batch size	16	16	32	72	40	32	32
Verbose	0	0	0	0	0	0	0
Loss func	Mse	Mse	Mse	Mse	Mse	Mse	Mse
Optimizer	adam	adam	adam	adam	adam	adam	adam
Dense	100	100	100	100	100	100	100
RMSE	391.256	383.512	369.620	371.905	385.120	372.576	382.769

Βλέπουμε την καλύτερη απόδοση έχει το τρίτο μοντέλο του πίνακα με συνολικό RMSE 369.620. Αναλυτικά, οι επιδόσεις του έχουν ως εξής:

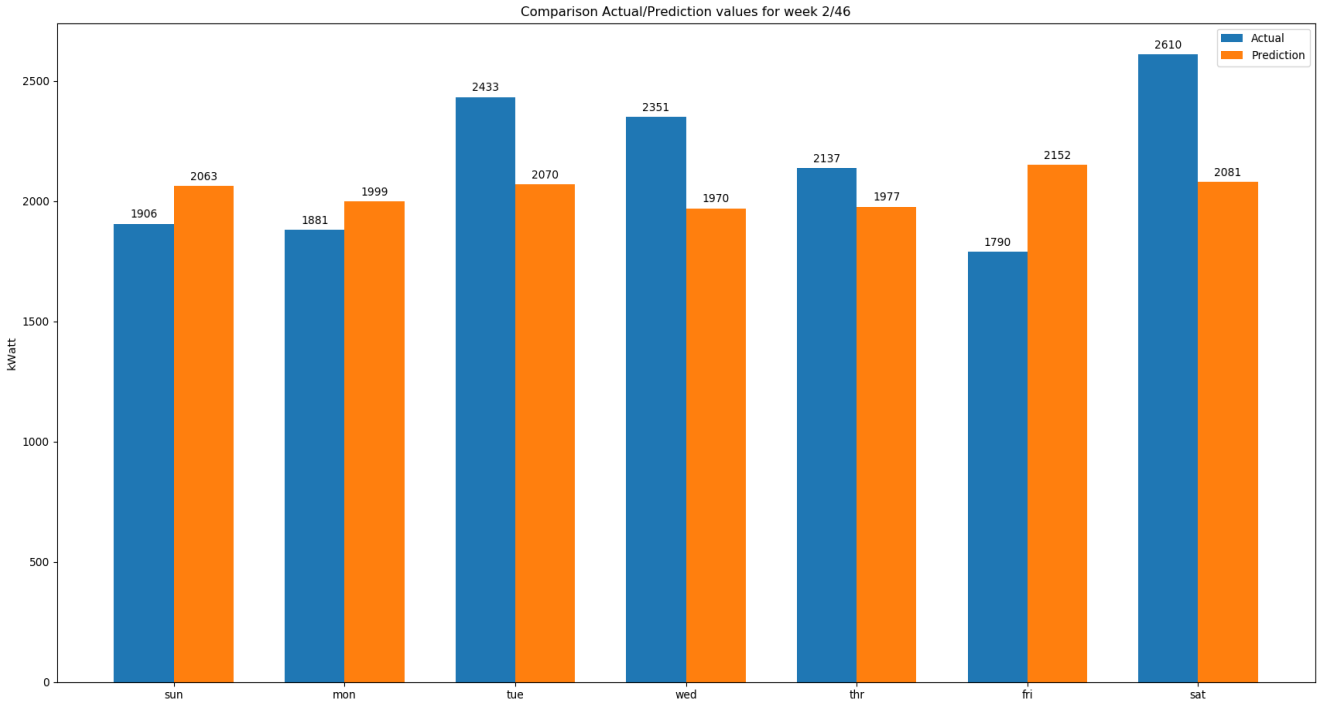
Πίνακας 6. Απόδοση βέλτιστου univariate LSTM με είσοδο 14 ημέρες

Mean RMSE	**Sunday**	**Monday**	**Tuesday**	**Wednesday**	**Thursday**	**Friday**	**Saturday**
369.620	361.4	378.0	351.0	382.1	368.7	325.8	414.1

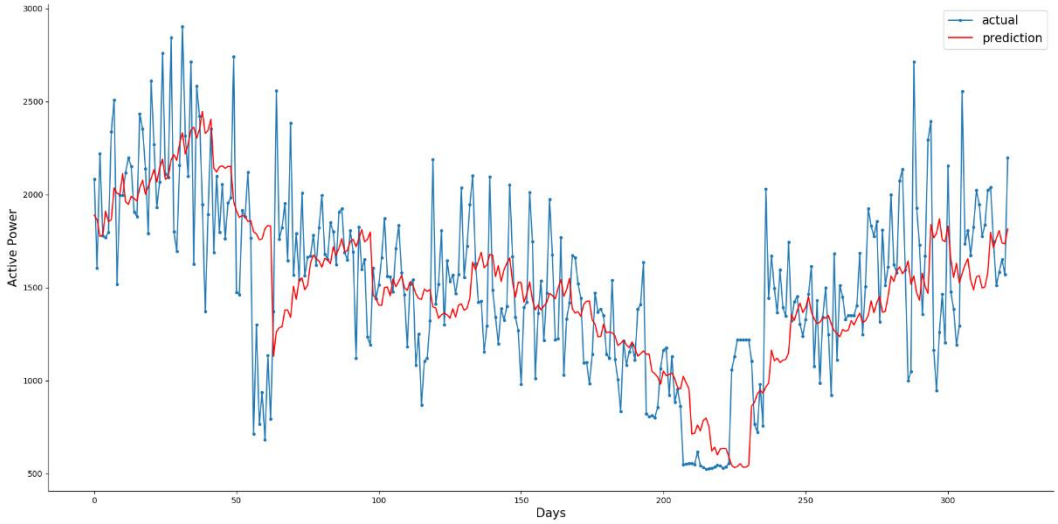


Εικόνα 33: Γραφική παράσταση των επιδόσεων του βέλτιστου univariate LSTM μοντέλου για το dataset

Ακολουθεί μια σύγκριση των προβλεπόμενων τιμών με τις πραγματικές για μια ενδεικτική εβδομάδα. Οι διαφορές τους ποικίλουν από τρέξιμο σε τρέξιμο και από εβδομάδα σε εβδομάδα. Μια πιο συνολική εικόνα έχουμε στην Εικόνα 35 όπου παρουσιάζεται μια σύγκριση των προβλεπόμενων τιμών με τις πραγματικές για ολόκληρο το testing dataset.



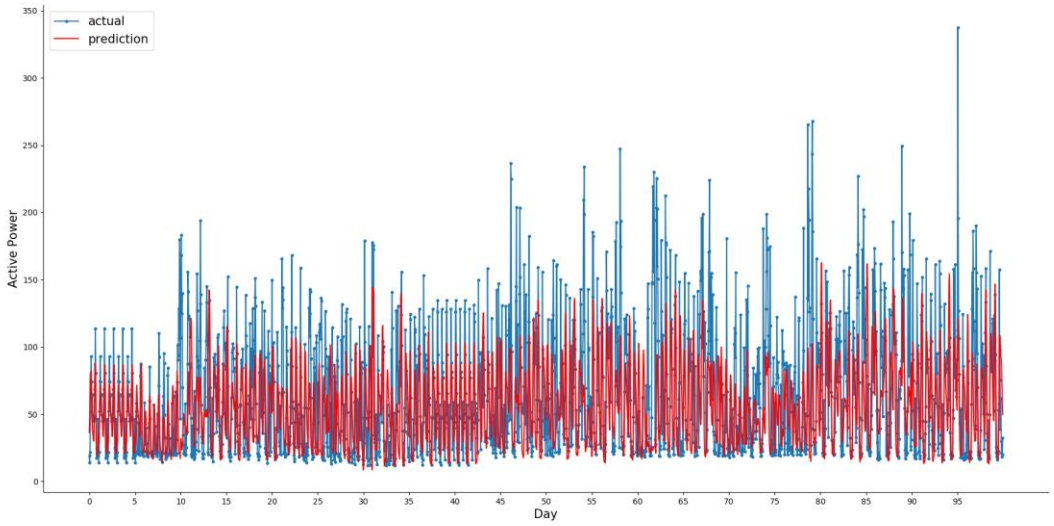
Εικόνα 34: Σύγκριση των πραγματικών και των προβλεπόμενων τιμών του βέλτιστου univariate LSTM μοντέλου για το dataset, για τη δεύτερη από τις 46 εβδομάδες του testing dataset



Εικόνα 35: Σύγκριση των πραγματικών και των προβλεπόμενων τιμών του βέλτιστου univariate LSTM μοντέλου για ολόκληρο το testing dataset

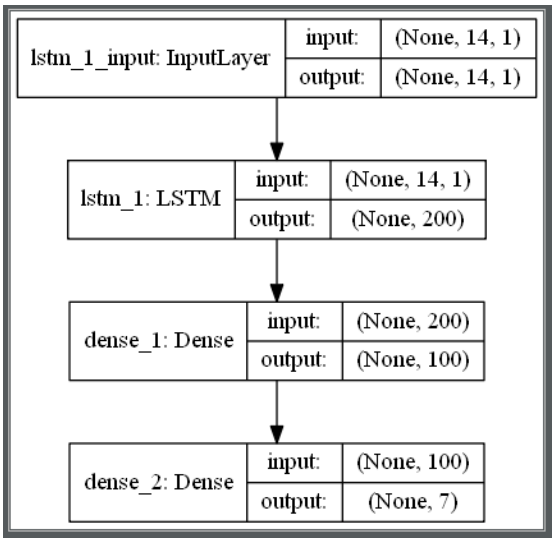
Το αρχικό dataset ήταν σε λεπτά. Το τροποποιήσαμε σε ημέρες παίρνοντας το μέσο όρο αυτών των τιμών για να εξετάσουμε σε μια βδομάδα την επίδοση του μοντέλου. Θα μπορούσαμε να εξετάσουμε την επίδοση του μοντέλου τροποποιώντας το αρχικό dataset σε ώρες αντί

για μέρες. Παρακάτω φαίνεται μια ελάχιστη καλύτερη σύγκριση των προβλεπόμενων και των πραγματικών τιμών, έχοντας χρησιμοποιήσει το 80% του dataset για training και το 20% για testing.



Εικόνα 37: Σύγκριση των πραγματικών και των προβλεπόμενων τιμών του βέλτιστου univariate LSTM μοντέλου για το ολόκληρο το testing dataset τροποποιημένου σε ώρες.

Τέλος, δημιουργούμε ένα σχήμα με την αποτύπωση της δομής του univariate LSTM μοντέλου μας.



Εικόνα 38: Σχηματική αναπαράσταση του univariate LSTM μοντέλου του προγράμματος (μέσω Keras)

7.1.3.2 Encoder – Decoder univariate LSTM

Σε αυτή την ενότητα θα ενημερώσουμε το univariate LSTM για να χρησιμοποιήσουμε ένα μοντέλο encoder-decoder.

Αυτό σημαίνει ότι το μοντέλο δεν θα δίνει άμεσα μια διανυσματική ακολουθία. Αντίθετα, το μοντέλο θα αποτελείται από δύο υπο-μοντέλα, τον κωδικοποιητή για την ανάγνωση και κωδικοποίηση της ακολουθίας εισόδου και τον αποκωδικοποιητή που θα διαβάζει την κωδικοποιημένη ακολουθία εισόδου και θα κάνει μια πρόβλεψη ενός σταδίου για κάθε στοιχείο της ακολουθίας εξόδου.

Η διαφορά των δύο μοντέλων είναι λεπτή, καθώς στην πράξη και οι δύο προσεγγίσεις προβλέπουν μιας ακολουθία ως έξοδο.

Αυτό που πρέπει να εστιάσουμε είναι ότι ένα μοντέλο LSTM χρησιμοποιείται στον αποκωδικοποιητή, επιτρέποντάς του να γνωρίζει και αυτό που είχε προβλεφθεί για την προηγούμενη ημέρα στην ακολουθία και να συσσωρεύσει την εσωτερική κατάσταση ενώ παράγει την ακολουθία.

Όπως και πριν, ορίζουμε ένα κρυφό στρώμα LSTM με 200 units. Αυτό είναι το μοντέλο αποκωδικοποιητή που θα διαβάσει την ακολουθία εισόδου και θα εξάγει ένα διάνυσμα 200 στοιχείων (μία έξοδο ανά unit) που καταγράφει χαρακτηριστικά από την ακολουθία εισόδου. Θα χρησιμοποιήσουμε 14 ημέρες συνολικής κατανάλωσης ενέργειας ως είσοδο.

Χρησιμοποιήσουμε μια απλή αρχιτεκτονική encoder-decoder, που είναι εύκολη στην εφαρμογή μέσω Keras, και η οποία είναι όμοια με την αρχιτεκτονική ενός autoencoder LSTM.

Πρώτον, η εσωτερική αναπαράσταση της ακολουθίας εισόδου επαναλαμβάνεται πολλές φορές, μία φορά για κάθε βήμα χρόνου στην ακολουθία εξόδου. Αυτή η διανυσματική ακολουθία θα παρουσιαστεί στον αποκωδικοποιητή LSTM.

Στη συνέχεια ορίζουμε τον αποκωδικοποιητή ως κρυφό στρώμα LSTM με 200 μονάδες. Είναι σημαντικό ότι ο αποκωδικοποιητής θα εξάγει ολόκληρη την ακολουθία, όχι μόνο το τέλος της ακολουθίας όπως κάναμε με τον κωδικοποιητή. Αυτό σημαίνει ότι καθένα από τα 200 units θα αποδώσει μια τιμή για κάθε μία από τις επτά ημέρες, που αντιπροσωπεύει τη βάση για το τι να προβλέψει για κάθε ημέρα στην ακολουθία εξόδου.

Στη συνέχεια θα χρησιμοποιήσουμε ένα πλήρως συνδεδεμένο στρώμα για να ερμηνεύσουμε κάθε βήμα της ακολουθίας εξόδου πριν από την τελική στρώση εξόδου. Είναι σημαντικό ότι η στρώση εξόδου προβλέπει

ένα μόνο βήμα στην ακολουθία εξόδου, όχι όλες τις επτά ημέρες κάθε φορά.

Αυτό σημαίνει ότι θα χρησιμοποιήσουμε τα ίδια στρώματα που εφαρμόζονται σε κάθε βήμα στην ακολουθία εξόδου. Δηλαδή, το ίδιο πλήρως συνδεδεμένο στρώμα και το στρώμα εξόδου θα χρησιμοποιηθούν για την επεξεργασία κάθε βήματος που παρέχεται από τον αποκωδικοποιητή. Για να το επιτύχουμε αυτό, θα τυλίξουμε το στρώμα ερμηνείας και το στρώμα εξόδου σε ένα TimeDistributed περιτύλιγμα που επιτρέπει την χρήση των τυλιγμένων στρωμάτων για κάθε βήμα του χρόνου από τον αποκωδικοποιητή.

Αυτό επιτρέπει στον αποκωδικοποιητή LSTM να καταλάβει το πλαίσιο που απαιτείται για κάθε βήμα της ακολουθίας εξόδου και τα περιτυλιγμένα πυκνά στρώματα για να ερμηνεύσει ξεχωριστά κάθε βήμα, αλλά επαναχρησιμοποιώντας τα ίδια βάρη για να εκτελέσει την ερμηνεία.

Συνεπώς, το δίκτυο εξάγει ένα τρισδιάστατο διάνυσμα με την ίδια δομή με αυτή της εισόδου, με τις διαστάσεις [δείγματα, χρονικά διαστήματα, χαρακτηριστικά].

Υπάρχει ένα ενιαίο χαρακτηριστικό, η ημερήσια συνολική ισχύς που καταναλώνεται και υπάρχουν πάντα επτά χαρακτηριστικά. Επομένως, μια ενιαία πρόβλεψη μιας εβδομάδας θα έχει το μέγεθος: [1, 7, 1].

Επομένως, κατά την εκπαίδευση του μοντέλου, πρέπει να αναδιαρθρώσουμε τα δεδομένα εξόδου (y) για να έχουμε την τρισδιάστατη δομή αντί για τη δισδιάστατη δομή των [δειγμάτων, χαρακτηριστικών] που χρησιμοποιήθηκαν στην προηγούμενη ενότητα.

Μπορούμε να συνδέσουμε όλα αυτά μαζί στην ενημερωμένη συνάρτηση build_model().

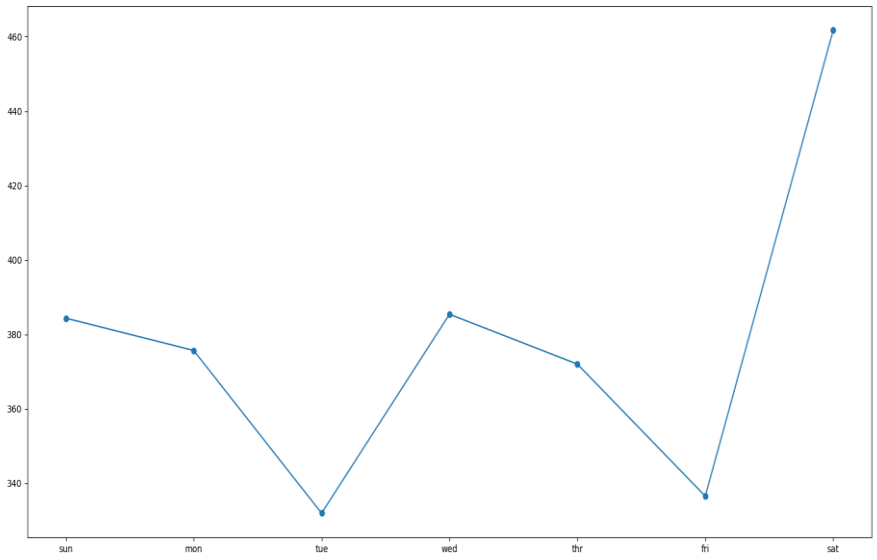
Η εκτέλεση του προγράμματος προσαρμόζει το μοντέλο και συνοψίζει την απόδοση στο test dataset.

Τα συγκεκριμένα αποτελέσματα μπορεί να διαφέρουν ανάλογα με την στοχαστική φύση του αλγορίθμου. Μπορούμε να δούμε ότι σε αυτή την περίπτωση, το μοντέλο είναι επιδέξιο, επιτυγχάνοντας συνολική βαθμολογία RMSE περίπου 380 kilowatt.

Πίνακας 6. Απόδοση Encoder-Decoder univariate LSTM μοντέλου

Mean RMSE	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
380.299	384.4	375.7	331.9	385.4	372.0	336.5	461.7

Επίσης δημιουργείται και ένα γραμμικό γράφημα του RMSE ανά ημέρα, που εμφανίζει παρόμοιο μοτίβο όπως και στην προηγούμενη ενότητα.



Εικόνα 39: Γραφική παράσταση των επιδόσεων του univariate EncoderDecoder-LSTM μοντέλου για το dataset

Επαναλαμβάνουμε τη διαδικασία κάνοντας μικροαλλαγές στη δομή του μοντέλου. Τα αποτελέσματα παρουσιάζονται στον ακόλουθο πίνακα.

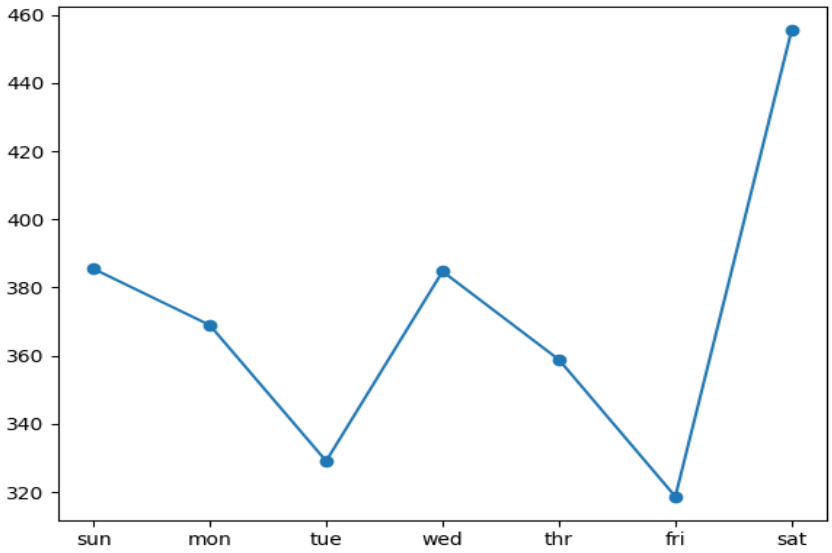
Πίνακας 7. Απόδοση EncoderDecoder-LSTM μοντέλων

LSTM MODEL RESULTS							
inputs	14	14	14	14	14	14	14
Neurons1	200	200	200	200	200	300	100
Neurons2	200	100	100	100	200	200	50
Neurons3	-	-	-	100	100	100	-
Epochs	20	20	50	20	20	20	20
Batch size	16	16	30	16	16	16	16
Verbose	0	0	0	0	0	0	0
Loss func	Mse	Mse	Mse	Mse	Mse	Mse	Mse
Optimizer	adam	adam	adam	adam	adam	adam	adam
Dense	100	100	100	100	100	100	100
Dense	1	1	1	1	1	1	1
RMSE	380.299	373.982	382.137	399.274	381.468	380.452	385.817

Βλέπουμε την καλύτερη απόδοση έχει το δεύτερο μοντέλο του πίνακα με συνολικό RMSE 373.982. Αναλυτικά, οι επιδόσεις του έχουν ως εξής:

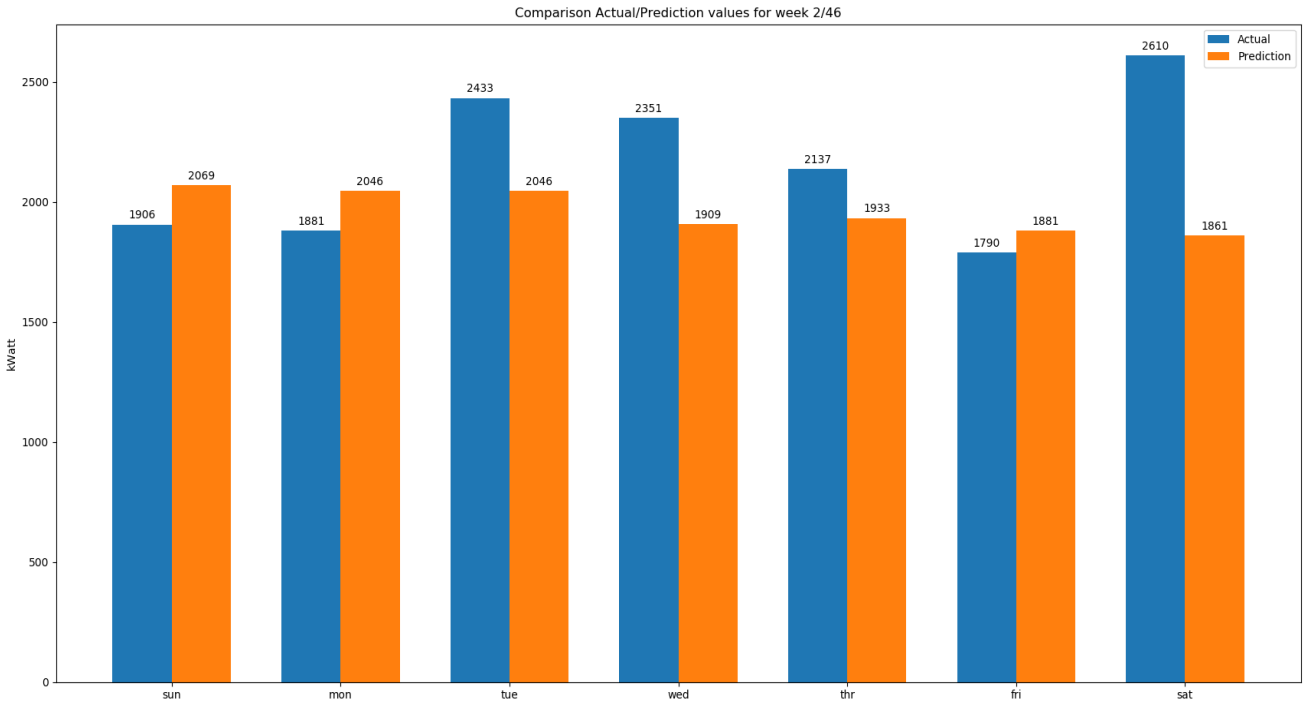
Πίνακας 8. Απόδοση βέλτιστου Encoder-Decoder univariate LSTM μοντέλου

Mean RMSE	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
373.982	385.5	369.0	329.1	384.7	358.9	318.7	455.6

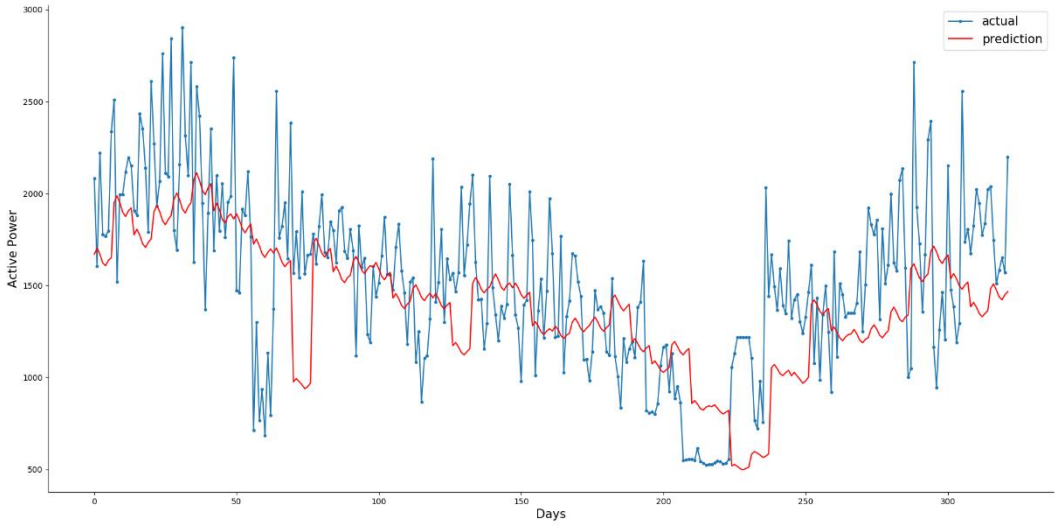


Εικόνα 40: Γραφική παράσταση των επιδόσεων του βέλτιστου univariate EncoderDecoder-LSTM μοντέλου για το dataset

Ακολουθεί μια σύγκριση των προβλεπόμενων τιμών με τις πραγματικές για μια ενδεικτική εβδομάδα. Οι διαφορές τους ποικίλουν από τρέξιμο σε τρέξιμο και από εβδομάδα σε εβδομάδα. Μια πιο συνολική εικόνα έχουμε στην Εικόνα 42 όπου παρουσιάζεται μια σύγκριση των προβλεπόμενων τιμών με τις πραγματικές για ολόκληρο το testing dataset.

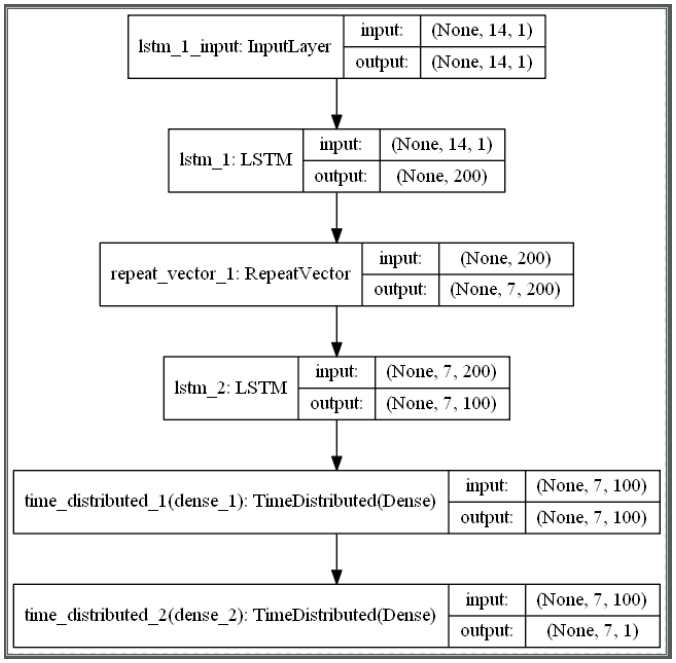


Εικόνα 41: Σύγκριση των πραγματικών και των προβλεπόμενων τιμών του βέλτιστου univariate Encoder-Decoder LSTM μοντέλου για το dataset, για τη δεύτερη από τις 46 εβδομάδες του testing dataset



Εικόνα 42: Σύγκριση των πραγματικών και των προβλεπόμενων τιμών του βέλτιστου univariate Encoder-Decoder LSTM μοντέλου για ολόκληρο το testing dataset

Τέλος, δημιουργούμε ένα σχήμα με την αποτύπωση της δομής του univariate Encoder-Decoder LSTM μοντέλου μας.



Εικόνα 43: Σχηματική αναπαράσταση του univariate Encoder-Decoder LSTM μοντέλου του προγράμματος (μέσω Keras)

7.1.3.3 Encoder – Decoder multivariate LSTM

Σε αυτή την ενότητα θα ενημερώσουμε τον encoder-decoder LSTM που αναπτύχθηκε στην προηγούμενη ενότητα ώστε να χρησιμοποιήσουμε καθεμία από τις οκτώ μεταβλητές χρονολογικών σειρών και να προβλέψουμε για την επόμενη κανονική εβδομάδα, τη συνολική ημερήσια κατανάλωση ενέργειας.

Αυτό θα το επιτύχουμε παρέχοντας κάθε μονοδιάστατη χρονολογική σειρά στο μοντέλο ως ξεχωριστή ακολουθία εισόδου.

Το LSTM με τη σειρά του δημιουργεί μια εσωτερική αναπαράσταση κάθε ακολουθίας εισόδου που ερμηνεύεται μαζί από τον αποκωδικοποιητή.

Η χρήση πολλών μεταβλητών εισόδων είναι χρήσιμη για εκείνα τα προβλήματα όπου η ακολουθία εξόδου είναι κάποια συνάρτηση των παρατηρήσεων σε προηγούμενα βήματα από πολλαπλές διαφορετικές λειτουργίες, όχι μόνο (ή συμπεριλαμβανομένης) της προβλεπόμενης δυνατότητας. Δεν είναι σαφές εάν συμβαίνει αυτό στο πρόβλημα κατανάλωσης ρεύματος, αλλά μπορούμε να το διερευνήσουμε.

Πρώτον, πρέπει να ενημερώσουμε την προετοιμασία του training dataset για να συμπεριλάβουμε όλα τα οκτώ χαρακτηριστικά, όχι μόνο τη συνολική ημερήσια ισχύ που καταναλώνουμε.

Πρέπει επίσης να ενημερώσουμε τη συνάρτηση που χρησιμοποιείται για την πρόβλεψη με το μοντέλο προσαρμογής για να χρησιμοποιήσουμε και τις οκτώ λειτουργίες από τα βήματα του προηγούμενου χρόνου.

Η ίδια αρχιτεκτονική μοντέλου και διαμόρφωση χρησιμοποιείται άμεσα, αν και θα αυξήσουμε τον αριθμό των εποχών εκπαίδευσης από 20 σε 50 δεδομένης της 8πλάσιας αύξησης της ποσότητας των δεδομένων εισόδου.

Η εκτέλεση του προγράμματος προσαρμόζει το μοντέλο και συνοψίζει την απόδοση στο test dataset.

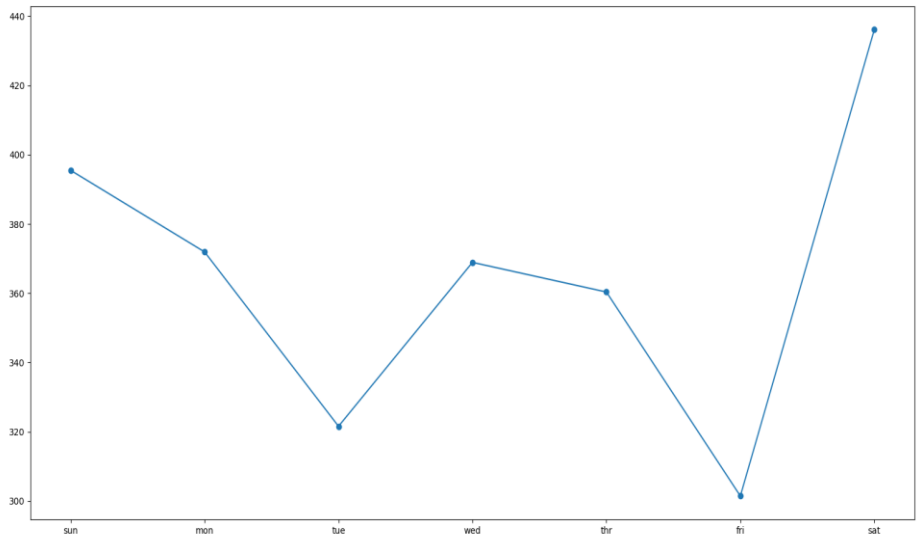
Διαπιστώνουμε ότι αυτό το μοντέλο εμφανίζεται λιγότερο σταθερό από το Encoder-Decoder LSTM μια μόνο μεταβλητής και μπορεί να σχετίζεται με τις διαφορετικές κλίμακες των οκτώ μεταβλητών εισόδου.

Τα συγκεκριμένα αποτελέσματα μπορεί να διαφέρουν ανάλογα με την στοχαστική φύση του αλγορίθμου. Διαπιστώνουμε ότι σε αυτή την περίπτωση, το μοντέλο είναι επιδέξιο, επιτυγχάνοντας συνολική βαθμολογία RMSE περίπου 367 kilowatt.

Πίνακας 7. Απόδοση Encoder-Decoder multivariate LSTM μοντέλου

Mean RMSE	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
367.451	395.5	371.9	321.6	368.9	360.4	301.5	436.1

Επίσης δημιουργείται μια γραφική παράσταση του RMSE ανά ημέρα.



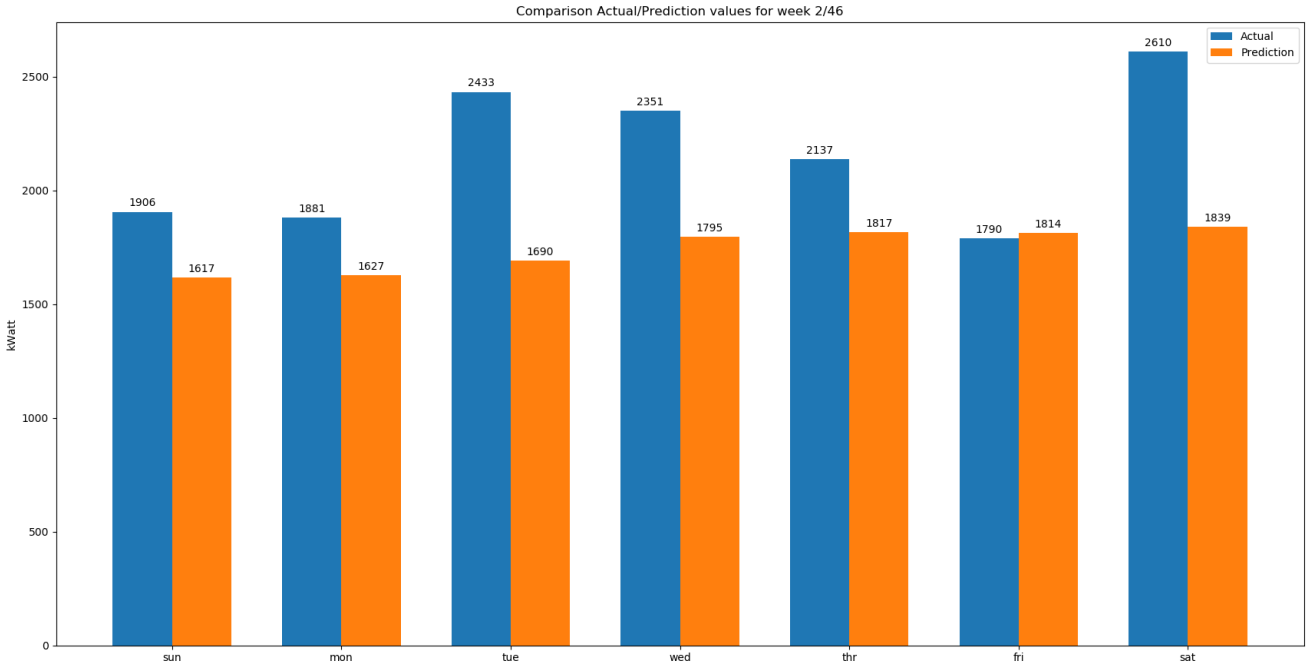
Εικόνα 44: Γραφική παράσταση των επιδόσεων του βέλτιστου multivariate EncoderDecoder-LSTM μοντέλου για το dataset

Πίνακας 8. Απόδοση πολλών μεταβλητών EncoderDecoder-LSTM μοντέλων

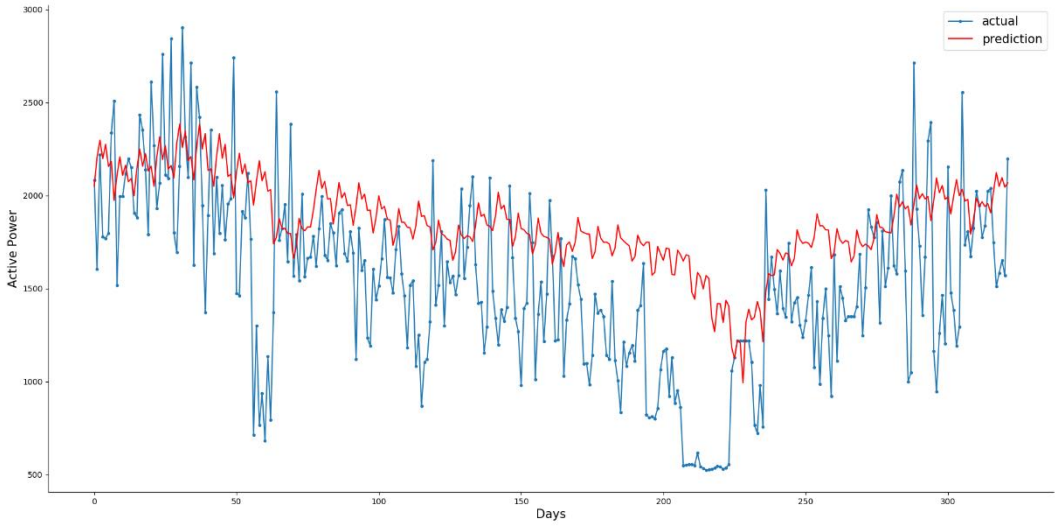
LSTM MODEL RESULTS							
inputs	14	14	14	14	14	14	14
Neurons1	200	200	200	200	200	200	200
Neurons2	200	100	200	200	200	200	200
Neurons3	-	-	200	100	-	-	-
Epochs	50	50	50	50	50	50	70
Batch size	16	16	16	16	30	50	30
Verbose	0	0	0	0	0	0	0
Loss func	Mse	Mse	Mse	Mse	Mse	Mse	Mse
Optimizer	adam	adam	adam	adam	adam	adam	adam
Dense	100	100	100	100	100	100	100
Dense	1	1	1	1	1	1	1
RMSE	367.451	1591.250	528.877	1117.752	429.944	603.583	405.83

Βλέπουμε την καλύτερη απόδοση έχει το πρώτο μοντέλο του πίνακα με συνολικό RMSE 367.451.

Ακολουθεί μια σύγκριση των προβλεπόμενων τιμών με τις πραγματικές για μια ενδεικτική εβδομάδα. Οι διαφορές τους ποικίλουν από τρέξιμο σε τρέξιμο και από εβδομάδα σε εβδομάδα. Μια πιο συνολική εικόνα έχουμε στην Εικόνα 46 όπου παρουσιάζεται μια σύγκριση των προβλεπόμενων τιμών με τις πραγματικές για ολόκληρο το testing dataset.

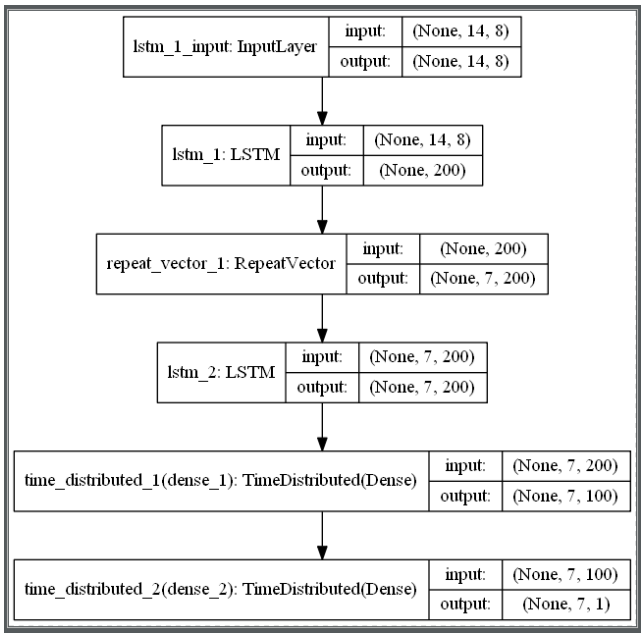


Εικόνα 45: Σύγκριση των πραγματικών και των προβλεπόμενων τιμών του βέλτιστου multivariate Encoder-Decoder LSTM μοντέλου για το dataset, για τη δεύτερη από τις 46 εβδομάδες του testing dataset



Εικόνα 46: Σύγκριση των πραγματικών και των προβλεπόμενων τιμών του βέλτιστου multivariate Encoder-Decoder LSTM μοντέλου για ολόκληρο το testing dataset

Τέλος, δημιουργούμε ένα σχήμα με την αποτύπωση της δομής του multivariate Encoder-Decoder LSTM μοντέλου μας.



Εικόνα 47: Σχηματική αναπαράσταση του multivariate Encoder-Decoder LSTM μοντέλου του προγράμματος (μέσω Keras)

7.1.3.4 Encoder-Decoder CNN- univariate LSTM

Ένα CNN νευρωνικό δίκτυο, μπορεί να χρησιμοποιηθεί ως κωδικοποιητής σε μια αρχιτεκτονική encoder-decoder.

Το CNN δεν υποστηρίζει απευθείας την είσοδο ακολουθίας. Αντίθετα, ένα 1D CNN είναι ικανό να διαβάσει την είσοδο αλληλουχίας και να μάθει αυτόματα τα κυριότερα χαρακτηριστικά. Αυτά μπορούν στη συνέχεια να ερμηνευθούν από έναν αποκωδικοποιητή LSTM σύμφωνα με την κανονική λειτουργικότητά του. Αναφερόμαστε σε υβριδικά μοντέλα που χρησιμοποιούν μοντέλα CNN και LSTM ως μοντέλα CNN-LSTM και σε αυτήν την περίπτωση τα χρησιμοποιούμε μαζί σε μια αρχιτεκτονική encoder-decoder.

Το CNN αναμένει ότι τα δεδομένα εισόδου θα έχουν την ίδια δομή 3D με το μοντέλο LSTM, παρόλο που πολλαπλές λειτουργίες διαβάζονται ως διαφορετικά κανάλια που τελικά έχουν το ίδιο αποτέλεσμα.

Όπως και πριν, θα χρησιμοποιήσουμε ακολουθίες εισόδου αποτελούμενες από 14 ημέρες ημερήσιας συνολικής κατανάλωσης ενέργειας.

Θα ορίσουμε μια απλή αλλά αποτελεσματική αρχιτεκτονική του CNN για τον κωδικοποιητή, ο οποίος αποτελείται από δύο συνελικτικά στρώματα ακολουθούμενα από ένα max pooling στρώμα συγκέντρωσης, τα αποτελέσματα των οποίων στη συνέχεια ομαλοποιούνται.

Το πρώτο συνελικτικό στρώμα διαβάζει την ακολουθία εισόδου και προβάλλει τα αποτελέσματα σε maps χαρακτηριστικών. Ο δεύτερος εκτελεί την ίδια λειτουργία στους maps χαρακτηριστικών που δημιουργούνται από το πρώτο στρώμα, προσπαθώντας να ενισχύσει οποιεσδήποτε σημαντικές λειτουργίες. Θα χρησιμοποιήσουμε 64 maps χαρακτηριστικών ανά συνελικτικό στρώμα και θα διαβάσουμε τις ακολουθίες εισόδου με kernel size τριών χρονικών βημάτων.

Το max pooling στρώμα συγκέντρωσης απλοποιεί τα maps χαρακτηριστικών διατηρώντας το 1/4 των τιμών με το μεγαλύτερο (μέγιστο) σήμα. Οι σταλμένοι maps χαρακτηριστικών μετά το pooling στρώμα ισιώνουν σε ένα μακρύ διάνυσμα που στη συνέχεια μπορεί να χρησιμοποιηθεί ως είσοδος στη διαδικασία αποκωδικοποίησης.

Ο αποκωδικοποιητής είναι ο ίδιος με αυτόν που ορίστηκε σε προηγούμενες ενότητες.

Η μόνη άλλη αλλαγή είναι ότι ορίστηκε ο αριθμός των εποχών εκπαίδευσης σε 20.

Τώρα είμαστε έτοιμοι να δοκιμάσουμε την αρχιτεκτονική encoder-decoder με κωδικοποιητή CNN.

Η εκτέλεση του προγράμματος προσαρμόζει το μοντέλο και συνοψίζει την απόδοση στο testing dataset.

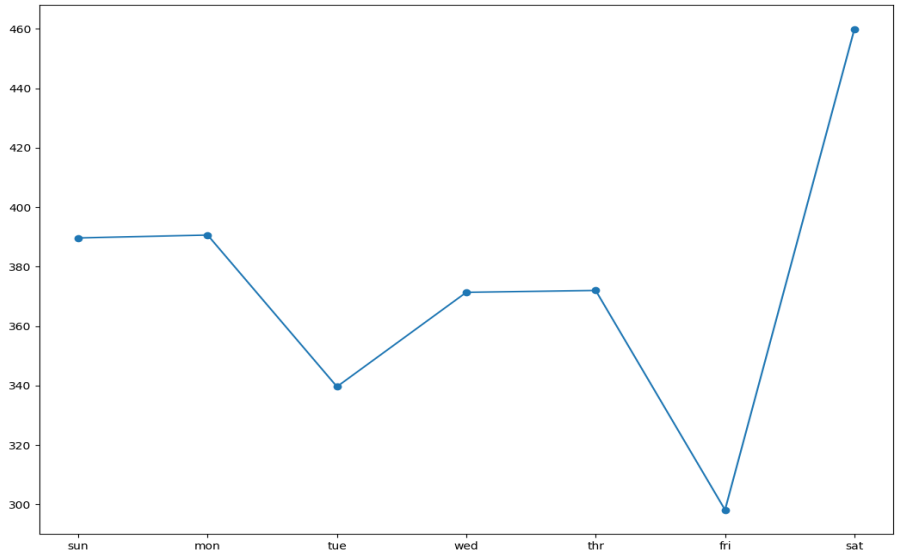
Ένας μικρός πειραματισμός έδειξε ότι η χρήση δύο συνθετικών στρώσεων έκανε το μοντέλο πιο σταθερό από το να χρησιμοποιεί μόνο ένα στρώμα.

Τα συγκεκριμένα αποτελέσματα μπορεί να διαφέρουν ανάλογα με την στοχαστική φύση του αλγορίθμου. Μπορούμε να δούμε ότι σε αυτή την περίπτωση το μοντέλο είναι επιδέξιο, επιτυγχάνοντας συνολική βαθμολογία RMSE περίπου 377 kilowatt.

Πίνακας 9. Απόδοση Encoder-Decoder CNN univariate LSTM μοντέλου

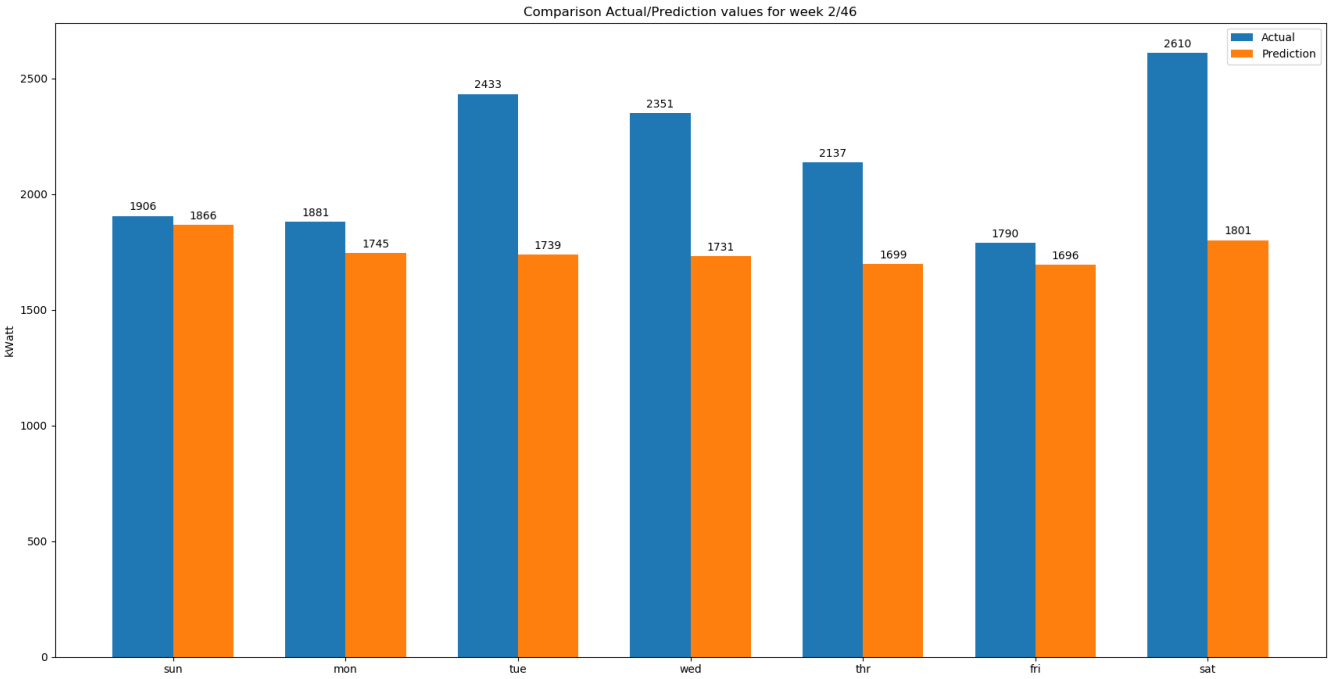
Mean RMSE	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
377.303	389.7	390.6	339.6	371.4	372.0	298.2	459.9

Επίσης δημιουργείται μια γραφική παράσταση του RMSE ανά ημέρα.

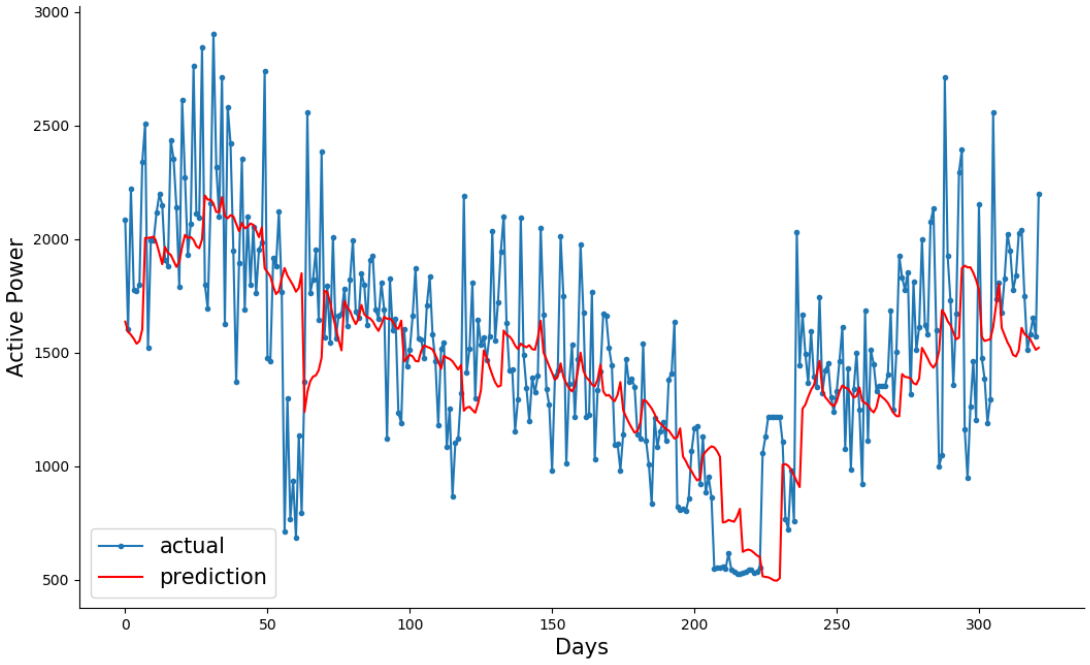


Εικόνα 48: Γραφική παράσταση των επιδόσεων του βέλτιστου univariate EncoderDecoder-CNN-LSTM μοντέλου για το dataset

Ακολουθεί μια σύγκριση των προβλεπόμενων τιμών με τις πραγματικές για μια ενδεικτική εβδομάδα. Οι διαφορές τους ποικίλουν από τρέξιμο σε τρέξιμο και από εβδομάδα σε εβδομάδα. Μια πιο συνολική εικόνα έχουμε στην Εικόνα 50 όπου παρουσιάζεται μια σύγκριση των προβλεπόμενων τιμών με τις πραγματικές για ολόκληρο το testing dataset.

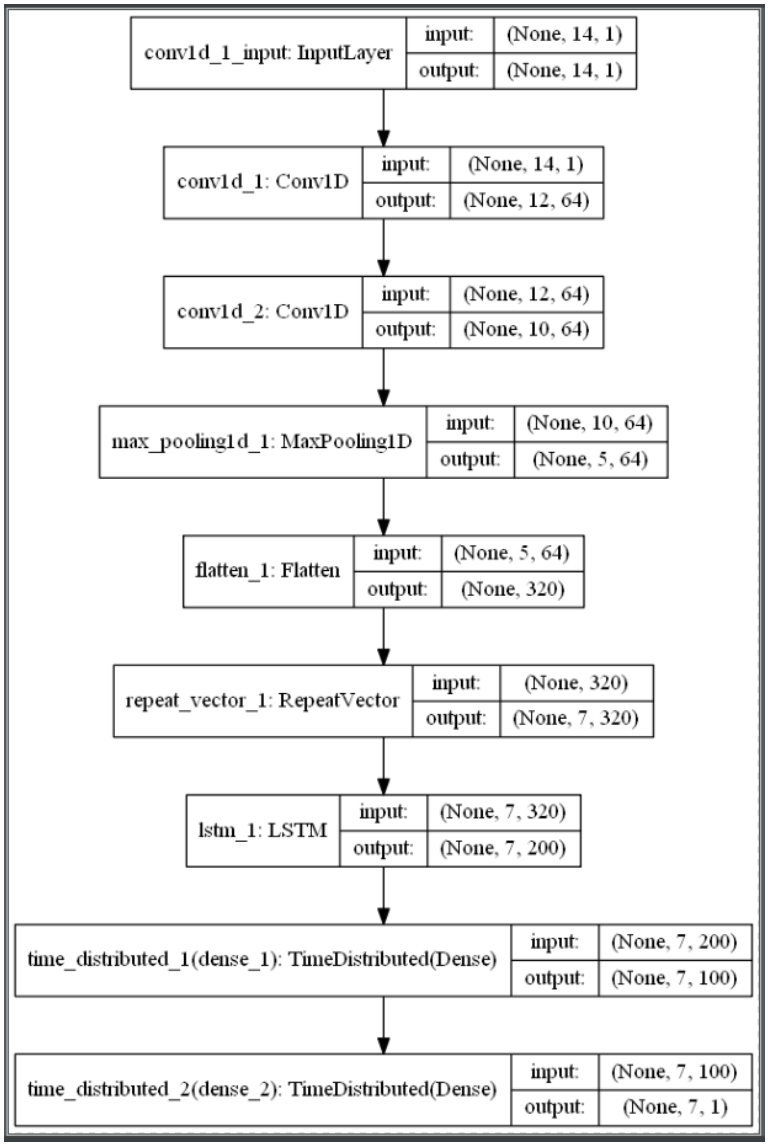


Εικόνα 49: Σύγκριση των πραγματικών και των προβλεπόμενων τιμών του βέλτιστου univariate Encoder-Decoder CNN LSTM μοντέλου για το dataset, για τη δεύτερη από τις 46 εβδομάδες του testing dataset



Εικόνα 50: Σύγκριση των πραγματικών και των προβλεπόμενων τιμών του βέλτιστου univariate Encoder-Decoder CNN LSTM μοντέλου για ολόκληρο το testing dataset

Τέλος, δημιουργούμε ένα σχήμα με την αποτύπωση της δομής του univariate Encoder-Decoder CNN LSTM μοντέλου μας.



Εικόνα 51: Σχηματική αναπαράσταση του univariate Encoder-Decoder CNN LSTM μοντέλου του προγράμματος (μέσω Keras)

7.1.3.5 Encoder-Decoder ConvLSTM – univariate

Μια περαιτέρω επέκταση της προσέγγισης CNN-LSTM είναι η εκτέλεση των συνελίξεων του CNN (π.χ., πώς το CNN διαβάζει τα δεδομένα ακολουθίας εισόδου) ως μέρος του LSTM για κάθε βήμα χρόνου. Αυτός ο συνδυασμός ονομάζεται Convolutional LSTM ή ConvLSTM.

Σε αντίθεση με ένα LSTM που διαβάζει τα δεδομένα απευθείας για να υπολογίσει τις εσωτερικές μεταβολές και της κατάστασης και σε αντίθεση με το CNN-LSTM που ερμηνεύει την έξοδο από μοντέλα CNN, το ConvLSTM

χρησιμοποιεί συνελίξεις απευθείας ως μέρος της ανάγνωσης εισόδου στις μονάδες LSTM.

Η βιβλιοθήκη Keras παρέχει την κλάση ConvLSTM2D που υποστηρίζει το μοντέλο ConvLSTM για δεδομένα 2D. Μπορεί να ρυθμιστεί και για πρόβλεψη χρονοσειρών 1D πολλαπλών μεταβλητών.

Η κλάση ConvLSTM2D, από προεπιλογή, αναμένει ότι τα δεδομένα εισόδου θα έχουν το σχήμα:

[samples, timesteps, rows, cols, channels]

Όπου κάθε βήμα δεδομένων ορίζεται ως εικόνα των σημείων δεδομένων (σειρές * στήλες).

Συνεργαζόμαστε με μια μονοδιάστατη ακολουθία της συνολικής κατανάλωσης ενέργειας, την οποία μπορούμε να ερμηνεύσουμε ως μία σειρά με 14 στήλες, αν υποθέσουμε ότι χρησιμοποιούμε δεδομένα εισόδου δύο εβδομάδων.

Για το ConvLSTM, αυτό θα ήταν μια μόνο ανάγνωση: δηλαδή, το LSTM θα διαβάσει ένα βήμα των 14 ημερών και θα εκτελέσει μια συνέλιξη σε αυτά τα χρονικά βήματα. Αυτό δεν είναι ιδανικό. Αντ' αυτού, μπορούμε να χωρίσουμε τις 14 ημέρες σε δύο υποενότητες με μήκος επτά ημερών. Το ConvLSTM μπορεί στη συνέχεια να διαβάσει τα δύο χρονικά βήματα και να εκτελέσει τη διαδικασία του CNN στις επτά ημέρες των δεδομένων μέσα σε κάθε μία.

Για αυτή την επιλεγμένη διαμόρφωση του προβλήματος, η είσοδος για το ConvLSTM2D είναι συνεπώς: [n, 2, 1, 7, 1]

Δηλαδή:

- Samples: n, για τον αριθμό των παραδειγμάτων στο σύνολο δεδομένων κατάρτισης.
- Time: 2, για τις δύο υποενότητες που διαιρέσαμε ένα παράθυρο 14 ημερών.
- Rows: 1, για το μονοδιάστατο σχήμα κάθε υπο-ακολουθίας.
- Columns: 7, για τις επτά ημέρες σε κάθε υποενότητα.
- Channels: 1, για την ενιαία λειτουργία με την οποία δουλεύουμε ως είσοδος.

Μπορούμε τώρα να προετοιμάσουμε τα δεδομένα για το μοντέλο ConvLSTM2D.

Πρώτον, πρέπει να αναμορφώσουμε το training dataset στην αναμενόμενη δομή των [samples, timesteps, rows, cols, channels].

Στη συνέχεια, μπορούμε να καθορίσουμε τον κωδικοποιητή ως κρυφό στρώμα ConvLSTM που ακολουθείται από ένα επίπεδο επιπέδων που είναι έτοιμο για αποκωδικοποίηση.

Επίσης, θα παραμετροποιήσουμε τον αριθμό των ακολουθιών (n_steps) και το μήκος κάθε υποενότητας (n_length) και θα τα περάσουμε ως arguments.

Το υπόλοιπο μοντέλο και η εκπαίδευση είναι τα ίδια.

Αυτό το μοντέλο αναμένει ότι εισάγονται πενταδιάστατα δεδομένα. Επομένως, πρέπει επίσης να ενημερώσουμε την προετοιμασία ενός μόνο δείγματος στη συνάρτηση forecast() όταν πραγματοποιούμε μια πρόβλεψη.

Έχουμε τώρα όλα τα στοιχεία για την αξιολόγηση μιας αρχιτεκτονικής encoder-decoder για την πρόβλεψη χρονοσειρών πολλαπλών σταδίων όπου χρησιμοποιείται το ConvLSTM ως κωδικοποιητής.

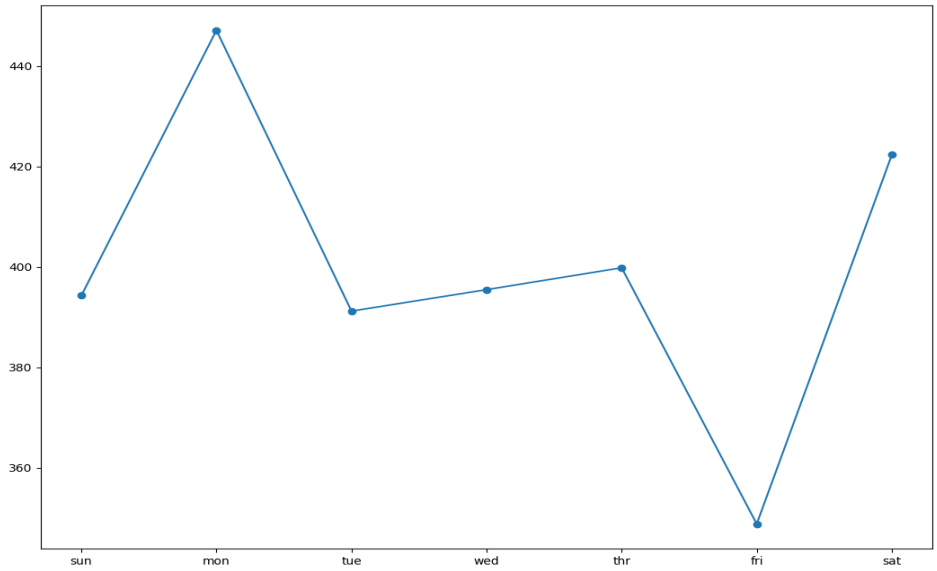
Ένας μικρός πειραματισμός έδειξε ότι η χρήση δύο συνθετικών στρώσεων έκανε το μοντέλο πιο σταθερό από το να χρησιμοποιεί μόνο ένα στρώμα.

Μπορούμε να δούμε ότι σε αυτή την περίπτωση το μοντέλο είναι επιδέξιο, επιτυγχάνοντας συνολική βαθμολογία RMSE περίπου 379 kilowatt.

Πίνακας 10. Απόδοση Encoder-Decoder univariate ConvLSTM μοντέλου

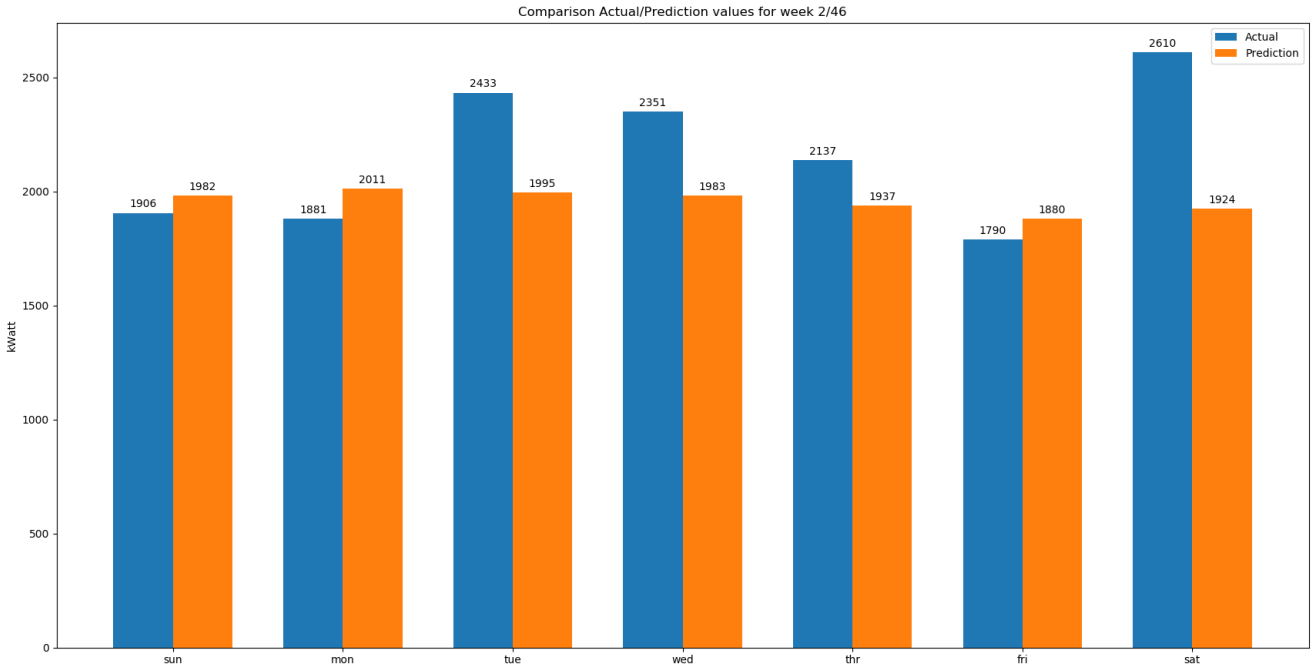
Mean RMSE	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
379.307	401.8	411.1	344.3	375.4	369.3	300.2	436.7

Επίσης δημιουργείται μια γραφική παράσταση του RMSE ανά ημέρα.

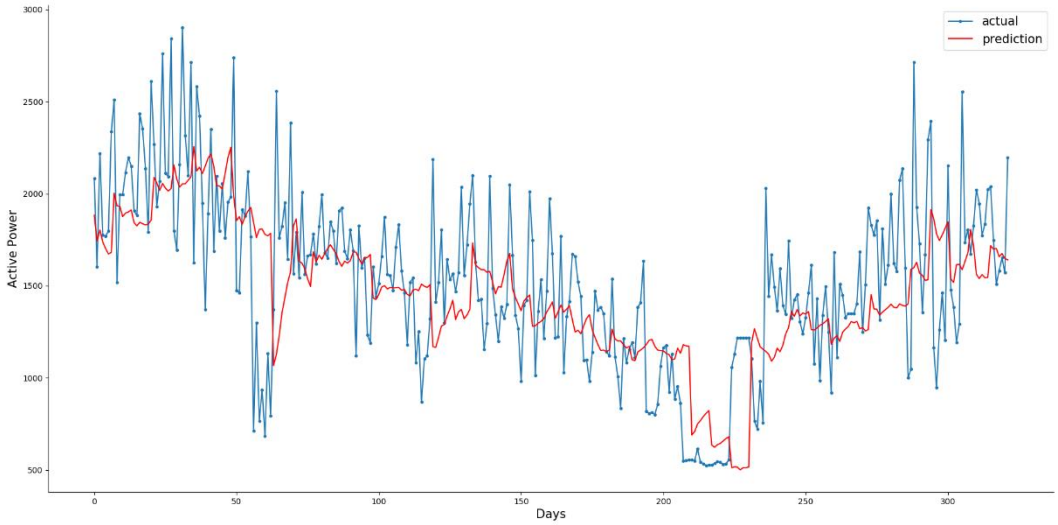


Εικόνα 52: Γραφική παράσταση των επιδόσεων του βέλτιστου univariate EncoderDecoder-Conv-LSTM μοντέλου για το dataset

Ακολουθεί μια σύγκριση των προβλεπόμενων τιμών με τις πραγματικές για μια ενδεικτική εβδομάδα. Οι διαφορές τους ποικίλουν από τρέξιμο σε τρέξιμο και από εβδομάδα σε εβδομάδα. Μια πιο συνολική εικόνα έχουμε στην Εικόνα 54 όπου παρουσιάζεται μια σύγκριση των προβλεπόμενων τιμών με τις πραγματικές για ολόκληρο το testing dataset.

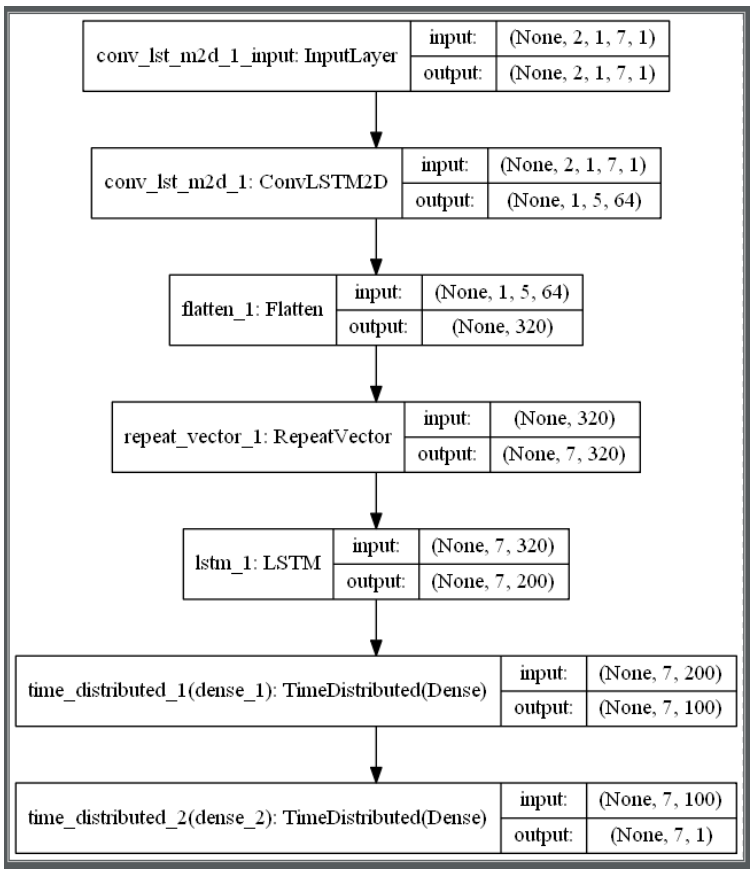


Εικόνα 53: Σύγκριση των πραγματικών και των προβλεπόμενων τιμών του βέλτιστου univariate Encoder-Decoder-Conv-LSTM μοντέλου για το dataset, για τη δεύτερη από τις 46 εβδομάδες του testing dataset



Εικόνα 54: Σύγκριση των πραγματικών και των προβλεπόμενων τιμών του βέλτιστου univariate Encoder-Decoder-Conv-LSTM μοντέλου για ολόκληρο το testing dataset

Τέλος, δημιουργούμε ένα σχήμα με την αποτύπωση της δομής του univariate Encoder-Decoder-Conv- LSTM μοντέλου μας.



Εικόνα 55: Σχηματική αναπαράσταση του univariate Encoder-Decoder-Conv- LSTM μοντέλου του προγράμματος (μέσω Keras)

7.2 Δεδομένα Συγκέντρωσης Αιωρούμενων Σωματιδίων PM2.5

Το Air Quality Prediction dataset παρέχεται από το UCI Machine Learning Repository. Αποτελείται από ωριαίες μετρήσεις καιρικών μεταβλητών και ρύπανσης για περίοδο πέντε χρόνων με κέντρο λήψης την Αμερικάνικη Πρεσβεία στο Πεκίνο, Κίνα.

Συγκεκριμένα, τα δεδομένα περιλαμβάνουν την ημερομηνία-ώρα, τη ρύπανση από μικροσωματίδια (συγκέντρωση PM2.5) και πληροφορίες για τις καιρικές συνθήκες, συμπεριλαμβανομένου του σημείου δρόσου, της θερμοκρασίας, της πίεσης, της κατεύθυνσης του ανέμου, της ταχύτητας του ανέμου και του συνολικού αριθμού ωρών χιονιού και βροχής. Ο πλήρης κατάλογος χαρακτηριστικών στα ακατέργαστα δεδομένα έχει ως εξής:

- **No**: αριθμός σειράς
- **year**: έτος λήψης των δεδομένων αυτής της σειράς
- **month**: μήνας λήψης των δεδομένων σε αυτή τη σειρά
- **day**: ημέρα λήψης των δεδομένων σε αυτή τη σειρά
- **hour**: ώρα λήψης των δεδομένων σε αυτή τη σειρά
- **PM2.5**: συγκέντρωση PM2.5
- **DEWP**: σημείο δρόσου
- **TEMP**: θερμοκρασία
- **PRES**: πίεση
- **cbwd**: συνδυαστική διεύθυνση ανέμου
- **Iws**: συνολική ταχύτητα ανέμου
- **Is**: συνολικές ώρες χιονιού
- **Ir**: συνολικές ώρες βροχής

Χρησιμοποιώντας τα παραπάνω δεδομένα, πλαισιώνουμε το εξής πρόβλημα πρόγνωσης: με βάση τις καιρικές συνθήκες και τη ρύπανση των προηγούμενων ωρών, να προβλεφθεί η ρύπανση την επόμενη ώρα.

7.2.1 Προετοιμασία των δεδομένων

Τα δεδομένα δεν είναι έτοιμα για χρήση. Πρέπει να την προετοιμάσουμε πρώτα. Παρακάτω παρατίθενται οι έξη πρώτες γραμμές του ακατέργαστου dataset.

```
No,year,month,day,hour,pm2.5,DEWP,TEMP,PRES,cbwd,Iws,Is,Ir
1,2010,1,1,0,NA,-21,-11,1021,NW,1.79,0,0
2,2010,1,1,1,NA,-21,-12,1020,NW,4.92,0,0
3,2010,1,1,2,NA,-21,-11,1019,NW,6.71,0,0
4,2010,1,1,3,NA,-21,-14,1019,NW,9.84,0,0
5,2010,1,1,4,NA,-20,-12,1018,NW,12.97,0,0
```

Εικόνα 56: Οι πρώτες γραμμές του ακατέργαστου dataset

Σε πρώτο στάδιο, ενώνουμε τις πληροφορίες ημερομηνίας-ώρας σε μια datetime μεταβλητή, έτσι ώστε να μπορούμε να το χρησιμοποιήσουμε ως δείκτη στο Pandas dataframe.

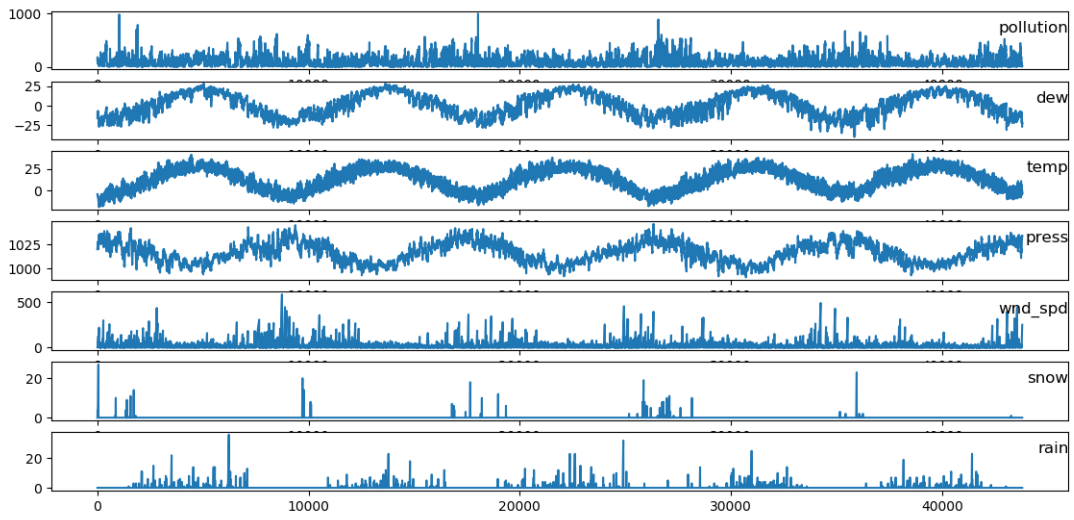
Ένας γρήγορος έλεγχος αποκαλύπτει NA τιμές για τη συγκέντρωση PM2.5 τις πρώτες 24 ώρες. Επομένως, θα πρέπει να αφαιρέσουμε την πρώτη σειρά δεδομένων. Υπάρχουν επίσης διάσπαρτες τιμές NA στο σύνολο των δεδομένων. Τις αντικαταστούμε με τιμή 0 για αρχή. Τέλος, η πρώτη στήλη "No" δεν κρίνεται με αυτές τις αλλαγές χρήσιμη στο πρόβλημά μας, επομένως θα αφαιρεθεί.

Οι αλλαγές αυτές αποθηκεύονται σε ένα νέο αρχείο με την ονομασία "pollution.csv". Ακολουθούν οι έξη πρώτες γραμμές του επεξεργασμένου συνόλου δεδομένων.

```
date,pollution,dew,temp,press,wnd_dir,wnd_spd,snow,rain
2010-01-02 00:00:00,129.0,-16,-4.0,1020.0,SE,1.79,0,0
2010-01-02 01:00:00,148.0,-15,-4.0,1020.0,SE,2.68,0,0
2010-01-02 02:00:00,159.0,-11,-5.0,1021.0,SE,3.57,0,0
2010-01-02 03:00:00,181.0,-7,-5.0,1022.0,SE,5.36,1,0
2010-01-02 04:00:00,138.0,-7,-5.0,1022.0,SE,6.25,2,0
```

Εικόνα 57: Οι πρώτες γραμμές του τροποποιημένου dataset

Έχοντας τα δεδομένα σε αυτή τη μορφή, δημιουργούμε τα γραφήματα για κάθε μια χρονοσειρά του επεξεργασμένου συνόλου, εκτός από τη διεύθυνση της ταχύτητας του ανέμου.



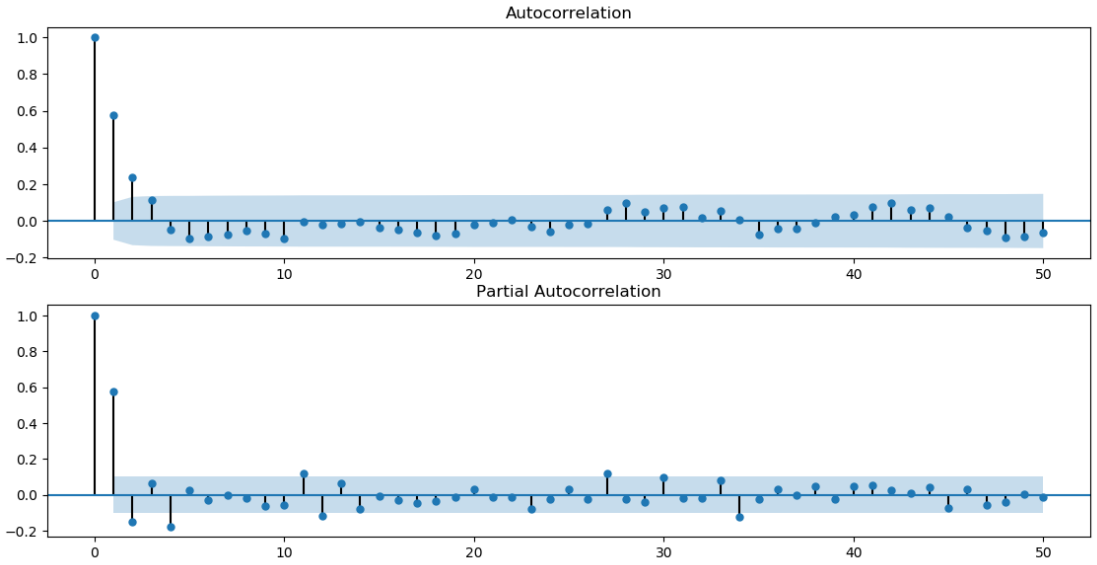
Εικόνα 58: Η γραφική αναπαράσταση των μεταβλητών του dataset από τη διεύθυνση της ταχύτητας του ανέμου για όλη την περιεχόμενη περίοδο

7.2.2 ARIMA

7.2.2.1 Autocorrelation Analysis

Όπως και στο προηγούμενο dataset που αναλύσαμε, έτσι και εδώ πριν να ορίσουμε το ARIMA μοντέλο, θα υπολογίσουμε τη συσχέτιση των παρατηρήσεων μέσω των γραφημάτων ACF και PACF, χρησιμοποιώντας τις συναρτήσεις plot_acf() και plot_pacf().

Η διαδικασία που ακολουθείται είναι παρόμοια. Μεγεθύνουμε κάθε γράφημα, θέτοντας το lag των παρατηρήσεων 50 ώστε να είναι ευδιάκριτο.



Εικόνα 59: Οι γραφικές παραστάσεις των ACF και PACF με lag 50

Μπορούμε να δούμε ένα οικείο μοτίβο αυτοσυσχέτισης και στα δύο γραφήματα. Αυτό το μοτίβο έχει τα εξής στοιχεία:

ACF και PACF: Μερικές σημαντικές παρατηρήσεις υστέρησης που πέφτουν απότομα καθώς αυξάνεται η υστέρηση.

Οι γραφικές παραστάσεις ACF και PACF υποδεικνύουν ότι υπάρχει αυτοσυσχέτιση μεταξύ των παρατηρήσεων και είναι διακριτό για τις τρεις πρώτες περίπου παρατηρήσεις υστερήσεων.

Αυτό υποδηλώνει ότι ένα καλό μοντέλο έναρξης θα ήταν ένα AR (3). Αυτό είναι ένα μοντέλο autoregression με 24 παρατηρήσεις υστέρησης που χρησιμοποιούνται ως είσοδος.

7.2.2.2 Ανάπτυξη Autoregression Μοντέλου

Μπορούμε να αναπτύξουμε ένα autoregression μοντέλο για μια ενιαία χρονοσειρά ωριαίας συγκέντρωσης PM2.5.

Η βιβλιοθήκη Statsmodels παρέχει πολλούς τρόπους για την ανάπτυξη ενός μοντέλου AR, όπως η χρήση των κλάσεων AR, ARMA, ARIMA και SARIMAX.

Θα χρησιμοποιήσουμε την κλάση ARIMA και εδώ, καθώς επιτρέπει την εύκολη επεκτασιμότητα στη μέθοδο των διαφορών (differencing) και κινητών μέσων (moving average).

Αναπτύχθηκαν και άλλα ARIMA μοντέλα με τα τελικά αποτελέσματα να παρουσιάζονται στον παρακάτω πίνακα:

Πίνακας 11. Απόδοση ARIMA μοντέλων

Model	RMSE
ARIMA (3,0,0)	72.237
ARIMA (4,0,0)	72.253
ARIMA (5,0,0)	72.252
ARIMA (6,0,0)	72.242
ARIMA (7,0,0)	72.207

Η εκτέλεση του προγράμματος εκτυπώνει την απόδοση του AR (7) στο σύνολο δεδομένων δοκιμής.

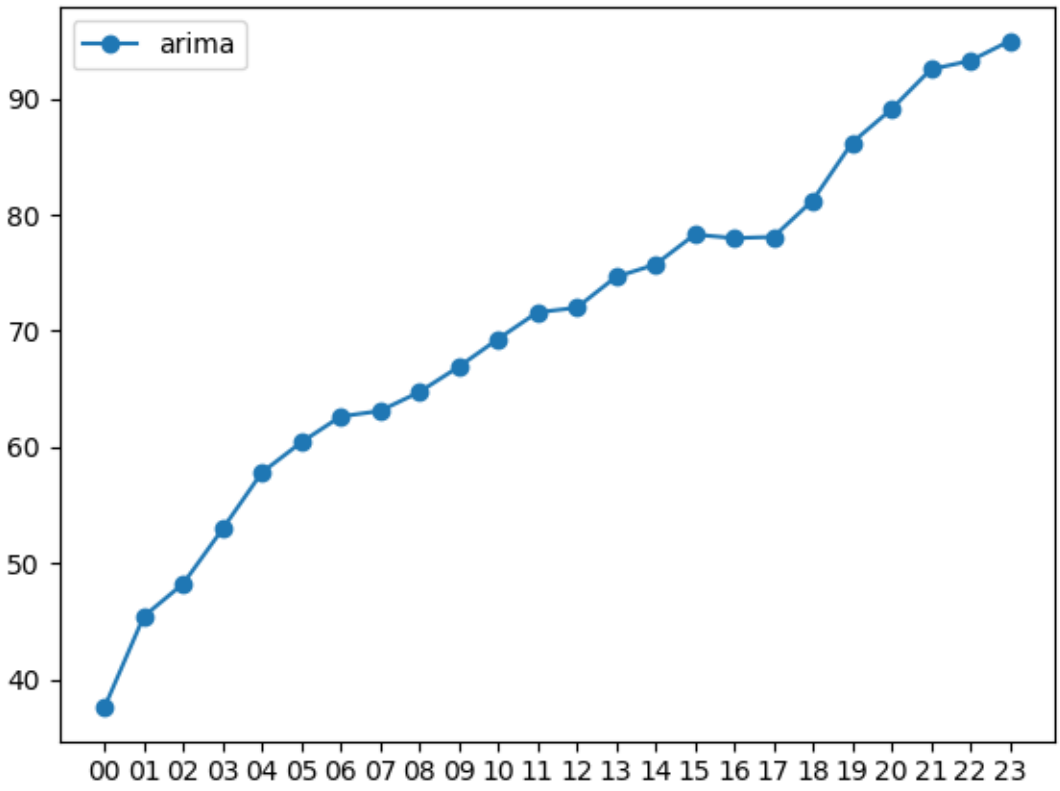
Βλέπουμε ότι το μοντέλο επιτυγχάνει συνολικό RMSE 72.207.

Πίνακας 12. Απόδοση ARIMA(7,0,0) μοντέλου

Mean RMSE	0	1	2	3	4	5	6	7	8	9	10	11
72.207	37.6	45.4	48.2	52.9	57.8	60.4	62.6	63.1	64.7	66.9	69.3	71.6
12	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	
72.0	74.7	75.7	78.3	78.0	78.1	81.2	86.2	89.1	92.5	93.2	94.9	

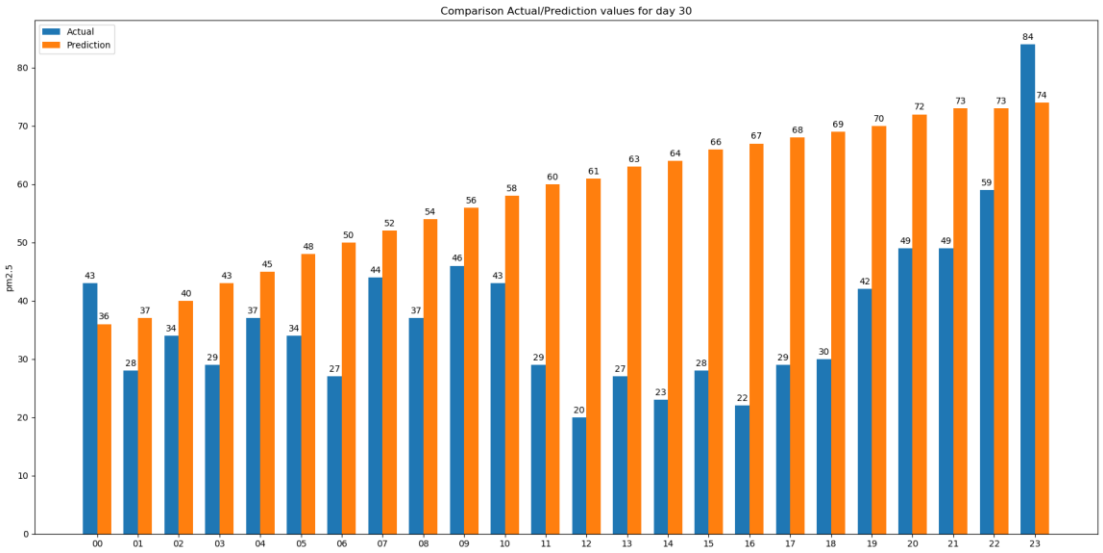
Επίσης δημιουργείται μια γραφική παράσταση της πρόβλεψης, η οποία δείχνει το RMSE για καθέναν από τους 24 χρόνους παράδοσης της πρόβλεψης.

Γενικά αναμένουμε ότι οι αρχικοί χρόνοι πρόβλεψης είναι ευκολότεροι να προβλεφθούν από τους μεταγενέστερους , καθώς το σφάλμα σε κάθε διαδοχική χρονική περίοδο συνθέτει. Αυτό επιβεβαιώνεται από το διάγραμμα που δημιουργήθηκε.

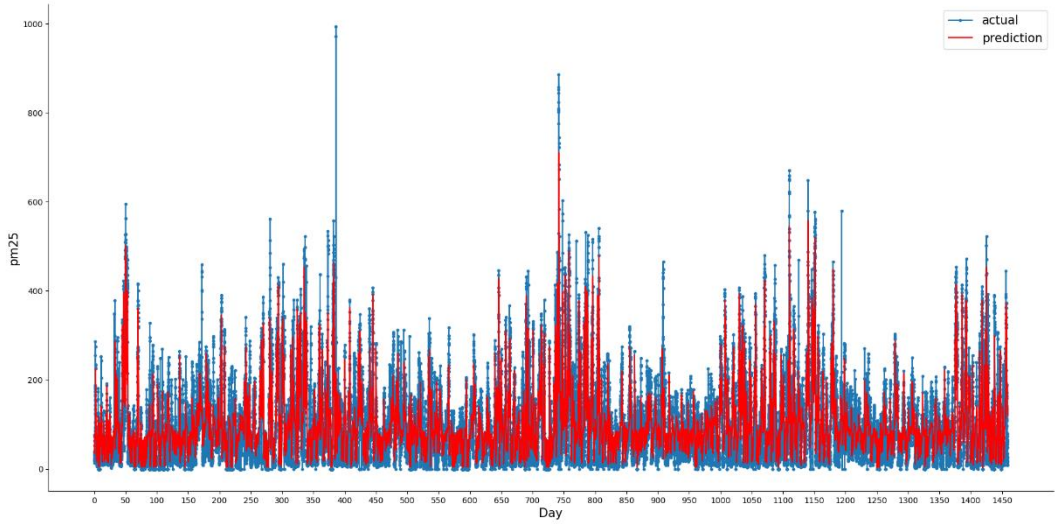


Εικόνα 60: Γραφική παράσταση των επιδόσεων του βέλτιστου ARIMA μοντέλου για το dataset

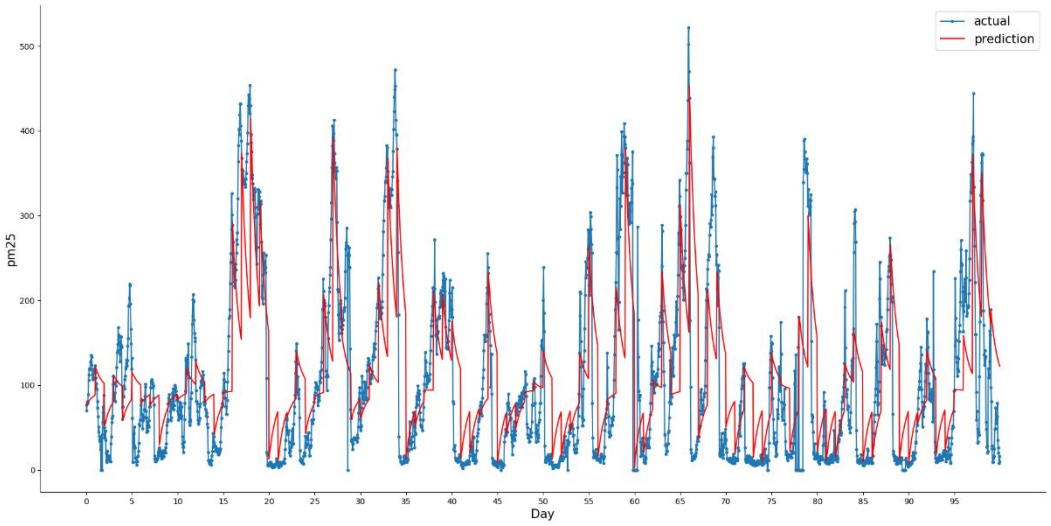
Ακολουθεί μια σύγκριση των προβλεπόμενων τιμών με τις πραγματικές για μια ενδεικτική μέρα. Οι διαφορές τους ποικίλουν από τρέξιμο σε τρέξιμο και από μέρα σε μέρα. Μια πιο συνολική εικόνα έχουμε στην Εικόνα 62 όπου παρουσιάζεται μια σύγκριση των προβλεπόμενων τιμών με τις πραγματικές για ολόκληρο το testing dataset, ενώ στην Εικόνα 63 εστιάζουμε στις 100 τελευταίες ημέρες του testing dataset για μια καλύτερη παρουσίαση των αποτελεσμάτων σύγκρισης.



Εικόνα 61: Σύγκριση των πραγματικών και των προβλεπόμενων τιμών του βέλτιστου ARIMA μοντέλου για το dataset, για τη 30η ημέρα από το testing dataset



Εικόνα 62: Σύγκριση των πραγματικών και των προβλεπόμενων τιμών του βέλτιστου ARIMA μοντέλου για το dataset, για το σύνολο του testing dataset.



Εικόνα 63: Σύγκριση των πραγματικών και των προβλεπόμενων τιμών του βέλτιστου ARIMA μοντέλου για το dataset, για τις τελευταίες 100 ημέρες του testing dataset.

7.2.3 LSTMs

7.2.3.1 LSTM multivariate

Σε αυτή την ενότητα, θα αναπτύξουμε ένα LSTM στο πρόβλημά μας. Το πρώτο βήμα είναι η προετοιμασία του συνόλου δεδομένων ρύπανσης για το LSTM. Αυτό περιλαμβάνει την πλαισίωση του συνόλου δεδομένων ως πρόβλημα επίβλεψης και την κανονικοποίηση των μεταβλητών εισόδου.

Πλαισιώνουμε το επίμαχο μαθησιακό πρόβλημα ως πρόβλεψη της ρύπανσης στην τρέχουσα ώρα (t) έχοντας ως δεδομένα τις μετρήσεις της ρύπανσης και των καιρικών συνθηκών στα προηγούμενα χρονικά βήματα.

Με βάση τα παραπάνω, αρχικά φορτώνεται το σύνολο δεδομένων του αρχείου "pollution.csv". Η χρονοσειρά της ταχύτητας ανέμου κωδικοποιείται με τη μέθοδο one-hot encoding. Στη συνέχεια, όλες οι μεταβλητές κανονικοποιούνται και έτσι το σύνολο των δεδομένων μετασχηματίζεται σε πρόβλημα επίβλεψης. Κατόπιν, αφαιρούνται οι μεταβλητές σχετικά με τις καιρικές συνθήκες για την προβλεπόμενη ώρα (t). Ακολουθεί μια πρώτη εκτύπωση των 8 μεταβλητών εισόδου και της μίας μεταβλητής εξόδου, αυτή της ρύπανσης την ζητούμενη ώρα.

	var1(t-1)	*var2(t-1)*	*var3(t-1)*	...	*var7(t-1)*	*var8(t-1)*	*var1(t)*
1	*0.129779*	*0.352941*	*0.245902*	...	*0.000000*	*0.0*	*0.148893*
2	*0.148893*	*0.367647*	*0.245902*	...	*0.000000*	*0.0*	*0.159960*
3	*0.159960*	*0.426471*	*0.229508*	...	*0.000000*	*0.0*	*0.182093*
4	*0.182093*	*0.485294*	*0.229508*	...	*0.037037*	*0.0*	*0.138833*

5 0.138833 0.485294 0.229508 ... 0.074074 0.0 0.109658

7.2.3.1.1 Ορισμός και προσαρμογή του μοντέλου

Σε αυτήν την ενότητα, θα τοποθετήσουμε ένα LSTM μοντέλο στα δεδομένα έχοντας ως είσοδο πολλές μεταβλητές.

Αρχικά, χωρίζουμε το προετοιμασμένο σύνολο δεδομένων σε train dataset και test dataset. Για να επιταχύνουμε την κατάρτιση του μοντέλου, εκπαιδεύουμε το μοντέλο μόνο με τα δεδομένα του πρώτου έτους και στη συνέχεια το αξιολογούμε με τα υπόλοιπα 4 χρόνια. Τέλος, οι είσοδοι μετασχηματίζονται σε μορφή 3D που αναμένει ένα LSTM μοντέλο [samples, timesteps, features]

Έτσι, το σχήμα εισόδου για το train dataset είναι: (8760, 1, 8) και εξόδου: (8760,) ενώ το σχήμα εισόδου για το test dataset είναι: (35039, 1, 8) και εξόδου: (35039,).

Τώρα μπορούμε να ορίσουμε και να προσαρμόσουμε το LSTM μοντέλο μας.

Ορίζουμε ένα LSTM μοντέλο με 50 νευρώνες στο πρώτο κρυφό στρώμα και 1 νευρώνα στο επίπεδο εξόδου για την πρόβλεψη ρύπανσης. Το σχήμα εισόδου έχει ένα χρονικό βήμα με 8 χαρακτηριστικά.

Χρησιμοποιούμε τη Mean Absolute Error (MAE) loss function και τον optimizer Adam. Το μοντέλο προσαρμόζεται για 50 training epochs με batch size 72. Υπενθυμίζεται ότι η εσωτερική κατάσταση του LSTM στον Keras επαναρυθμίζεται στο τέλος κάθε batch. Επιπλέον, παρακολουθούμε τόσο τις training όσο και τις test απώλειες κατά τη διάρκεια της εκπαίδευσης, θέτοντας το όρισμα validation_data= true στη συνάρτηση *fit()*.

7.2.3.1.2 Αξιολόγηση του μοντέλου

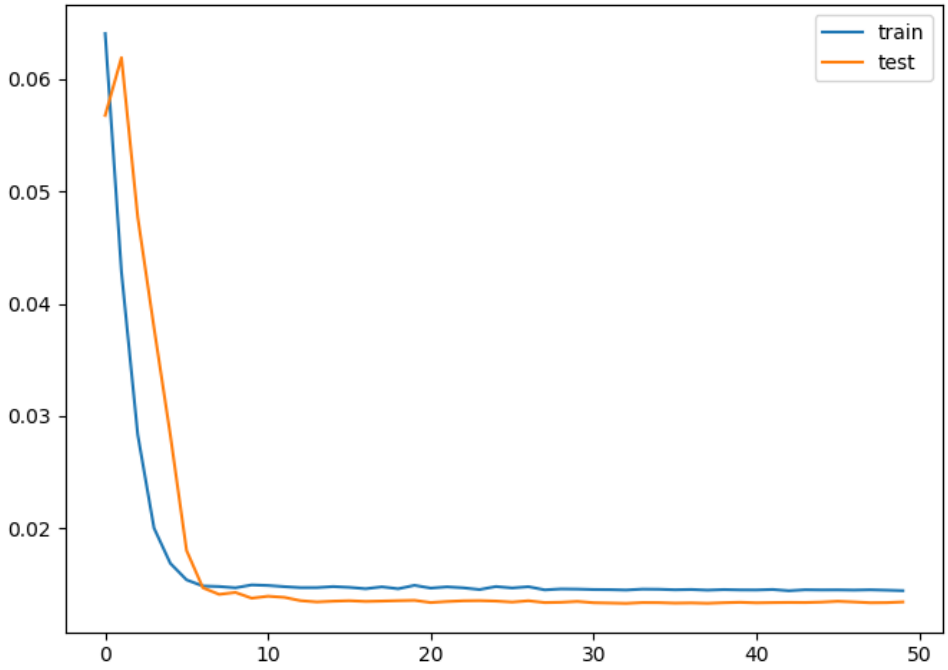
Αφού το μοντέλο είναι προσαρμοσμένο, μπορούμε να κάνουμε προβλέψεις για ολόκληρο το test dataset.

Συνδυάζουμε την πρόβλεψη με το test dataset και αντιστρέφουμε την κλίμακα. Αντιστρέψουμε επίσης την κλίμακα στο test dataset με τις αναμενόμενες τιμές ρύπανσης.

Με τις προβλέψεις και τις πραγματικές τιμές στην αρχική τους κλίμακα, μπορούμε να υπολογίσουμε το βαθμό σφάλματος για το μοντέλο. Σε αυτή την περίπτωση, υπολογίζουμε το RMSE που δίνει σφάλμα στις ίδιες μονάδες με την ίδια την μεταβλητή.

Η εκτέλεση του παραδείγματος δημιουργεί πρώτα ένα γράφημα που δείχνει τις train και test απώλειες κατά τη διάρκεια της εκπαίδευσης.

Είναι ενδιαφέρον ότι μπορούμε να δούμε ότι οι test απώλειες πέφτουν κάτω από τις train απώλειες. Η μέτρηση και η σχεδίαση του RMSE κατά τη διάρκεια της εκπαίδευσης μπορεί να αποδώσει περισσότερο φως σε αυτό.



Εικόνα 64: Γραφική παράσταση των επιδόσεων του univariate-LSTM μοντέλου για το train και το test dataset

Οι train και οι test απώλειες τυπώνονται στο τέλος κάθε εποχή εκπαίδευσης. Στο τέλος της εκτέλεσης του μοντέλου, εκτυπώνεται το τελικό RMSE του στο σύνολο του test dataset.

```
...
Epoch 46/50
- 1s - loss: 0.0145 - val_loss: 0.0135
Epoch 47/50
- 1s - loss: 0.0145 - val_loss: 0.0134
Epoch 48/50
- 1s - loss: 0.0145 - val_loss: 0.0133
Epoch 49/50
- 1s - loss: 0.0145 - val_loss: 0.0133
Epoch 50/50
- 1s - loss: 0.0144 - val_loss: 0.0134
Test RMSE: 26.543
```

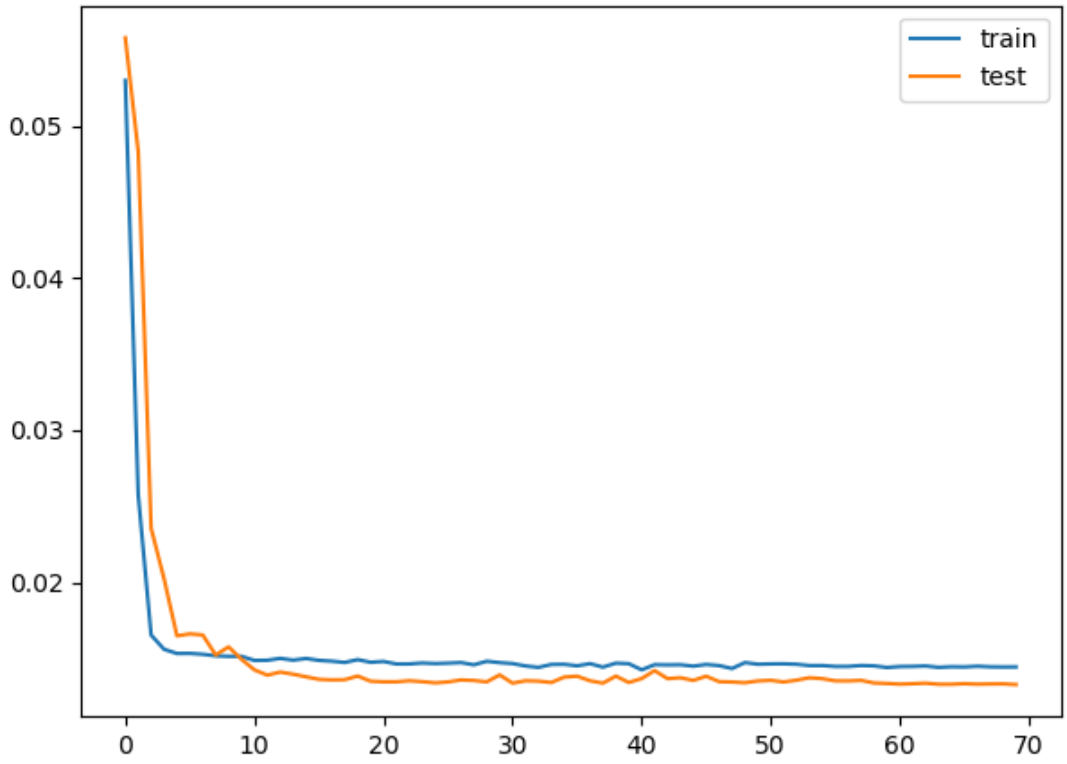
Παρατηρούμε ότι το μοντέλο επιτυγχάνει σκορ RMSE 26.543.

Επαναλαμβάνουμε τη διαδικασία κάνοντας μικροαλλαγές στη δομή του μοντέλου. Τα αποτελέσματα παρουσιάζονται στον ακόλουθο πίνακα.

Πίνακας 13. Απόδοση LSTM μοντέλων

LSTM MODEL RESULTS									
Neurons	50	50	50	50	100	200	200	200	200
Epochs	50	100	50	50	50	50	70	70	70
Batch size	72	72	24	72	72	72	72	24	96
Verbose	2	2	2	0	2	2	2	2	2
Loss func	Mae	Mae	Mae	Mae	Mae	Mae	Mae	Mae	Mae
Optimizer	adam	adam	adam	adam	adam	adam	adam	adam	adam
Dense	1	1	1	1	1	1	1	1	1
RMSE	26.543	26.799	28.402	26.601	26.435	26.381	26.229	28.912	28.592

Το αντίστοιχο διάγραμμα για το βέλτιστο μοντέλο LSTM που βρέθηκε (RMSE: 26.229) είναι το εξής:



Εικόνα 65: Γραφική παράσταση των επιδόσεων του βέλτιστου univariate-LSTM μοντέλου για το train και το test dataset

7.2.3.2 LSTM με multiple timesteps

Οι αλλαγές που απαιτούνται για την εκπαίδευση του μοντέλου σε πολλά προηγούμενα βήματα είναι ελάχιστες.

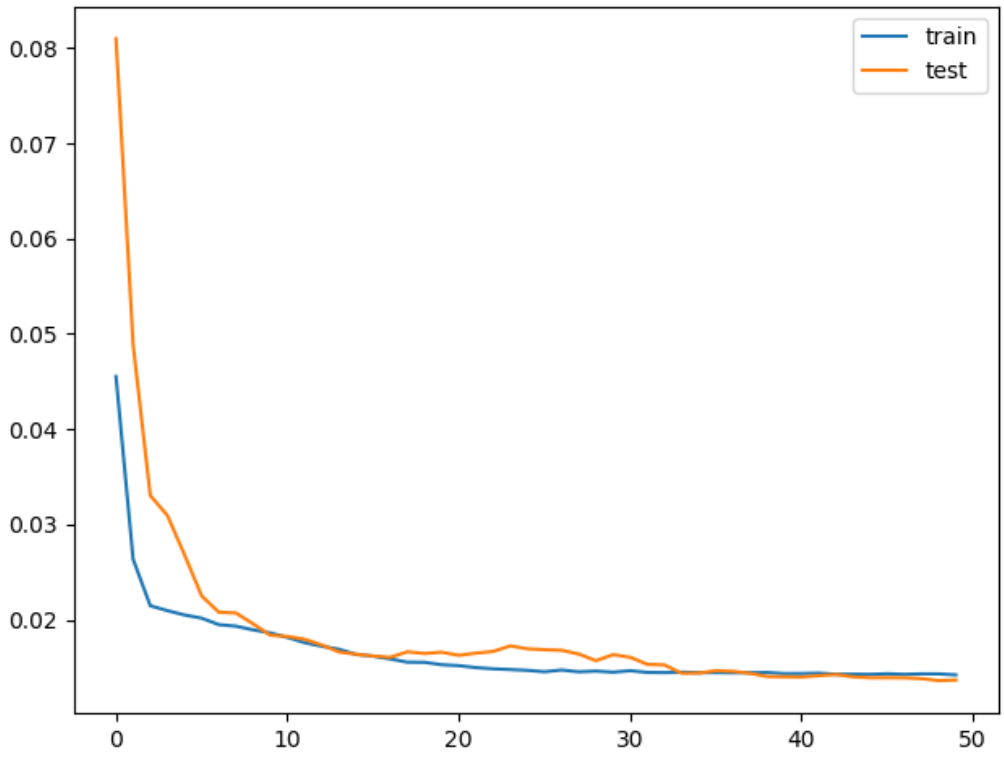
Πρώτα, πρέπει να πλαισιώσουμε το πρόβλημα κατάλληλα όταν καλούμε τη συνάρτηση series_to_supervised(). Χρησιμοποιούμε 3 ώρες δεδομένων ως είσοδο. Στη συνέχεια, πρέπει να είμαστε πιο προσεκτικοί στον προσδιορισμό της στήλης για είσοδο και έξοδο. Έχουμε 3 * 8 + 8 στήλες στο σύνολο των δεδομένων μας. Θα χρειαστούμε 3 * 8 ή 24 στήλες ως είσοδο για τη συμπλήρωση όλων των χαρακτηριστικών κατά τις προηγούμενες 3 ώρες. Παίρνουμε μόνο τη μεταβλητή ρύπανσης ως έξοδο την επόμενη ώρα.

Στη συνέχεια, αναμορφώνουμε τα δεδομένα εισόδου μας σωστά για να έχουμε σωστά τα βήματα και τα χαρακτηριστικά χρόνου.

Η προσαρμογή του μοντέλου είναι ίδια.

Η μόνη άλλη μικρή αλλαγή είναι στον τρόπο αξιολόγησης του μοντέλου. Συγκεκριμένα, πώς ανασυνθέτουμε τις σειρές με 8 στήλες που είναι κατάλληλες για την αντιστροφή της λειτουργίας κλιμάκωσης για να πάρουμε τις μεταβλητές πίσω στην αρχική κλίμακα έτσι ώστε να μπορέσουμε να υπολογίσουμε το RMSE.

Ακολουθεί και πάλι ένα γράφημα που δείχνει τις train και test απώλειες κατά τη διάρκεια της εκπαίδευσης.



Εικόνα 66: Γραφική παράσταση των επιδόσεων του LSTM μοντέλου με πολλά timesteps για το train και το test dataset

Τέλος, εκτυπώνεται το Test RMSE, που δεν δείχνει κανένα πλεονέκτημα στην ικανότητα, τουλάχιστον σε αυτό το πρόβλημα.

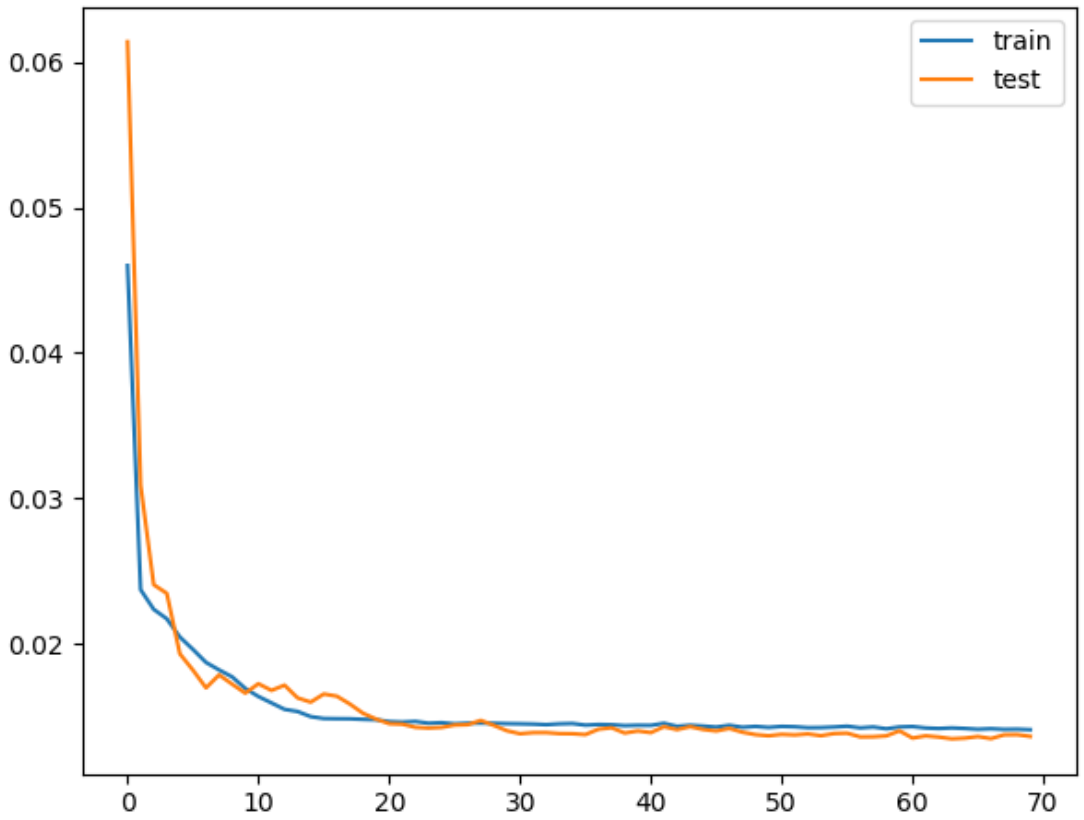
```
...
Epoch 46/50
- 2s - loss: 0.0143 - val_loss: 0.0143
Epoch 47/50
- 2s - loss: 0.0144 - val_loss: 0.0141
Epoch 48/50
- 2s - loss: 0.0143 - val_loss: 0.0141
Epoch 49/50
- 2s - loss: 0.0143 - val_loss: 0.0139
Epoch 50/50
- 2s - loss: 0.0143 - val_loss: 0.0145
Test RMSE: 26.271
```

Επαναλαμβάνουμε, και πάλι, τη διαδικασία κάνοντας μικροαλλαγές στη δομή του μοντέλου. Τα αποτελέσματα παρουσιάζονται στον ακόλουθο πίνακα.

Πίνακας 14. Απόδοση multiple timesteps LSTM μοντέλων

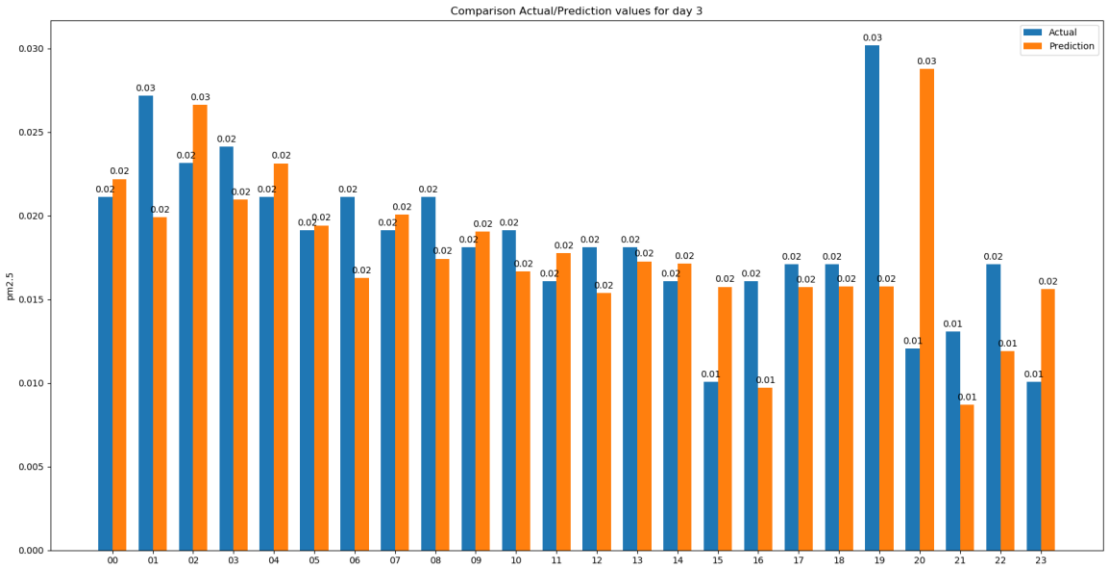
LSTM MODEL RESULTS									
N_hours	3	3	3	3	1	4	3	3	3
N_features	8	8	8	8	8	8	8	8	8
Neurons	50	50	50	100	100	100	200	100	100
Epochs	50	100	50	70	70	70	70	70	70
Batch size	72	72	24	72	72	72	72	24	96
Verbose	2	2	2	2	2	2	2	2	2
Loss func	Mae	Mae	Mae	Mae	Mae	Mae	Mae	Mae	Mae
Optimizer	adam	adam	adam	adam	adam	adam	adam	adam	adam
Dense	1	1	1	1	1	1	1	1	1
RMSE	26.271	26.615	29.471	26.250	26.383	26.329	26.457	27.845	27.755

Το αντίστοιχο διάγραμμα για το βέλτιστο μοντέλο LSTM που βρέθηκε (RMSE: 26.250) είναι το εξής:

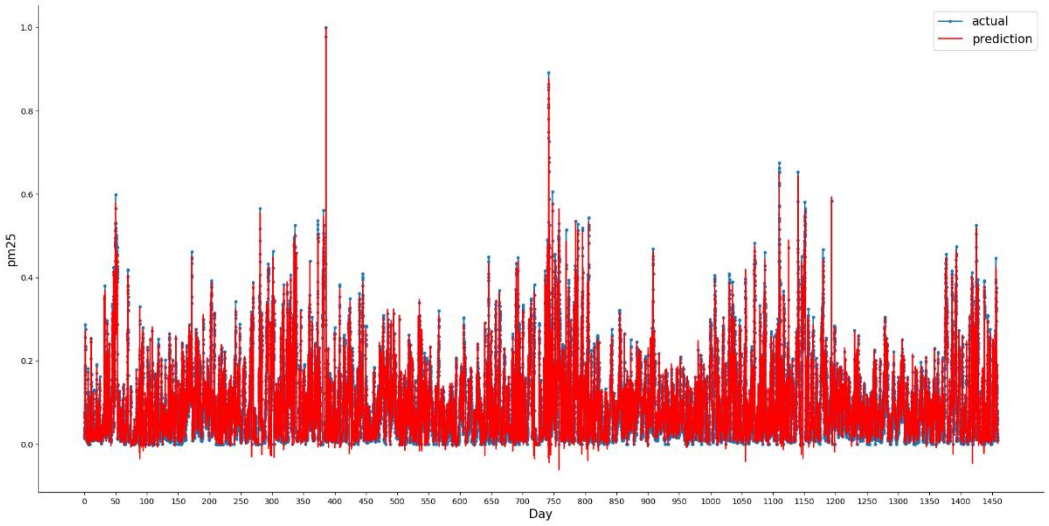


Εικόνα 67: Γραφική παράσταση των επιδόσεων του βέλτιστου LSTM μοντέλου με πολλά timesteps για το train και το test dataset

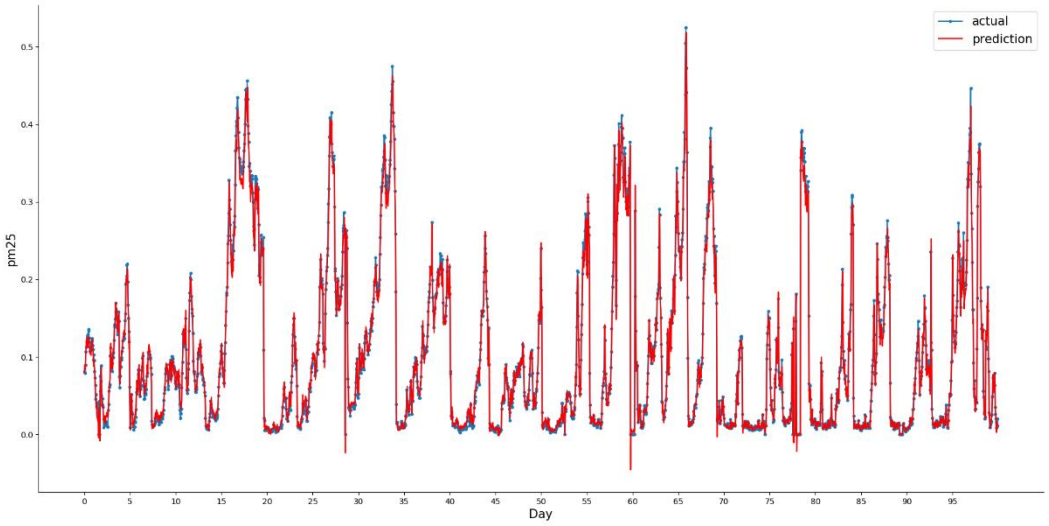
Ακολουθεί μια σύγκριση των προβλεπόμενων τιμών με τις πραγματικές για μια ενδεικτική μέρα. Οι διαφορές τους ποικίλουν από τρέξιμο σε τρέξιμο και από μέρα σε μέρα. Μια πιο συνολική εικόνα έχουμε στην Εικόνα 69 όπου παρουσιάζεται μια σύγκριση των προβλεπόμενων τιμών με τις πραγματικές για ολόκληρο το testing dataset, ενώ στην Εικόνα 70 εστιάζουμε στις 100 τελευταίες ημέρες του testing dataset για μια καλύτερη παρουσίαση των αποτελεσμάτων σύγκρισης.



Εικόνα 68: Σύγκριση των πραγματικών και των προβλεπόμενων τιμών του βέλτιστου multistep LSTM μοντέλου για το dataset, για τη 3η ημέρα από το testing dataset

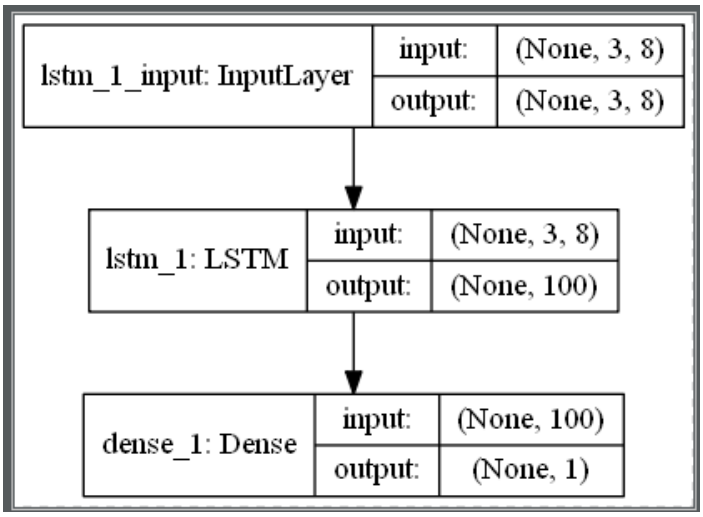


Εικόνα 69: Σύγκριση των πραγματικών και των προβλεπόμενων τιμών του βέλτιστου multistep LSTM μοντέλου για ολόκληρο το testing dataset



Εικόνα 70: Σύγκριση των πραγματικών και των προβλεπόμενων τιμών του βέλτιστου multistep LSTM μοντέλου για τις τελευταίες 100 ημέρες του testing dataset

Τέλος, δημιουργούμε ένα σχήμα με την αποτύπωση της δομής του multistep LSTM μοντέλου μας.



Εικόνα 71: Σχηματική αναπαράσταση του multistep LSTM μοντέλου του προγράμματος (μέσω Keras)

7.3 Δεδομένα Ποιότητας Ατμοσφαιρικού Αέρα

Το σύνολο δεδομένων πρόβλεψης της ποιότητας του ατμοσφαιρικού αέρα EMC, που παρέχονται από την Kaggle, αποτελούσαν τη βάση ενός παγκόσμιου διαγωνισμού πρόβλεψης. Αυτό το σύνολο περιγράφει ένα πρόβλημα πρόβλεψης πολλαπλών βημάτων, δεδομένης μιας χρονοσειράς πολλών παραγόντων, για πολλές φυσικές τοποθεσίες και

απαιτεί την πρόβλεψη των μετρήσεων ποιότητας του αέρα στις επόμενες ημέρες.

Συγκεκριμένα, οι παρατηρήσεις καιρού όπως η θερμοκρασία, η πίεση, η ταχύτητα του ανέμου και η κατεύθυνση του ανέμου παρέχονται ανά ώρα για 8 ημέρες για πολλαπλές φυσικές τοποθεσίες. Ο στόχος είναι να προβλεφθούν μετρήσεις ποιότητας του αέρα για τις επόμενες 3 ημέρες σε αυτές τις φυσικές τοποθεσίες. Οι χρόνοι παράδοσης των προβλέψεων δεν είναι συνεχόμενοι. Αντίθετα, πρέπει να προβλεφθούν συγκεκριμένοι χρόνοι παράδοσης για την περίοδο των 72 ωρών πρόβλεψης. Αυτοί είναι:

+1, +2, +3, +4, +5, +10, +17, +24, +48, +72.

Επιπλέον, το σύνολο δεδομένων χωρίζεται σε ξεχωριστά, αλλά συνεχόμενα πακέτα δεδομένων, με δεδομένα 8 ημερών ακολουθούμενα από 3 ημέρες που απαιτούν πρόβλεψη.

Γενικές παρατηρήσεις στο σύνολο των δεδομένων:

- Δεν είναι διαθέσιμες όλες οι μετρήσεις για τις καιρικές συνθήκες και την ποιότητα του αέρα για όλες τις τοποθεσίες. (incomplete data)
- Δεν έχουν όλες οι διαθέσιμες μετρήσεις των παραμέτρων πλήρες ιστορικό. (missing data)
- Για κάθε πρόβλεψη, τα inputs του μοντέλου αποτελούνται από παρατηρήσεις καιρικών συνθηκών πολλών παραμέτρων. (multivariate inputs)
- Τα outputs του μοντέλου είναι μια μη συνεχή ακολουθία προβλεπόμενων μετρήσεων ποιότητας του ατμοσφαιρικού αέρα. (multistep outputs)
- Το μοντέλο πρέπει να παράγει μια πρόβλεψη πολλών βημάτων που αφορά πολλές φυσικές τοποθεσίες. (multisite outputs)

Υπάρχουν 4 αρχεία διαθέσιμα για τη παραγωγή του μοντέλου προβλέψεων.

- SiteLocations.csv
- SiteLocations_with_more_sites.csv
- TrainingData.csv
- SubmissionZerosExpertNAs.csv

Εστιάζουμε στο αρχείο Trainingdata.csv όπου περιέχει το σύνολο των δεδομένων εκπαίδευσης. Συγκεκριμένα περιλαμβάνει τα δεδομένα σε πακέτα (chucks) όπου κάθε πακέτο είναι 8 συνεχείς ημέρες παρατηρήσεων και μεταβλητών στόχων.
Τα δεδομένα παρουσιάζονται με μη κανονικοποιημένο τρόπο. Κάθε σειρά δεδομένων περιέχει ένα σύνολο μετεωρολογικών μετρήσεων για μια ώρα,

για πολλαπλές τοποθεσίες, καθώς και τους στόχους ή τα αποτελέσματα για κάθε τοποθεσία για εκείνη την ώρα.

Οι μετρήσεις περιλαμβάνουν:

- Πληροφορίες χρόνου, όπως του αριθμού του πακέτου (chuckID), της θέσης μέσα στο πακέτο (position_within_chuck), του πιο συχνά εμφανιζόμενου μήνα (month_most_common), της ημέρας της εβδομάδας (weekday) και της ώρας της ημέρας (hour). Στήλες 0 έως 5
- Ηλιακή ακτινοβολία (Solar.radiation)
- Μετρήσεις ανέμου όπως κατεύθυνση (WindDirection) και ταχύτητα (WindSpeed).
- Μετρήσεις θερμοκρασίας, όπως η ελάχιστη (Ambient.Min.Temperature) και η μέγιστη θερμοκρασία περιβάλλοντος(Ambient.Max.Temperature) .
- Μετρήσεις πίεσης, όπως η ελάχιστη (Sample.Min.Baro.Pressure) και η μέγιστη βαρομετρική πίεση(Sample.Max.Baro.Pressure) . Συνολικά οι μετεωρολογικές μεταβλητές είναι 50. Και αφορούν 23 διαφορετικές φυσικές τοποθεσίες στην περιοχή του Σικάγου.
- Οι μεταβλητές-στόχοι είναι μια συλλογή διαφορετικών μετρήσεων ποιότητας ή ρύπανσης του αέρα σε διαφορετικές φυσικές τοποθεσίες (target_1_57, target_10_4002 , target_10_8003 κ.α.). Αυτές συνολικά είναι 39. Ο πρώτο κωδικός αναφέρεται σε κάποιο ρύπο, ενώ ο δεύτερος στο κωδικό τις φυσικής τοποθεσίας.

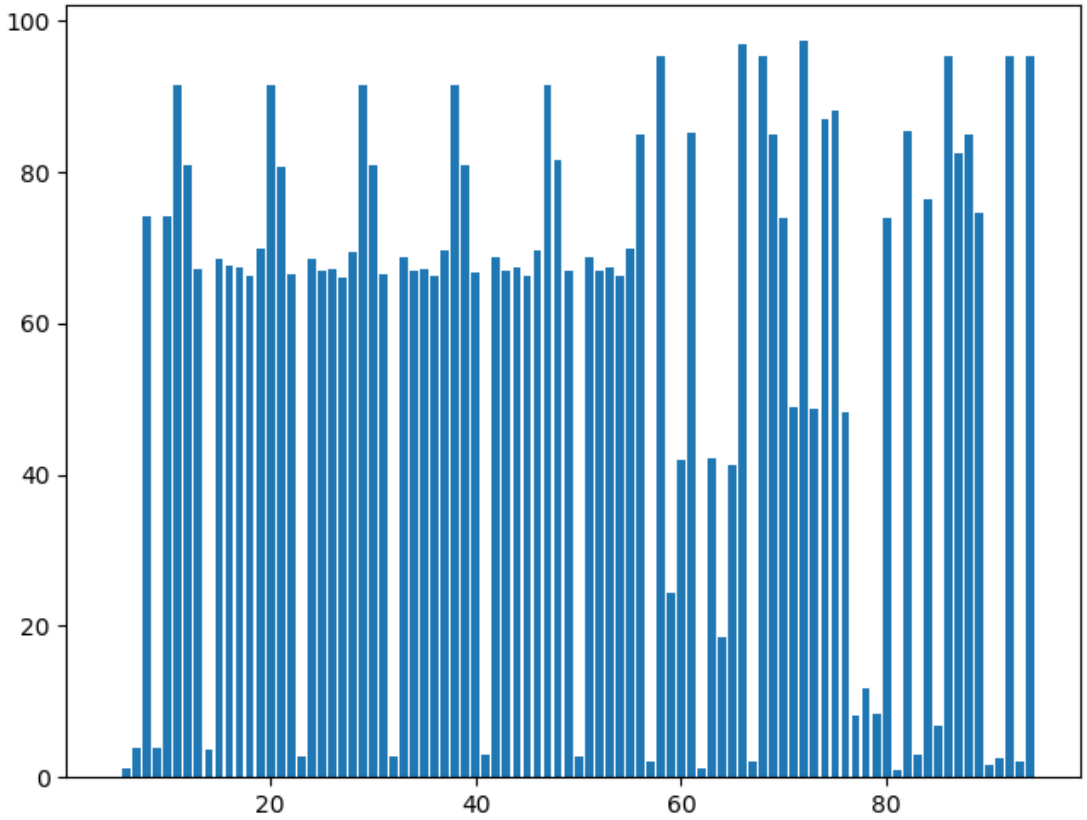
Δεν υπάρχουν όλες οι τοποθεσίες σε όλες τις μετρήσεις καιρού και δεν αφορούν όλες τις τοποθεσίες με όλες τις μεταβλητές-στόχους. Επιπλέον, για τις καταγεγραμμένες μεταβλητές λείπουν αρκετές τιμές, οι οποίες επισημαίνονται ως NA και η Pandas θα μετατρέψει αυτόματα σε NumPy.NaN.
Επιπλέον, η στήλη «weekday» περιέχει την ημέρα ως συμβολοσειρά (string), ενώ όλα τα άλλα δεδομένα είναι αριθμητικά (numeric).

Φορτώνουμε το αρχείο δεδομένων στη μνήμη χρησιμοποιώντας τη συνάρτηση Pandas read_csv(). Σε πρώτη φάση, Μπορούμε επίσης να έχουμε μια γρήγορη ιδέα για το πόσα NA δεδομένα υπάρχουν στο σύνολο δεδομένων. Το κάνουμε αυτό αρχικά περικόπτοντας τις πρώτες στήλες για να αφαιρέσουμε τα δεδομένα της εβδομάδας των συμβολοσειρών και να μετατρέψουμε τις υπόλοιπες στήλες σε τιμές float. Στη συνέχεια, μπορούμε να υπολογίσουμε τον συνολικό αριθμό των ελλιπών παρατηρήσεων και το ποσοστό των τιμών που λείπουν.

(37821, 95)
Total Missing: 1443977/3366069 (42.9%)

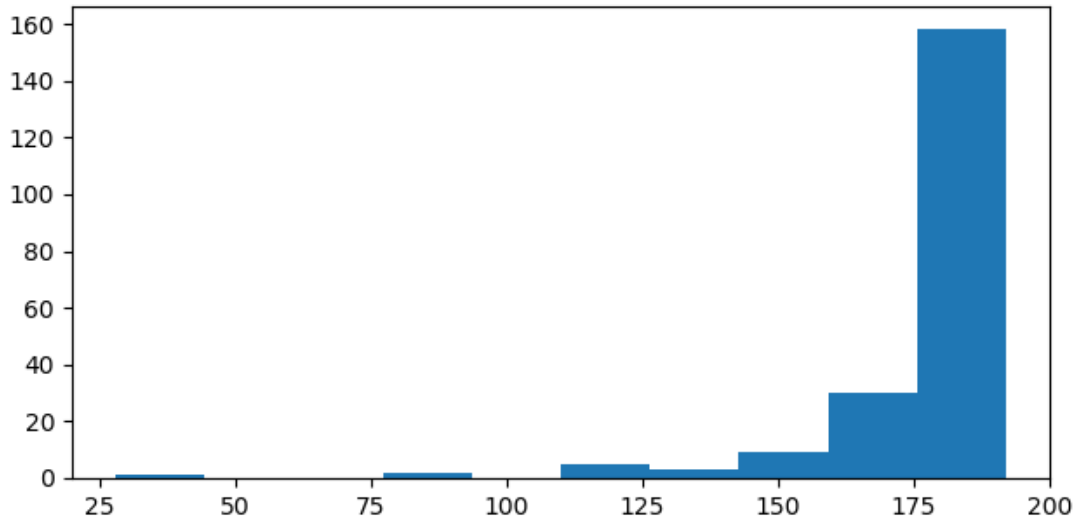
Παρατηρούμε ότι έχουμε περίπου 37.000 σειρές και 95 στήλες. Υπενθυμίζουμε ωστόσο ότι τα δεδομένα στην πραγματικότητα χωρίζονται σε πακέτα και οι στήλες χωρίζονται στις ίδιες παρατηρήσεις σε διαφορετικές τοποθεσίες.

Μπορούμε επίσης να δούμε ότι λίγο περισσότερο από το 40% των δεδομένων λείπει.
Πιο συγκεκριμένα, βλέπουμε από το παρακάτω διάγραμμα, ότι τουλάχιστον 12 μεταβλητές είναι πάνω από 90% με κενές τιμές.

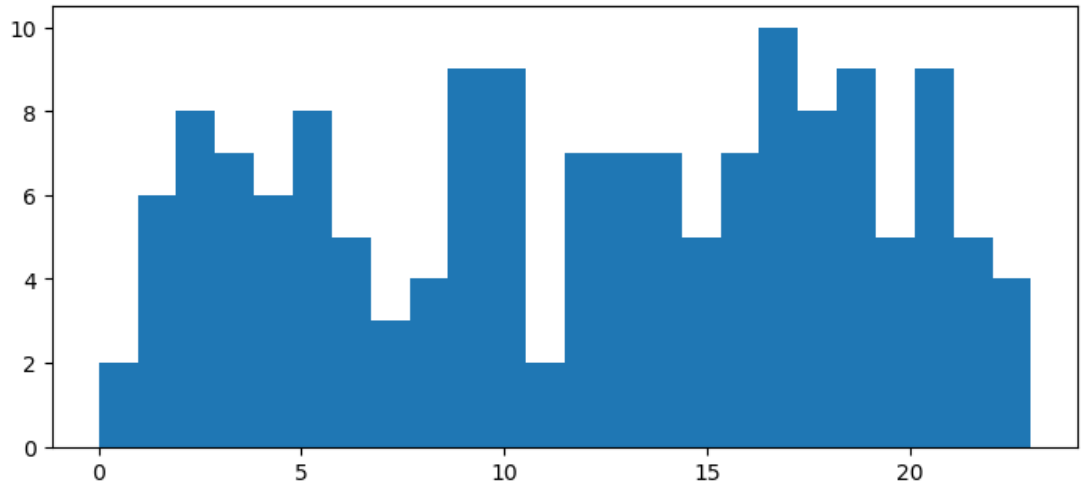


Εικόνα 72: Ποσοστό των κενών τιμών ανά στήλη στο σύνολο του dataset.

Μπορούμε να ομαδοποιήσουμε δεδομένα από τη μεταβλητή chunkID (δείκτης στήλης 1). Εάν κάθε chuck περιέχει οκτώ ημέρες και οι παρατηρήσεις είναι ωριαίες, τότε θα περιμέναμε (8 * 24) ή 192 σειρές δεδομένων ανά chuck. Εάν υπάρχουν 37.821 σειρές δεδομένων, τότε πρέπει να υπάρχουν chucks με περισσότερες ή λιγότερες από 192 ώρες, καθώς 37.821 / 192 είναι περίπου 196.9 chucks. Ας δούμε πρώτα τα δεδομένα σε chucks. Μπορούμε πρώτα να λάβουμε μια λίστα με τα μοναδικά αναγνωριστικά των chucks.



Εικόνα 73: Κατανομή των ωρών που περιέχονται στα πακέτα



Εικόνα 74: Κατανομή των ωριαίων μετρήσεων για μια μέρα για κάθε πακέτο

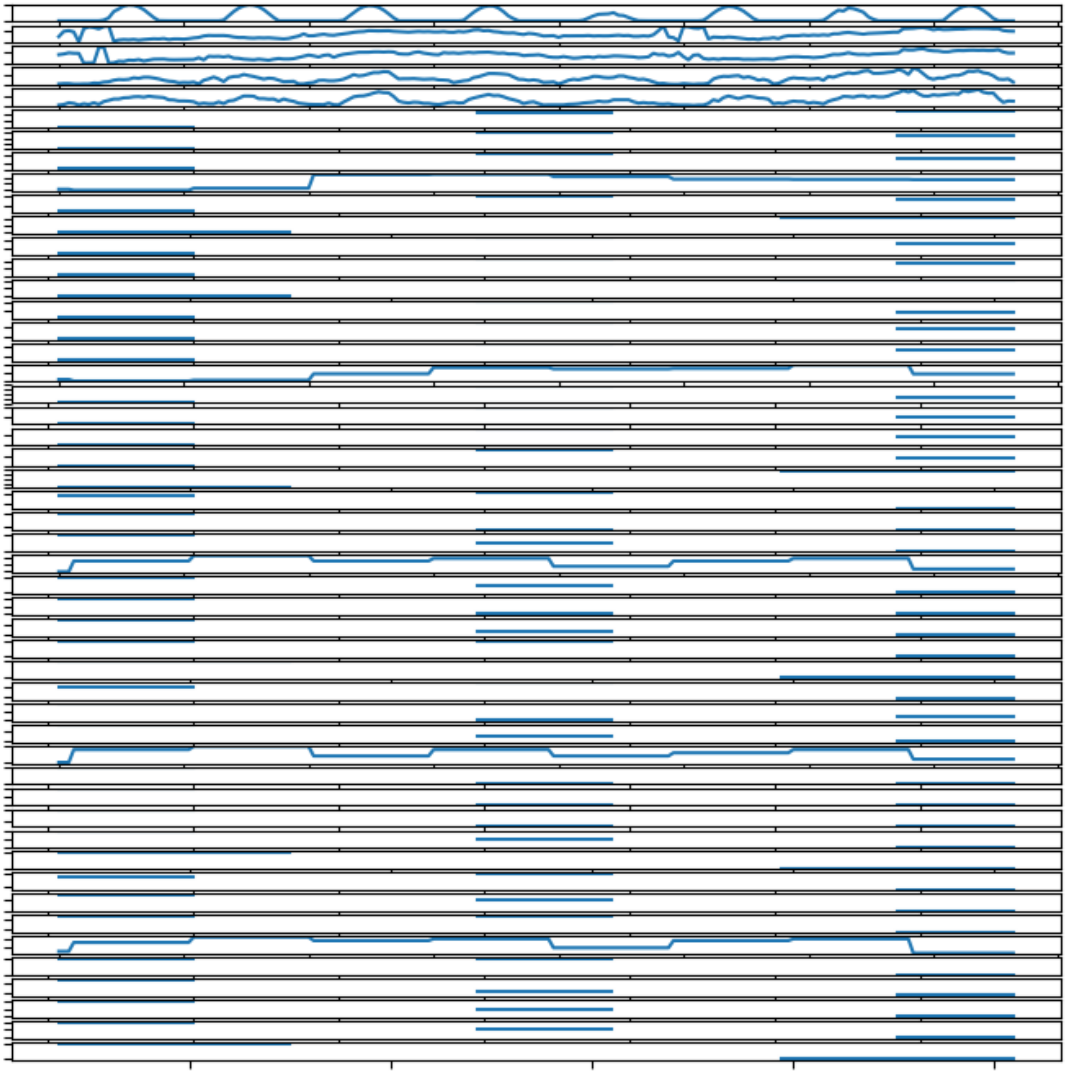
7.3.1 Προετοιμασία των δεδομένων

Ομαδοποιήσουμε τα δεδομένα με βάση τη μεταβλητή chuckID, που βρίσκεται στη στήλη 1. Για το σκοπό αυτό, παίρνουμε σε πρώτη φάση τα μοναδικά chuckIDs που περιέχονται στο αρχείο μας και τα αποθηκεύουμε σε έναν NumPy πίνακα (chuck_ids). Στη συνέχεια, συλλέγουμε όλες τις σειρές για κάθε πακέτο (chuck) και να τις αποθηκεύσουμε σε ένα λεξικό (chucks) για γρηγορότερη πρόσβαση στα δεδομένα. Μπορούμε να συγκεντρώσουμε τα παραπάνω βήματα σε μια συνάρτηση to_chucks().

Τρέχοντας το πρόγραμμα βρίσκουμε ότι ο συνολικός αριθμός των πακέτων (chucks) είναι 208.

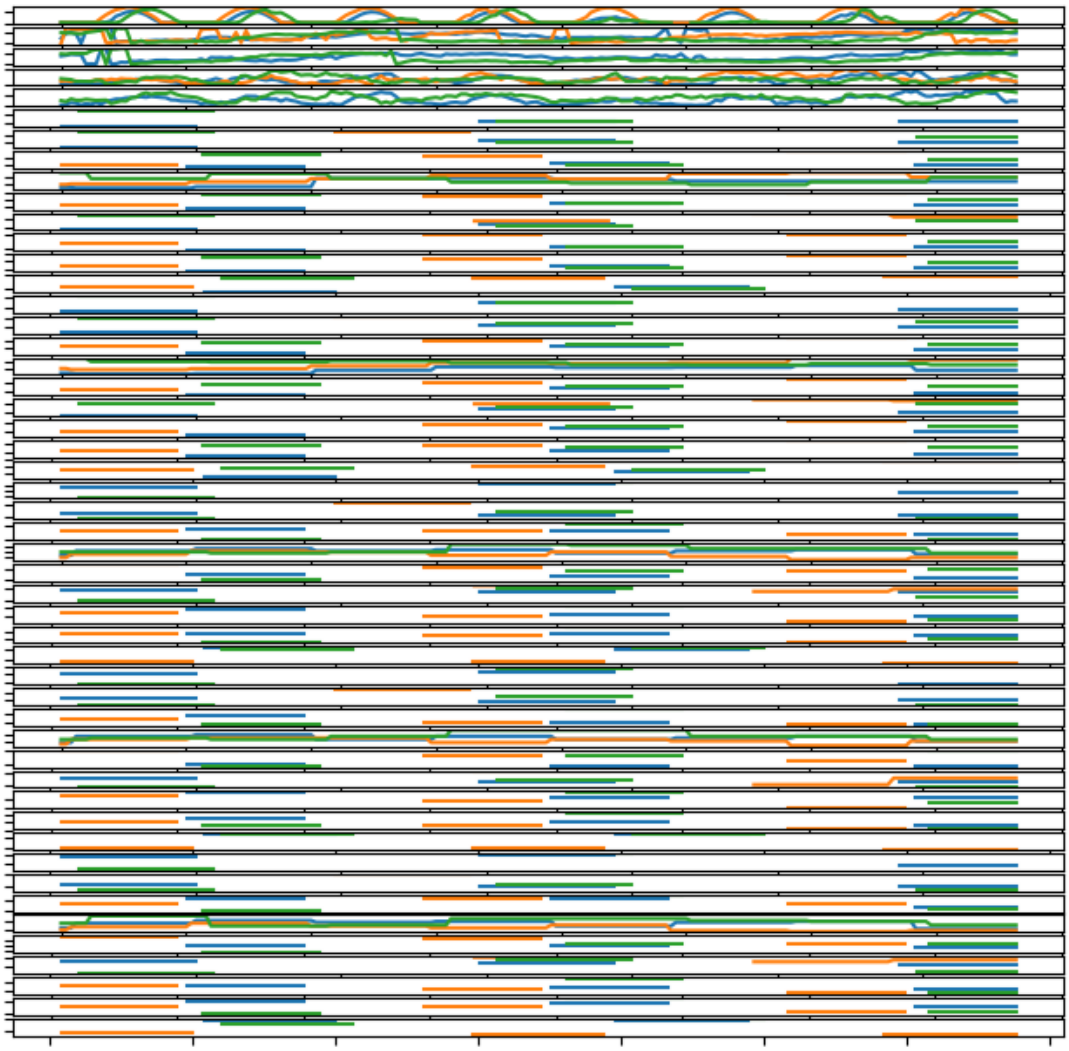
Από τη γραφική παράσταση όλων των μεταβλητών εισόδου για το πρώτο πακέτο των δεδομένων μπορούμε να δούμε ότι οι παρατηρήσεις

για τις πρώτες πέντε μεταβλητές φαίνονται αρκετά πλήρεις. Αυτές είναι η ηλιακή ακτινοβολία, η ταχύτητα ανέμου και η κατεύθυνση του ανέμου. Οι υπόλοιπες μεταβλητές φαίνονται αρκετά αποσπασματικές, τουλάχιστον για αυτό το πακέτο.



Εικόνα 75: Η γραφική παράσταση όλων των μεταβλητών εισόδου για το πρώτο πακέτο των δεδομένων.

Από τη γραφική παράσταση όλων των μεταβλητών εισόδου για τα τρία πρώτα πακέτα των δεδομένων, βλέπουμε παρόμοια συμπεριφορά.



Εικόνα 76: Η γραφική παράσταση όλων των μεταβλητών εισόδου για τα τρία πρώτα πακέτα των δεδομένων.

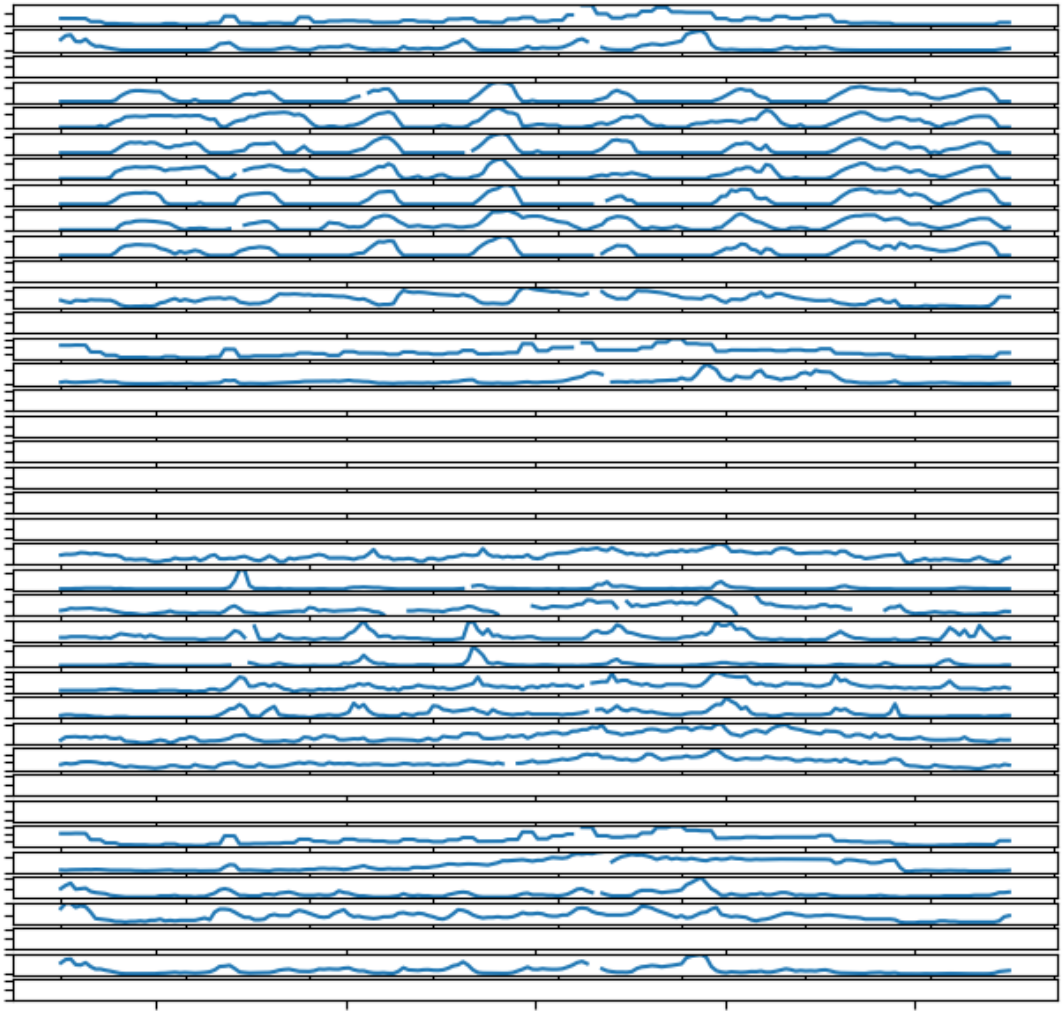
Από τη γραφική παράσταση όλων των μεταβλητών στόχων για το πρώτο πακέτο των δεδομένων μπορούμε να δούμε ότι υπάρχουν περισσότερες από μερικές μεταβλητές που δεν έχουν δεδομένα για αυτό το πακέτο. Αυτές δεν μπορούν να προβλεφθούν άμεσα, και πιθανόν ούτε έμμεσα.

Μπορούμε επίσης να δούμε κενά σε ορισμένες από τις σειρές λόγω των τιμών που λείπουν. Αυτό υποδηλώνει ότι παρόλο που μπορούμε να έχουμε παρατηρήσεις για κάθε βήμα χρόνου στο πακέτο, μπορεί να μην έχουμε μια συνεχόμενη σειρά για όλες τις μεταβλητές στο πακέτο.

Υπάρχει κυκλική δομή σε πολλές από τις σειρές. Οι περισσότερες έχουν οκτώ κορυφές, πολύ πιθανόν να αντιστοιχούν στις οκτώ ημέρες των

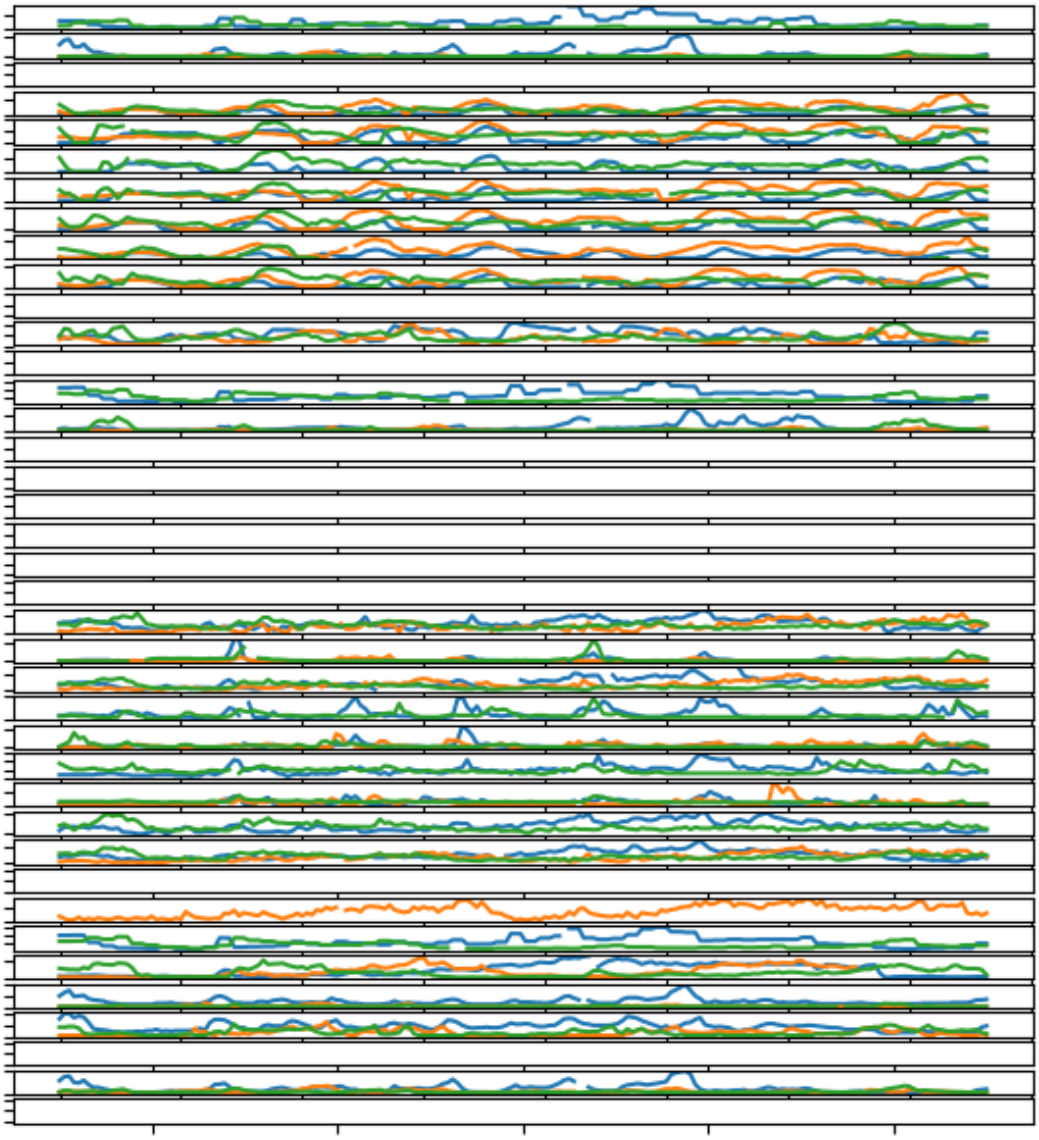
παρατηρήσεων μέσα στο πακέτο. Αυτή η εποχιακή δομή θα μπορούσε να μοντελοποιηθεί άμεσα, και ίσως να αφαιρεθεί από τα δεδομένα κατά τη μοντελοποίηση και να προστεθεί ξανά στο προβλεπόμενο διάστημα.

Τέλος, δεν φαίνεται να υπάρχει τάση στη σειρά.



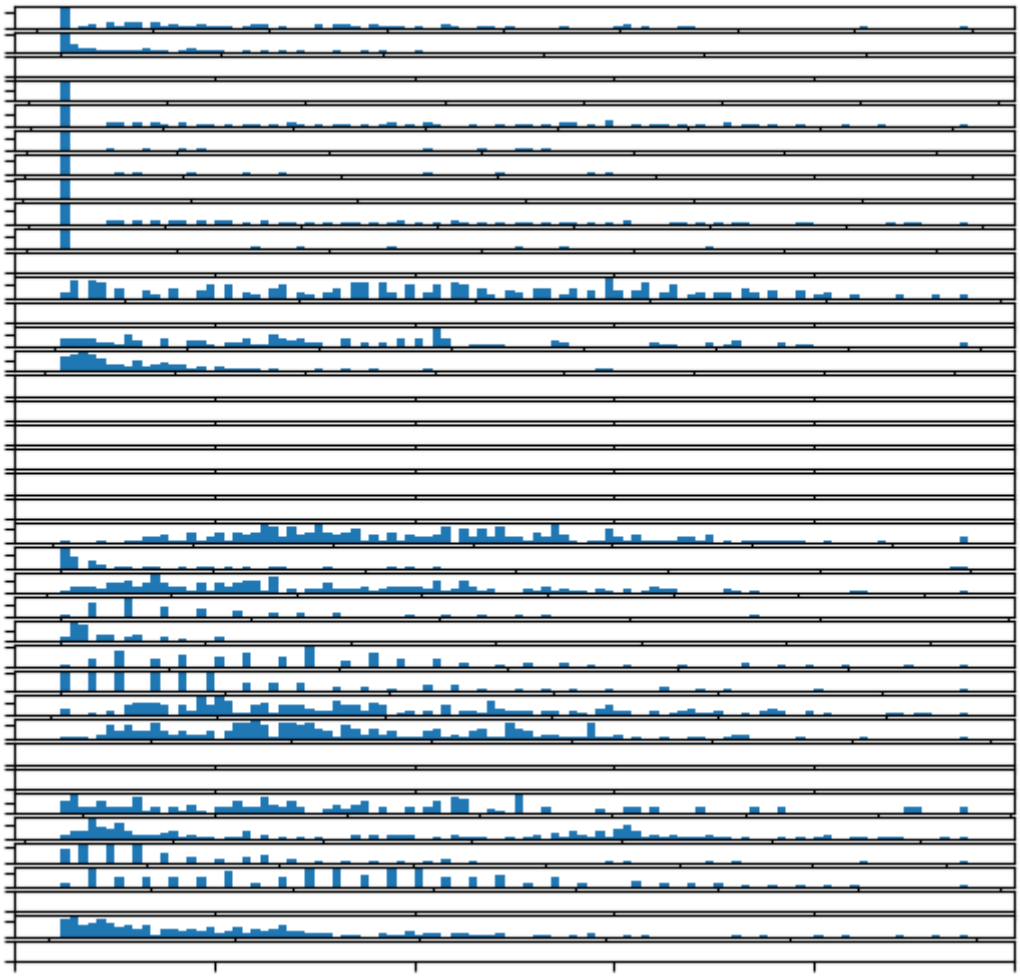
Εικόνα 77: Η γραφική παράσταση όλων των μεταβλητών στόχων για το πρώτο πακέτο των δεδομένων.

Από τη γραφική παράσταση όλων των μεταβλητών στόχων για τα τρία πρώτα πακέτα των δεδομένων, μπορούμε να δούμε ότι πολλές από τις μεταβλητές έχουν μια κυκλική ημερήσια δομή, και ότι αυτή επαναλαμβάνεται στα πακέτα.



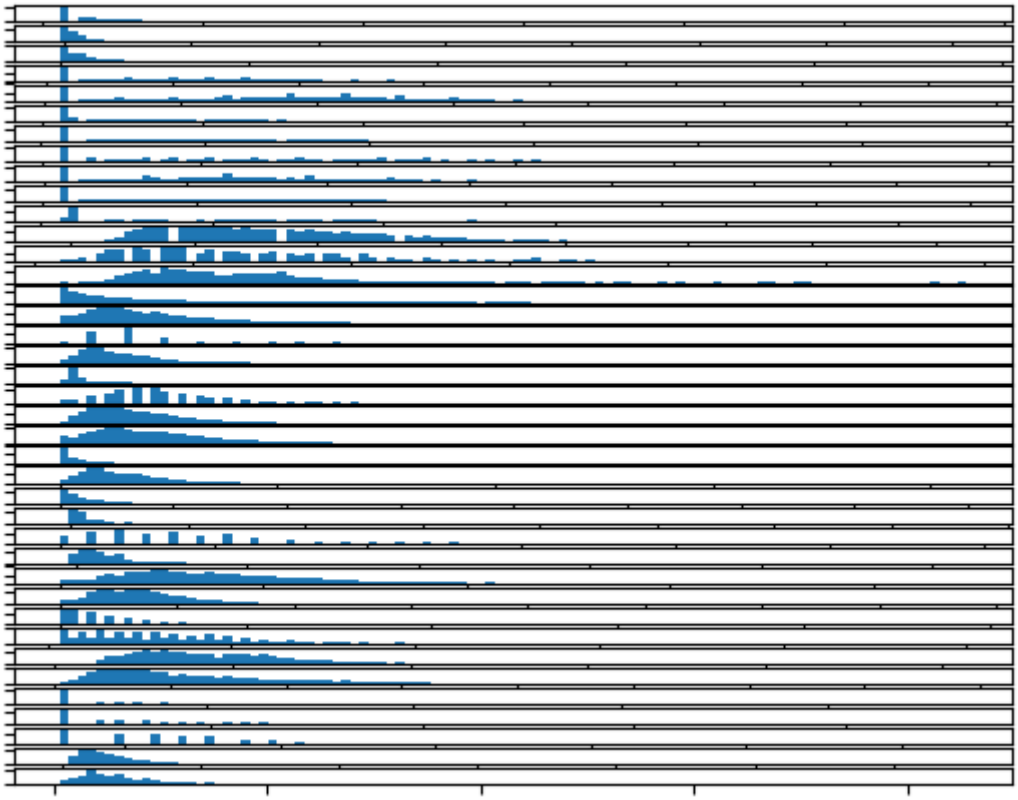
Εικόνα 78: Η γραφική παράσταση όλων των μεταβλητών στόχων για τα τρία πρώτα πακέτα των δεδομένων.

Από την κατανομή των μεταβλητών στόχων για ένα πακέτο δεν μπορούμε να πούμε πως έχουν κάποια προφανή Γκαουσιανή κατανομή.



Εικόνα 79: Η κατανομή όλων των μεταβλητών στόχων για ένα πακέτο.

Από την κατανομή των μεταβλητών στόχων για ολόκληρο το dataset, παρατηρούμε κάποιες να έχουν μια σχεδόν Γκαουσιανή κατανομή.



Εικόνα 80: Η κατανομή όλων των μεταβλητών στόχων για ολόκληρο το dataset.

7.3.1.1 Διαχωρισμός δεδομένων σε training και testing datasets

Το επόμενο βήμα είναι ο διαχωρισμός των δεδομένων σε train data και test data.

Κάθε πακέτο (chuck) καλύπτει ένα διάστημα 8 ημερών από ωριαίες παρατηρήσεις, αν και ο αριθμός των πραγματικών παρατηρήσεων μέσα σε κάθε πακέτο, μπορεί να διαφέρει ευρέως.

Μπορούμε να χωρίσουμε κάθε πακέτο στις 5 πρώτες μέρες με σκοπό το training και τις τελευταίες 3 μέρες για σκοπό testing.

Κάθε παρατήρηση έχει μια σειρά που ονομάζεται 'position_within_chunk' και κυμαίνεται από 1 έως 192 (8 ημέρες * 24 ώρες). Επομένως, μπορούμε να πάρουμε όλες τις σειρές με μια τιμή στη στήλη αυτή μικρότερη ή ίση με 120 (5 ημέρες * 24 ώρες) ως δεδομένα training και τυχόν τιμές πάνω από 120 ως δεδομένα testing.

Επιπλέον, αν υπάρχουν πακέτα χωρίς κάποια παρατήρηση στα διαχωριζόμενα training και testing δεδομένα, μπορούμε να τα απορρίψουμε ως μη αξιοποιήσιμα.

Στα naive μοντέλα, μας ενδιαφέρουν μόνο οι μεταβλητές-στόχοι και καμιά από τις μετεωρολογικές παραμέτρους ως είσοδο. Επομένως, μπορούμε να περιορίσουμε τα training και τα testing δεδομένα ώστε να περιέχουν μόνο τις 39 μεταβλητές-στόχους για κάθε πακέτο, τη θέση στο πακέτο καθώς και την ώρα της παρατήρησης.

Η συνάρτηση split_train_test() υλοποιεί τη παραπάνω περιγραφή και δίνει ένα λεξικό από πακέτα το οποίο το διαχωρίζει σε λίστα για training και σε λίστα για testing.

Επίσης, δεν απαιτείται για το πρόβλημα ολόκληρο το σύνολο δεδομένων για testing, καθώς ζητούνται παρατηρήσεις για συγκεκριμένες χρονικές στιγμές κατά τη διάρκεια της περιόδου των 3 ημερών. Έτσι, μπορούμε να βάλουμε αυτές τις επερχόμενες χρονικές στιγμές σε μια συνάρτηση get_lead-times() προς διευκόλυνσή μας.

Στη συνέχεια, μπορούμε να μειώσουμε το σύνολο των δεδομένων για testing σε μόνο στα δεδομένα με τους προτεινόμενους χρόνους παράδοσης. Μπορούμε να το κάνουμε αυτό ελέγχοντας τη στήλη 'position_within_chunk' και χρησιμοποιώντας τον χρόνο παράδοσης ως αντιστάθμιση από το τέλος του συνόλου training δεδομένων, π.χ. 120 + 1, 120 + 2 κ.λπ. Εάν βρεθεί μια σειρά που ταιριάζει στο testing σετ, αποθηκεύεται, διαφορετικά παράγεται μια σειρά παρατηρήσεων NaN.

Η συνάρτηση to_forecasts() υλοποιεί το παραπάνω σκεπτικό και επιστρέφει έναν πίνακα NumPy με μία σειρά για κάθε χρόνο πρόβλεψης, για κάθε πακέτο.

Τέλος, ενώνουμε τις παραπάνω διαδικασίες και αποθηκεύουμε τα αποτελέσματα σε νέα αρχεία csv (naive_train.csv και naive_test.csv) για τα μετέπειτα βήματα.

Τρέχοντας το πρόγραμμα, βλέπουμε ότι το πακέτο με chuckID =69 αφαιρέθηκε από το σύνολο των δεδομένων επειδή δεν περιέχει επαρκή δεδομένα. Επίσης, βλέπουμε ότι έχουμε 42 στήλες σε καθένα από τα σύνολα training και testing. Μια για το chuckID, μια για το position_within_chuck, μια για την ώρα παρατήρησης και οι υπόλοιπες 39 αφορούν τις μεταβλητές-στόχους.

>dropping chunk=69: train=(0, 95), test=(28, 95)
Train Rows: (23514, 42)
Test Rows: (2070, 42)

Παρατηρούμε τη σημαντική μείωση του συνόλου των δεδομένων προς testing, όπου περιέχει σειρές με μόνο στους χρόνους παράδοσης των προβλέψεων.

7.3.1.2 Αξιολόγηση μοντέλων

Μόλις ολοκληρωθούν οι προβλέψεις, πρέπει να αξιολογηθούν.

Είναι χρήσιμο να υπάρχει μια απλούστερη μορφή κατά την αξιολόγηση των προβλέψεων. Για παράδειγμα, θα χρησιμοποιήσουμε την τρισδιάστατη δομή [chucks] [variables] [time], όπου variable είναι ο αριθμός της μεταβλητής-στόχου από 0 έως 38 και time είναι ο δείκτης χρόνου εκτέλεσης από 0 έως 9.

Τα μοντέλα θέλουμε να κάνουν προβλέψεις σε αυτή τη μορφή.

Μπορούμε επίσης να αναδιαρθρώσουμε το σύνολο testing δεδομένων για να έχουμε και αυτό το σύνολο προς σύγκριση. Η συνάρτηση *prepare_test_forecasts()* υλοποιεί αυτήν την ενέργεια.

Θα αξιολογήσουμε ένα μοντέλο χρησιμοποιώντας το mean absolute error (MAE). Αυτή είναι η μετρική που χρησιμοποιήθηκε στον διαγωνισμό και είναι μια λογική επιλογή δεδομένης της μη-Γκαουσιανής κατανομής των μεταβλητών-στόχων.

Αν για κάποιο χρόνο πρόβλεψης δεν περιέχονται δεδομένα στο testing σετ (π.χ. NaN), τότε δεν θα υπολογιστεί σφάλμα για αυτήν την πρόβλεψη. Εάν για κάποιο χρόνο πρόβλεψης περιέχονται δεδομένα στο testing σετ, αλλά δεν υπάρχουν δεδομένα πρόβλεψης, τότε το πλήρες μέγεθος της παρατήρησης θα ληφθεί ως σφάλμα. Τέλος, εάν το testing σετ έχει μια παρατήρηση και έχει γίνει μια πρόβλεψη, τότε η απόλυτη διαφορά θα καταγράφεται ως το σφάλμα.

Η συνάρτηση calculate_error() υλοποιεί αυτούς τους κανόνες και επιστρέφει το σφάλμα για μια συγκεκριμένη πρόβλεψη.

Τα σφάλματα αθροίζονται σε όλα τα πακέτα και σε όλους τους χρόνους πρόβλεψης, και μετέπειτα υπολογίζεται ο μέσος όρος τους. Υπολογίζεται το συνολικό MAE, όπως επίσης και το MAE για κάθε χρόνο πρόβλεψης. Αυτό μπορεί να βοηθήσει στην επιλογή μοντέλου γενικά, καθώς, ορισμένα μοντέλα μπορεί να αποδίδουν διαφορετικά σε διαφορετικούς χρόνους πρόβλεψης.

Η συνάρτηση evaluation_forecasts() υλοποιεί τη περιγραφή αυτή, υπολογίζοντας το μέσο MAE και το MAE για κάθε χρόνο πρόβλεψης στη ζητούμενη μορφή [chunk] [variable] [time].

Μόλις έχουμε την αξιολόγηση ενός μοντέλου, μπορούμε να το παρουσιάσουμε. Η συνάρτηση summarize_error() πρώτα εκτυπώνει μια σύνοψη της απόδοσης ενός μοντέλου και στη συνέχεια δημιουργεί μια γραφική παράσταση του MAE για κάθε χρόνο πρόβλεψης.

7.3.1.3 Συμπλήρωση κενών τιμών

Οι κλασσικές μέθοδοι χρονοσειρών απαιτούν οι χρονοσειρές να είναι πλήρεις, δηλαδή να μην υπάρχουν τιμές που να λείπουν σε αυτές. Επομένως, το πρώτο βήμα είναι να διερευνήσουμε πόσο πλήρεις ή ελλιπείς είναι οι μεταβλητές στόχοι των χρονοσειρών προς εξέταση.

Για μια δεδομένη μεταβλητή, μπορεί να υπάρχουν ελλείψεις παρατηρήσεων που ορίζονται από τις ελλείπουσες σειρές. Συγκεκριμένα, κάθε παρατήρηση έχει μια θέση στο πακέτο. Αναμένουμε κάθε πακέτο, στο σύνολο δεδομένων κατάρτισης, να έχει 120 παρατηρήσεις, με 'positions_within_chunk', από 1 έως 120.

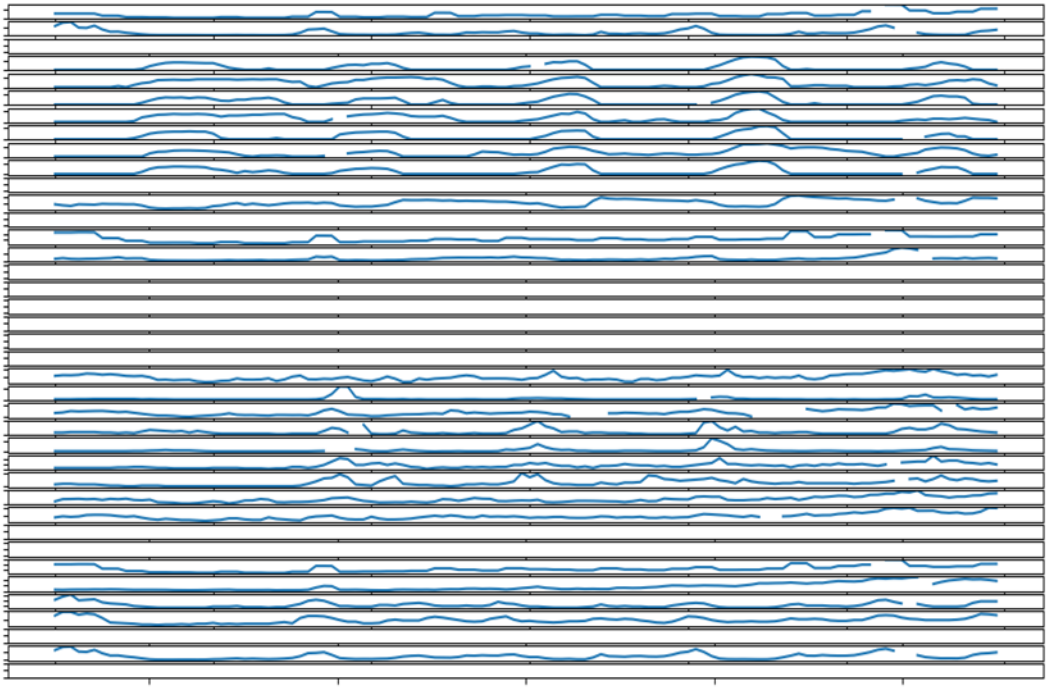
Επομένως, μπορούμε να δημιουργήσουμε ένα πίνακα τιμών 120 NaN για κάθε μεταβλητή, να σημειώσουμε όλες τις παρατηρήσεις στο πακέτο χρησιμοποιώντας τις τιμές 'positions_within_chunk' και οτιδήποτε απομείνει θα επισημανθεί με NaN. Στη συνέχεια, μπορούμε να αποτυπώσουμε γραφικά κάθε μεταβλητή και να αναζητήσουμε κενά.

Η συνάρτηση *variable_to_series()* θα λάβει τις σειρές για ένα πακέτο και ένα δεδομένο δείκτη στήλης για τη μεταβλητή-στόχο και θα επιστρέψει μια σειρά 120 βημάτων χρόνου για τη μεταβλητή, με όλα τα διαθέσιμα δεδομένα που σημειώνονται με τιμή από το πακέτο.

Στη συνέχεια, μπορούμε να καλέσουμε αυτή τη συνάρτηση για κάθε μεταβλητή-στόχο σε ένα πακέτο και να δημιουργήσουμε ένα γραμμικό γράφημα.

Η συνάρτηση plot_variables() θα το εφαρμόσει και θα δημιουργήσει ένα σχήμα με 39 γραμμικά γραφήματα στοιβαγμένα οριζόντια.

Τρέχοντας τον κώδικα, δημιουργείται μια γραφική παράσταση, με 39 γραμμικά γραφήματα, για κάθε μια μεταβλητή-στόχο του πρώτου πακέτου. (train_chuck[0])



Εικόνα 81: Γραφήματα για όλες τις μεταβλητές-στόχους του 1ου πακέτου

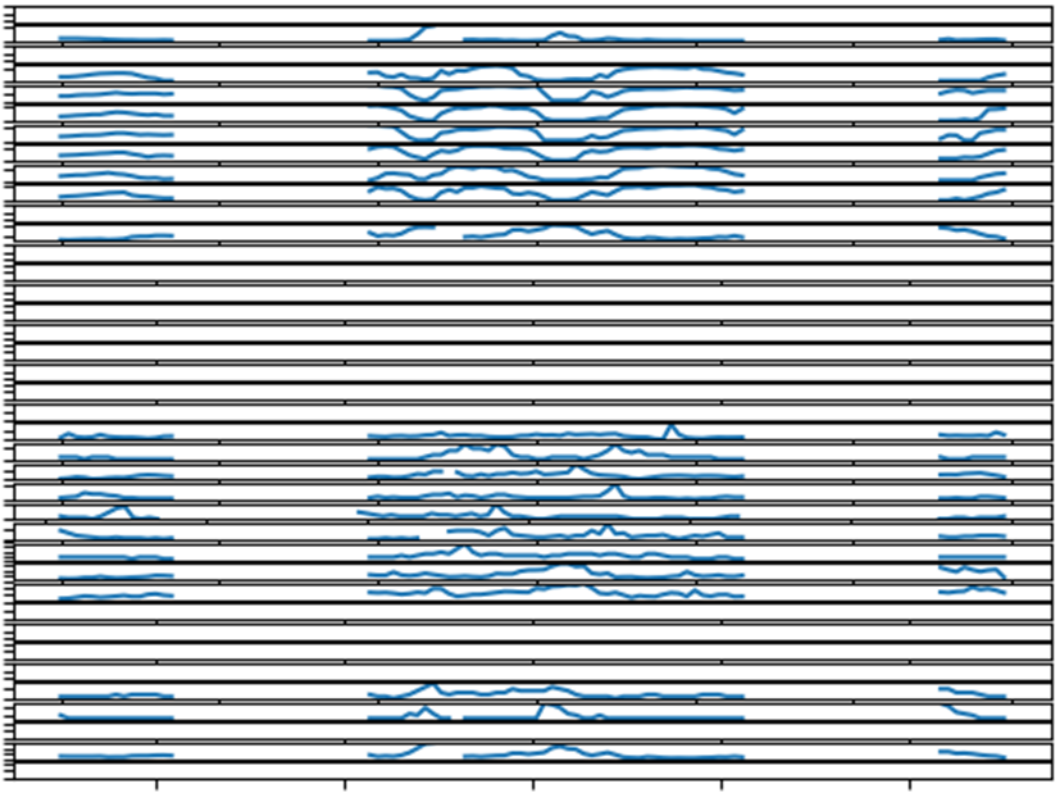
Μπορούμε να δούμε μια εποχική δομή σε πολλές από τις μεταβλητές-στόχους. Αυτό υποδηλώνει ότι μπορεί να είναι χρήσιμο να εκτελέσουμε μια 24ωρη εποχιακή διαφοροποίηση κάθε σειράς πριν από τη μοντελοποίηση.

Μπορούμε να δούμε ότι υπάρχουν μεταβλητές για τις οποίες δεν έχουμε δεδομένα. Αυτά μπορούν να εντοπιστούν και να αγνοηθούν καθώς δεν μπορούμε να τα μοντελοποιήσουμε ή να τα προβλέψουμε.

Μπορούμε να δούμε κενά σε πολλές από τις σειρές, αλλά τα κενά είναι σύντομα, διαρκούν για λίγες ώρες το πολύ. Σε αυτά θα μπορούσαν να αποδοθούν τιμές είτε με την παραμονή προηγούμενων τιμών ή τιμών στις ίδιες ώρες της ίδιας σειράς.

Κοιτάζοντας μερικά άλλα πακέτα τυχαία, πολλά καταλήγουν σε όμοια γραφήματα. Ωστόσο, αυτό δεν συμβαίνει πάντοτε.

Για παράδειγμα, τροποποιώντας ανάλογα των κώδικα, δημιουργούμε γραφικές παραστάσεις για τις μεταβλητές στόχους του τέταρτου πακέτου. (train_chuck[3])



Εικόνα 82: Γραφήματα για όλες τις μεταβλητές-στόχους του 4ου πακέτου

Σε αυτή τη περίπτωση, παρατηρούμε κενά στα δεδομένα που διαρκούν πολλές ώρες, ίσως μέχρι μία ημέρα ή και περισσότερο.

Αυτές οι χρονοσειρές απαιτούν δραματική επιδιόρθωση πριν μπορέσουν να χρησιμοποιηθούν σε ένα κλασσικό μοντέλο.

Η αντιστοίχιση των ελλειπόντων δεδομένων με τη χρήση συχνά εμφανιζόμενων τιμών ή παρατηρήσεων εντός της χρονοσειράς με την ίδια ώρα πιθανόν να μην είναι επαρκής. Ένας τρόπος είναι να συμπληρωθούν με τις μέσες τιμές, λαμβάνοντας υπόψη το σύνολο των δεδομένων εκπαίδευσης.

Υπάρχουν πολλοί τρόποι συμπλήρωσης των κενών δεδομένων χωρίς να μπορούμε να ξέρουμε εκ των προτέρων ποιος είναι ο βέλτιστος.

Αρχικά, πρέπει να υπολογίσουμε μια παράλληλη σειρά της ώρας της ημέρας για κάθε πακέτο που μπορούμε να χρησιμοποιήσουμε για να καταγράψουμε συγκεκριμένα δεδομένα ώρας για κάθε μεταβλητή στο πακέτο.

Δεδομένης μιας σειράς μερικώς γεμάτων ωρών ημέρας, η συνάρτηση *interpolate_hours()* θα συμπληρώσει με τιμές για τις ώρες της ημέρας που

λείπουν. Αυτό επιτυγχάνεται με την εύρεση της πρώτης ώρας με συμπληρωμένη τιμή, μετά την καταμέτρηση προς τα εμπρός, τη συμπλήρωση της κενής τιμής της ώρας της ημέρας και μετά την εκτέλεση της ίδιας λειτουργίας προς τα πίσω.

Χρησιμοποιούμε αυτή τη συνάρτηση για να προετοιμάσουμε μια σειρά ωριαίων τιμών για ένα πακέτο η οποία μπορεί να χρησιμοποιηθεί για να συμπληρωθούν οι τιμές που λείπουν από το πακέτο χρησιμοποιώντας πληροφορίες συγκεκριμένης ώρας.

Μπορούμε να καλέσουμε την ίδια συνάρτηση variable_to_series() από την προηγούμενη ενότητα για να δημιουργήσουμε τη σειρά ωριαίων τιμών για τις τιμές που λείπουν (στήλη 2), και στη συνέχεια να καλέσουμε τη συνάρτηση interpolate_hours() για να συμπληρώσουμε τα κενά.

Ας προσπαθήσουμε να συμπληρώσουμε τις τιμές που λείπουν σε ένα πακέτο με τιμές στην ίδια σειρά με την ίδια ώρα. Συγκεκριμένα, θα βρούμε όλες τις σειρές με την ίδια ώρα στη σειρά και υπολογίζουμε τη μέση τιμή.

Η συνάρτηση impute_missing() λαμβάνει όλες τις σειρές σε ένα πακέτο, την προετοιμασμένη ακολουθία ωρών της ημέρας για το πακέτο και τη σειρά με ελλείπουσες τιμές για μια μεταβλητή και το δείκτη της στήλης για μια μεταβλητή.

Πρώτα ελέγχει για να δει αν όλες οι σειρές λείπουν τα δεδομένα και επιστρέφουν αμέσως, αν αυτό συμβαίνει καθώς δεν μπορεί να γίνει συμπλήρωση. Στη συνέχεια απαριθμεί τα χρονικά βήματα της σειράς και όταν εντοπίζει ένα χρονικό βήμα χωρίς δεδομένα, συλλέγει όλες τις σειρές της σειράς με δεδομένα για την ίδια ώρα και υπολογίζει τη μέση τιμή.

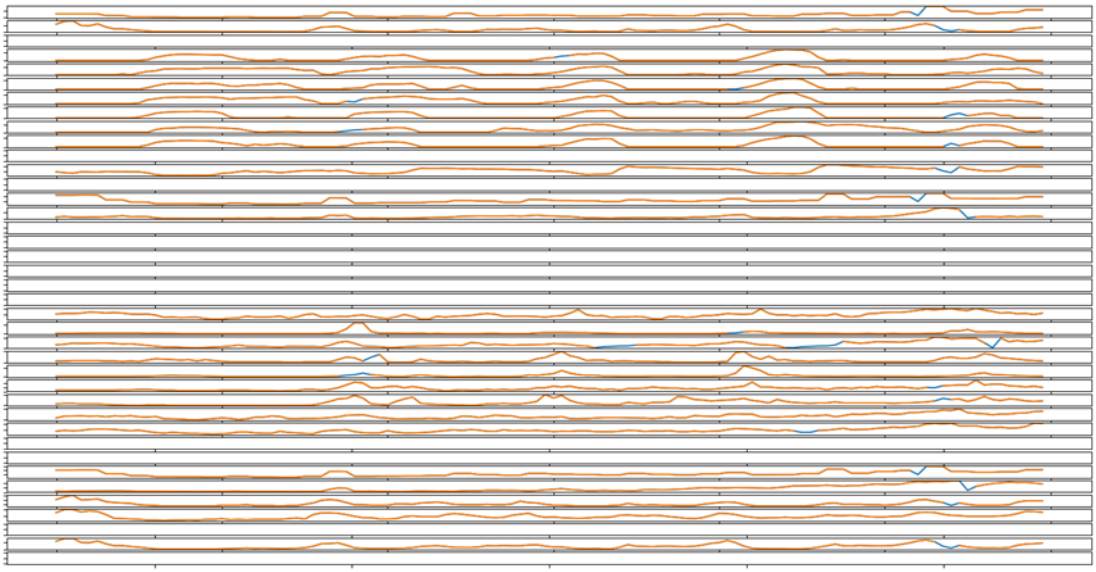
Για να δούμε τον αντίκτυπο αυτής της στρατηγικής συμπλήρωσης, μπορούμε να ενημερώσουμε τη συνάρτηση *plot_variables()* από την προηγούμενη ενότητα για να σχεδιάσουμε πρώτα την τεκμαρτή σειρά και στη συνέχεια να σχεδιάσουμε την αρχική σειρά με τιμές που λείπουν.

Αυτό θα επιτρέψει τις τεκμαρτές τιμές να λάμπουν μέσα στα κενά της αρχικής σειράς και μπορούμε να δούμε αν τα αποτελέσματα φαίνονται λογικά.

Η ενημερωμένη έκδοση της συνάρτησης *plot_variables()* με αυτή την αλλαγή, καλώντας τη συνάρτηση *impute_missing()* δημιουργεί την τεκμαρτή έκδοση της σειράς και λαμβάνοντας τις σειρές ωριαίων τιμών ως argument.

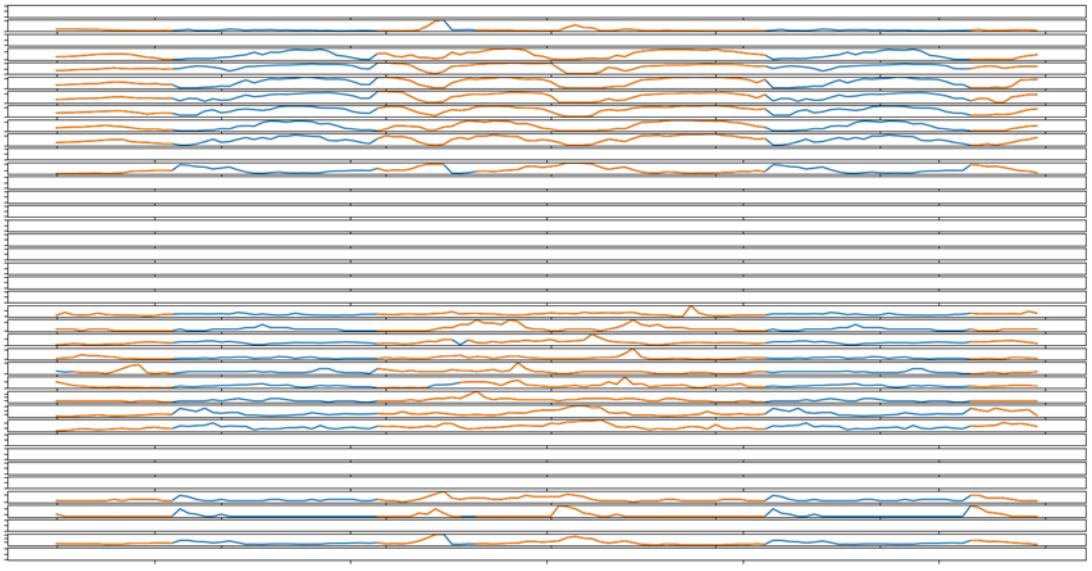
Όταν εκτελέσουμε το πρόγραμμα, αυτό δημιουργεί ένα σχήμα με 39 γραφήματα, ένα για κάθε μεταβλητή-στόχο στο πρώτο πακέτο του συνόλου των δεδομένων εκπαίδευσης.

Από το σχήμα βλέπουμε με πορτοκαλί χρώμα τα αρχικά δεδομένα και με μπλε έχουν σημειωθεί τα συμπληρωμένα κενά. Τα μπλε τμήματα φαίνονται λογικά.



Εικόνα 83: Γραφήματα για όλες τις μεταβλητές-στόχους του 1ου πακέτου. Με πορτοκαλί είναι οι υπάρχουσες τιμές του πακέτου και με μπλε οι συμπληρωμένες τιμές.

Ανάλογα αποτελέσματα φαίνονται και για το 4ο πακέτο του συνόλου των δεδομένων εκπαίδευσης, όπου έχουμε μεγάλα κενά από δεδομένα. Ακόμα και η εποχικότητα των μεταβλητών παρουσιάζεται αρκετά ικανοποιητικά.



Εικόνα 84: Γραφήματα για όλες τις μεταβλητές-στόχους του 4ου πακέτου. Με πορτοκαλί είναι οι υπάρχουσες τιμές του πακέτου και με μπλε οι συμπληρωμένες τιμές.

7.3.2 ARIMA

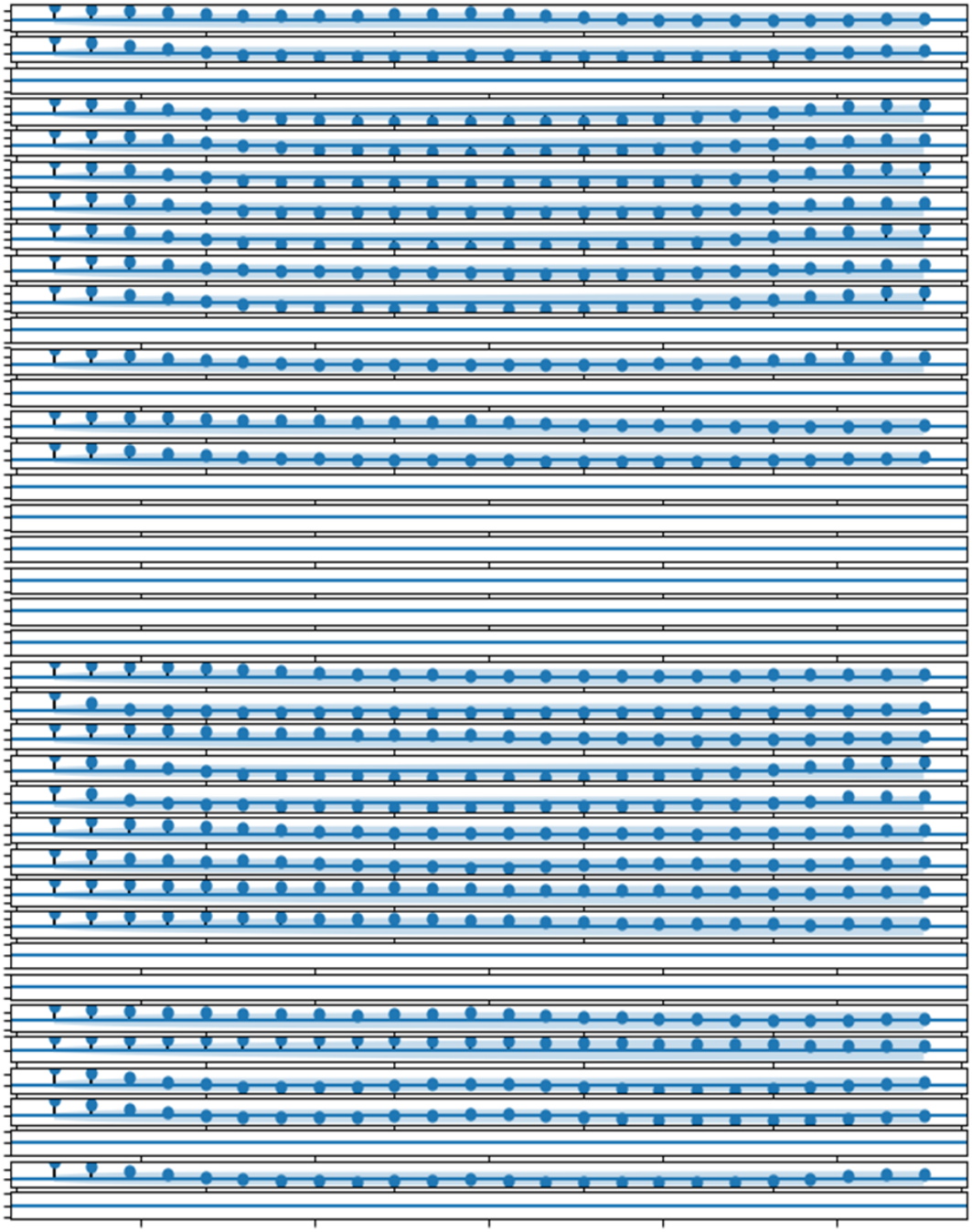
Τα διαγράμματα αυτοσυσχέτισης συνοψίζουν τη σχέση κάθε παρατήρησης με τις παρατηρήσεις των προηγούμενων βημάτων. Μαζί με ημιτελή διαγράμματα αυτοσυσχέτισης, αυτά μπορούν να χρησιμοποιηθούν για να προσδιορίσουν τη διαμόρφωση ενός μοντέλου ARMA.

Η βιβλιοθήκη statsmodels παρέχει τις συναρτήσεις plot_ac() και plot_pacf() που χρησιμεύουν για την δημιουργία γραφικών παραστάσεων ACF και PACF αντίστοιχα.

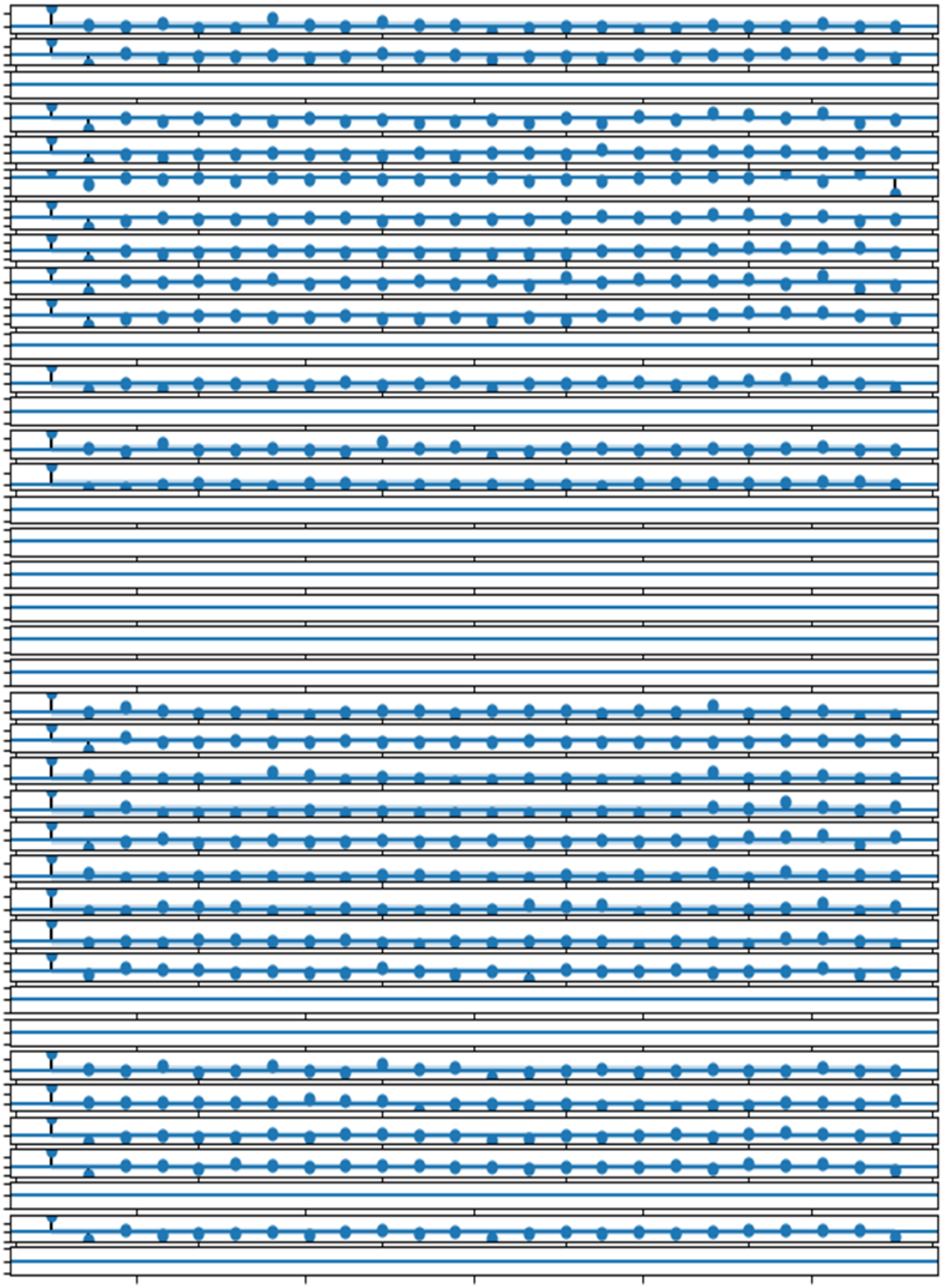
Ενημερώνοντας τη συνάρτηση plot_variables() για να δημιουργήσουμε αυτά τα διαγράμματα, ένα για κάθε τύπο για κάθε μία από τις 39 σειρές.

Στοιχίσουμε όλες τις γραφικές παραστάσεις ACF κάθετα αριστερά και όλες τις PACF κάθετα δεξιά. Δηλαδή φτιάχνουμε δύο στήλες 39 γραφημάτων. Θα περιορίσουμε τις υστερήσεις που γίνονται για τη δημιουργία των γραφημάτων σε 24 χρονικά βήματα (ώρες) και θα αγνοήσουμε τη συσχέτιση κάθε μεταβλητής με τον εαυτό της, καθώς είναι περιττή.

Με την εκτέλεση του προγράμματος, δημιουργείται ένα σχήμα με πολλά διαγράμματα για τις μεταβλητές-στόχους του πρώτου πακέτου των δεδομένων εκπαίδευσης.



Εικόνα 85: Διάγραμμα ACF για τις μεταβλητές-στόχους του πακέτου 1.



Εικόνα 86: Διάγραμμα PACF για τις μεταβλητές-στόχους του πακέτου 1.

Στο ACF σχήμα, στα περισσότερα διαγράμματα εμφανίζονται σημαντικές συσχετίσεις (τελείες πάνω από την περιοχή σημασίας) με υστερήσεις 1-2 βημάτων, ίσως και με υστερήσεις 1-3 βημάτων σε μερικές περιπτώσεις, παρουσιάζοντας μια αργή και σταθερή μείωση υστερήσεων.

Παρομοίως, στο PAFC σχήμα, μπορούμε να παρατηρήσουμε σημαντικές υστερήσεις με 1-2 βήματα, ωστόσο με σχετικά απότομη πτώση.

Αυτό δείχνει έντονα μια διαδικασία αυτοσυσχέτισης πιθανής τάξεως 1, 2 ή 3, π.χ. AR (3).

Στα ACF γραφήματα μπορούμε να δούμε έναν καθημερινό κύκλο στις συσχετίσεις. Αυτό μπορεί να υποδηλώνει κάποιο όφελος σε μια εποχιακή διαφοροποίηση των δεδομένων πριν από τη μοντελοποίηση ή τη χρήση ενός μοντέλου AR κατάλληλου για εποχική διαφοροποίηση.

Επαναλαμβάνοντας αυτή την ανάλυση των μεταβλητών-στόχων για άλλα πακέτα παρατηρούνται όμοια αποτελέσματα. Αυτό υποδεικνύει ότι ίσως είμαστε σε θέση να διαμορφώσουμε ένα γενικό AR μοντέλο για όλες τις σειρές σε όλα τα πακέτα.

7.3.2.1 Ανάπτυξη μοντέλου

Αρχικά θα πρέπει να φτιαχτεί μια γενική συνάρτηση με σκοπό τη πραγματοποίηση προβλέψεων για κάθε πακέτο.

Η συνάρτηση forecast_chucks() εκτελεί το σύνολο δεδομένων εκπαίδευσης με είσοδο τις στήλες (chuck_id,position_within_chuck, hour) για το σύνολο δεδομένων δοκιμών και επιστρέφει προβλέψεις για όλα τα πακέτα με την αναμενόμενη 3D μορφή [chunk] [variable] [time].

Η συνάρτηση απαριθμεί τα πακέτα στην πρόβλεψη και στη συνέχεια απαριθμεί τις 39 στήλες που περιέχουν τις μεταβλητές-στόχους, καλώντας τη συνάρτηση forecast_variable(), προκειμένου να πραγματοποιήσει μια πρόβλεψη για κάθε ζητούμενο χρόνο για μια δεδομένη μεταβλητή-στόχο.

Μπορούμε τώρα να εφαρμόσουμε μια έκδοση της συνάρτησης forecast_variable().

Για κάθε μεταβλητή, πρώτα ελέγχουμε την περίπτωση όπου δεν υπάρχουν δεδομένα (π.χ. όλα τα NaN) και αν ναι, επιστρέφουμε μια πρόβλεψη που έχει τη τιμή NaN για κάθε προγνωστική χρονική στιγμή.

Στη συνέχεια, δημιουργούμε μια σειρά από τη μεταβλητή χρησιμοποιώντας την συνάρτηση variable-to-series() και στη συνέχεια συμπληρώνοντας τις ελλείπουσες τιμές χρησιμοποιώντας το διάμεσο μέσα στη σειρά, καλώντας τη συνάρτηση impute_missing(). Αυτές έχουν αναλυθεί προηγουμένως.

Τέλος, καλείται η συνάρτηση fit_and_forecast() όπου ταιριάζει ένα μοντέλο και δημιουργεί προβλέψεις για τους ζητούμενους 10 χρόνους πρόβλεψης.

Αρχικά, πρέπει να καθορίσουμε το μοντέλο, συμπεριλαμβανομένης της σειράς της αυτορυθμιζόμενης διαδικασίας, όπως η AR(1).

Στη συνέχεια, το μοντέλο προσαρμόζεται στην καταλογισμένη σειρά. Απενεργοποιούμε τις περιττές πληροφορίες κατά τη διάρκεια της προσαρμογής, ρυθμίζοντας το disp σε False.

Το μοντέλο προσαρμογής χρησιμοποιείται στη συνέχεια για να προβλέψει τις επόμενες 72 ώρες μετά το τέλος της σειράς.

Μας ενδιαφέρουν μόνο συγκεκριμένοι χρόνοι, επομένως ετοιμάζουμε μια σειρά από αυτές τις χρονικές στιγμές, αφαιρούμε 1 για να τις μετατρέψουμε σε δείκτες πινάκων και στη συνέχεια να τις χρησιμοποιήσουμε για να επιλέξουμε τις τιμές των 10 χρόνων πρόβλεψης.

Τα statsmodels ARIMA μοντέλα χρησιμοποιούν βιβλιοθήκες γραμμικής άλγεβρας για να ταιριάζουν στο μοντέλο κάτω από τις απαιτήσεις και μερικές φορές η διαδικασία προσαρμογής μπορεί να είναι ασταθής σε ορισμένα δεδομένα. Ως εκ τούτου, μπορεί να πετάξει μια εξαίρεση ή να αναφέρει πολλές προειδοποιήσεις.

Θα συμπεριλάβουμε εξαιρέσεις όπου θα επιστρέφουν μια πρόβλεψη NaN και θα αγνοήσουμε όλες τις προειδοποιήσεις κατά τη διάρκεια της προσαρμογής και της αξιολόγησης.

Είμαστε πλέον έτοιμοι να αξιολογήσουμε μια αυτορυθμιζόμενη διαδικασία για κάθε μία από τις 39 σειρές σε κάθε ένα από τα 207 πακέτα εκπαίδευσης.

Θα ξεκινήσουμε δοκιμάζοντας μια AR(1) διαδικασία τρέχοντας το παραπάνω πρόγραμμα.

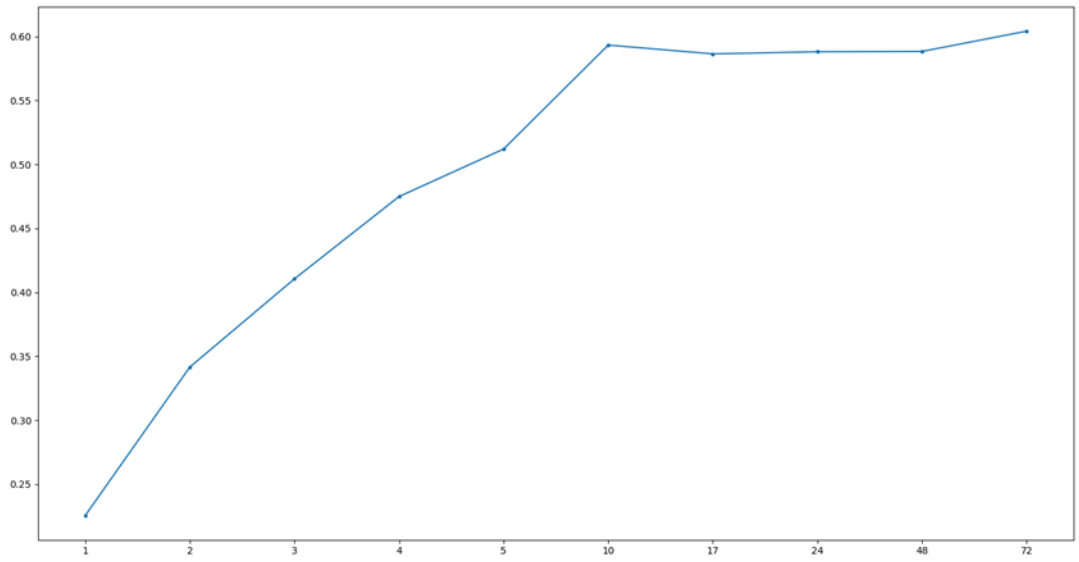
Η εκτέλεση του τυπώνει πρώτα το συνολικό MAE για το σετ δοκιμών, ακολουθούμενο από το MAE για κάθε χρόνο πρόβλεψης.

Πίνακας 15. Απόδοση ARIMA(1,0,0) μοντέλου

Mean MAE	+1	+2	+3	+4	+5	+10	+17	+24	+48	+72
0.492	0.225	0.342	0.410	0.475	0.512	0.593	0.586	0.588	0.588	0.604

Μπορούμε να δούμε ότι το μοντέλο επιτυγχάνει ένα MAE περίπου 0,492. Αυτό δείχνει ότι πράγματι η προσέγγιση έχει κάποια ικανότητα.

Επίσης τυπώνεται η γραφική παράσταση της MAE ανά ζητούμενο προβλεπόμενο χρόνο, που δείχνει τη γραμμική αύξηση του σφάλματος πρόβλεψης με την αύξηση του ζητούμενου χρόνου πρόβλεψης.



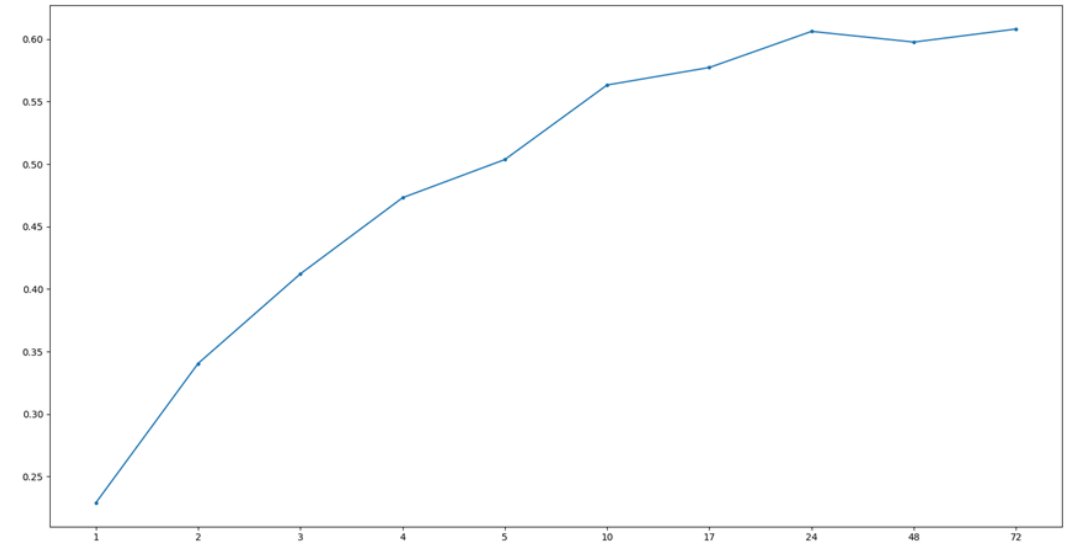
Εικόνα 87: MAE scores για AR(1) μοντέλο στους προβλεπόμενους χρόνους

Μπορούμε να αλλάξουμε τον κώδικα για να δοκιμάσουμε άλλα μοντέλα AR. Συγκεκριμένα, για το AR(2) τα αποτελέσματα δείχνουν μια περαιτέρω πτώση του σφάλματος σε μια συνολική MAE περίπου 0,490.

Πίνακας 16. Απόδοση ARIMA(2,0,0) μοντέλου

Mean MAE	**+1**	**+2**	**+3**	**+4**	**+5**	**+10**	**+17**	**+24**	**+48**	**+72**
0.490	0.229	0.340	0.412	0.473	0.504	0.563	0.577	0.606	0.598	0.608

Ακολουθεί το αντίστοιχο διάγραμμα των MAE αποτελεσμάτων για το AR(2).



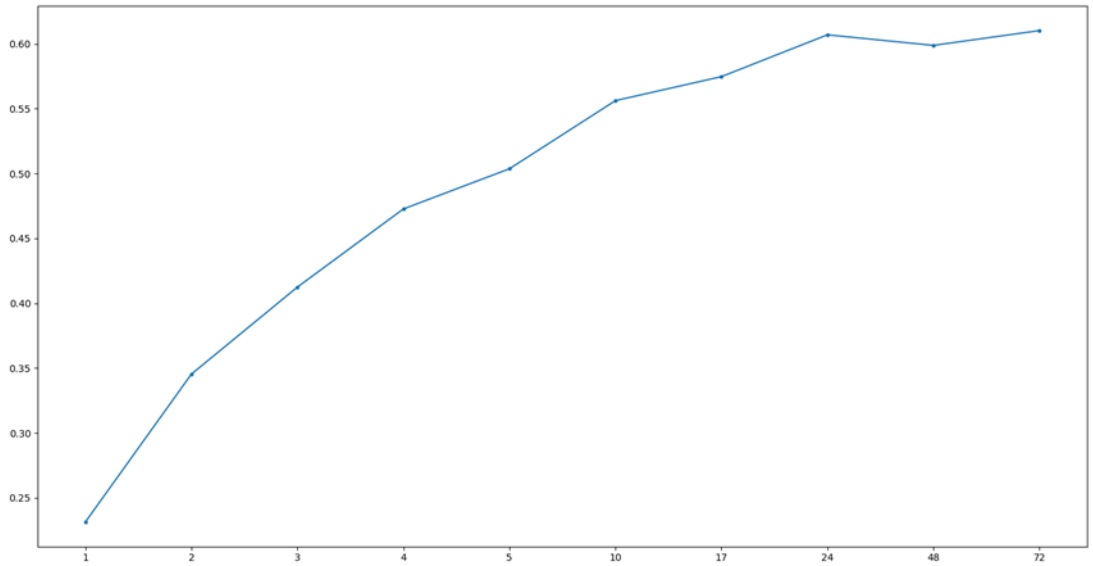
Εικόνα 88: MAE scores για AR(2) μοντέλο στους προβλεπόμενους χρόνους

Μπορούμε επίσης να δοκιμάσουμε ένα AR (3).

Πίνακας 17. Απόδοση ARIMA(3,0,0) μοντέλου

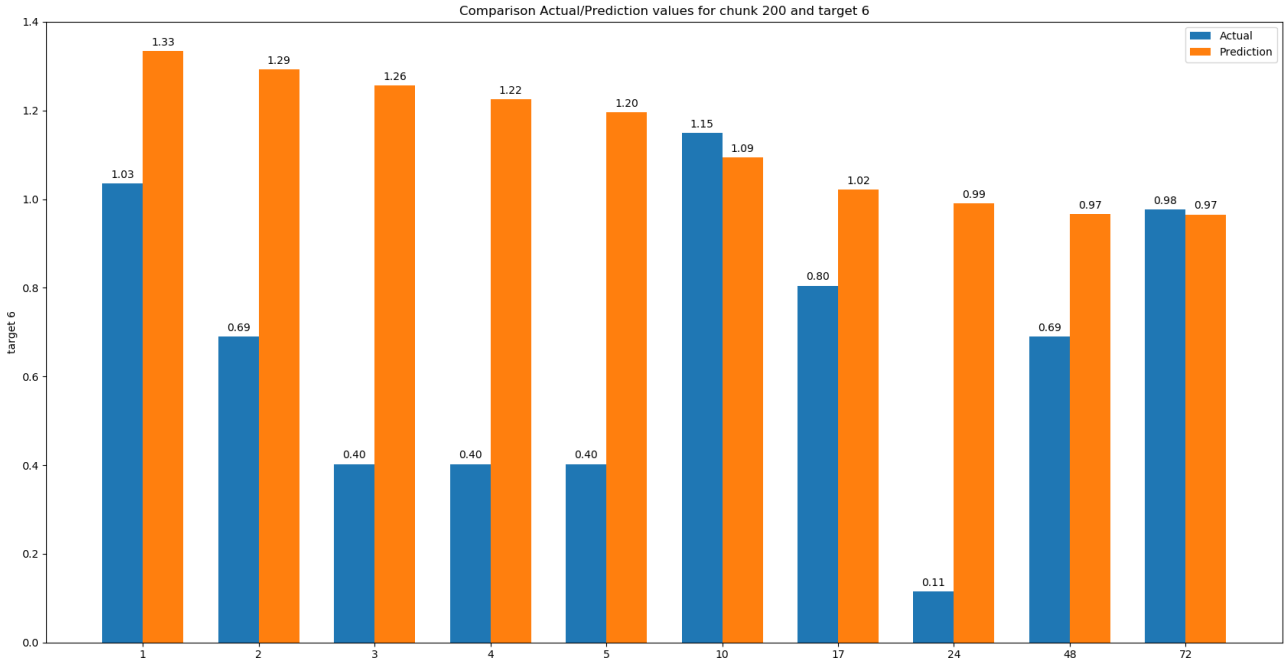
Mean MAE	+1	+2	+3	+4	+5	+10	+17	+24	+48	+72
0.491	0.231	0.345	0.412	0.473	0.504	0.556	0.575	0.607	0.599	0.610

Ακολουθεί το αντίστοιχο διάγραμμα των MAE αποτελεσμάτων για το AR(3).

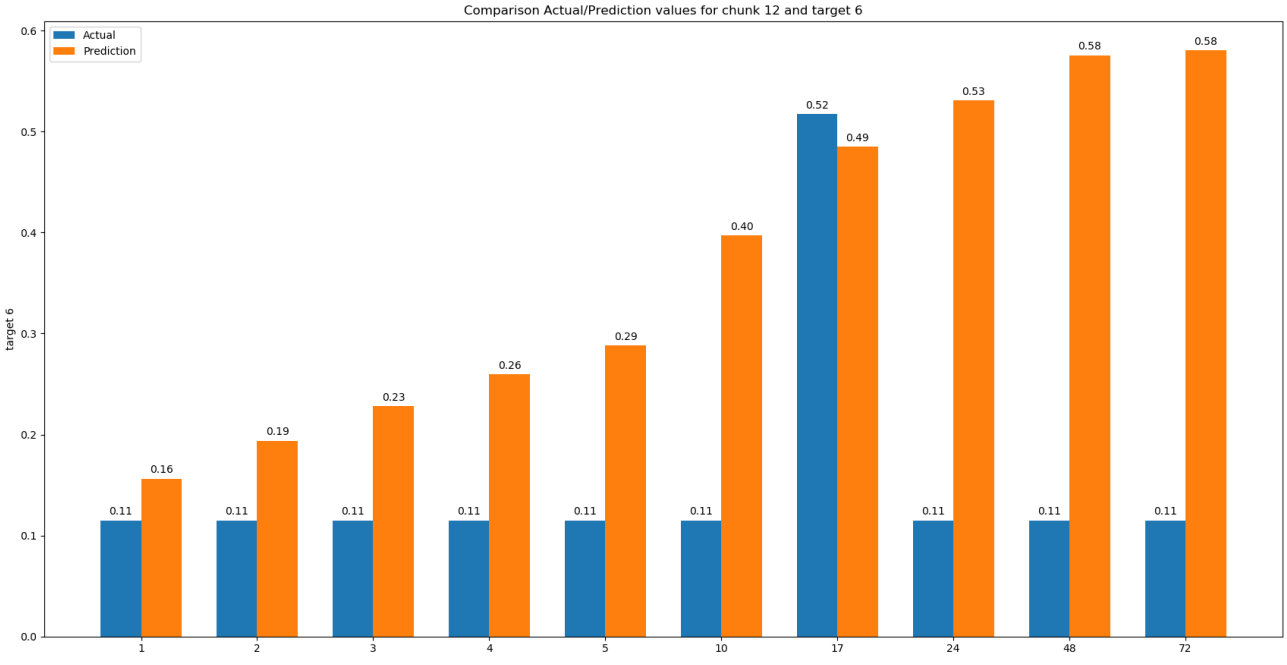


Εικόνα 89: MAE scores για AR(3) μοντέλο στους προβλεπόμενους χρόνους

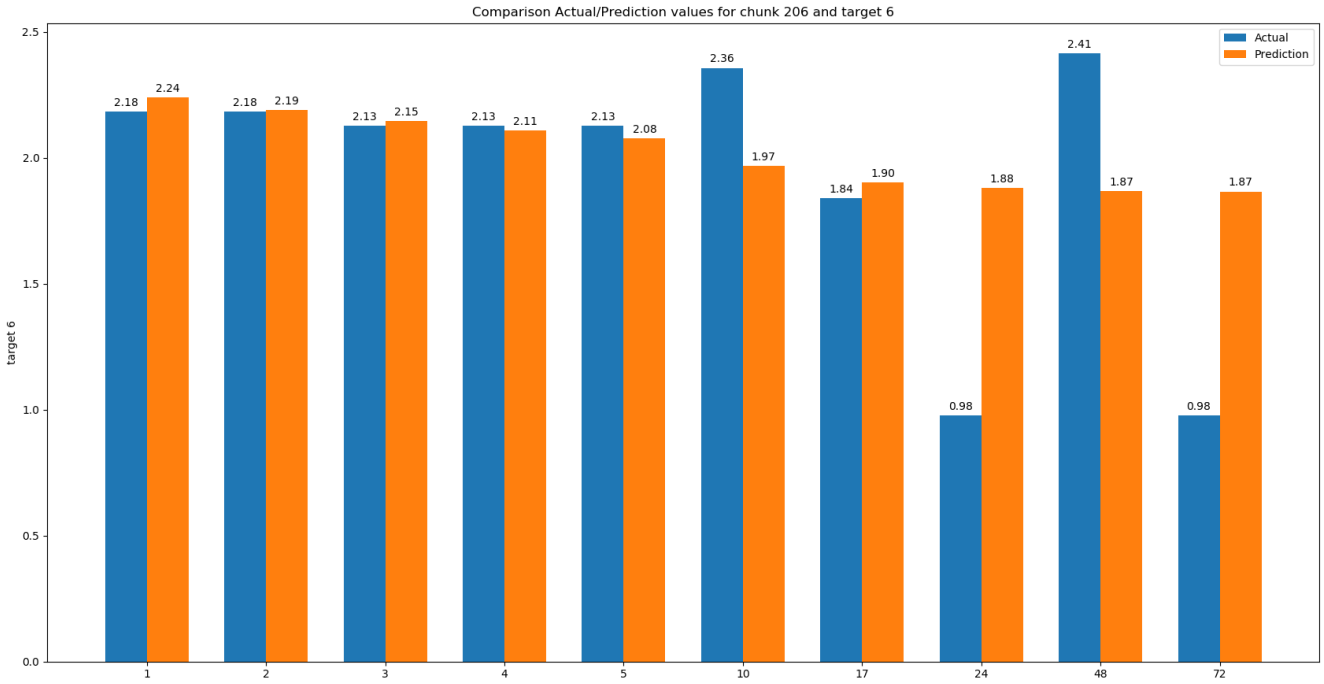
Η εκ νέου εκτέλεση του προγράμματος με την ενημέρωση δείχνει αύξηση της συνολικής MAE σε σύγκριση με ένα AR (2). Επομένως, ένα AR(2) μπορεί να είναι μια καλή επιλογή για γενική χρήση.



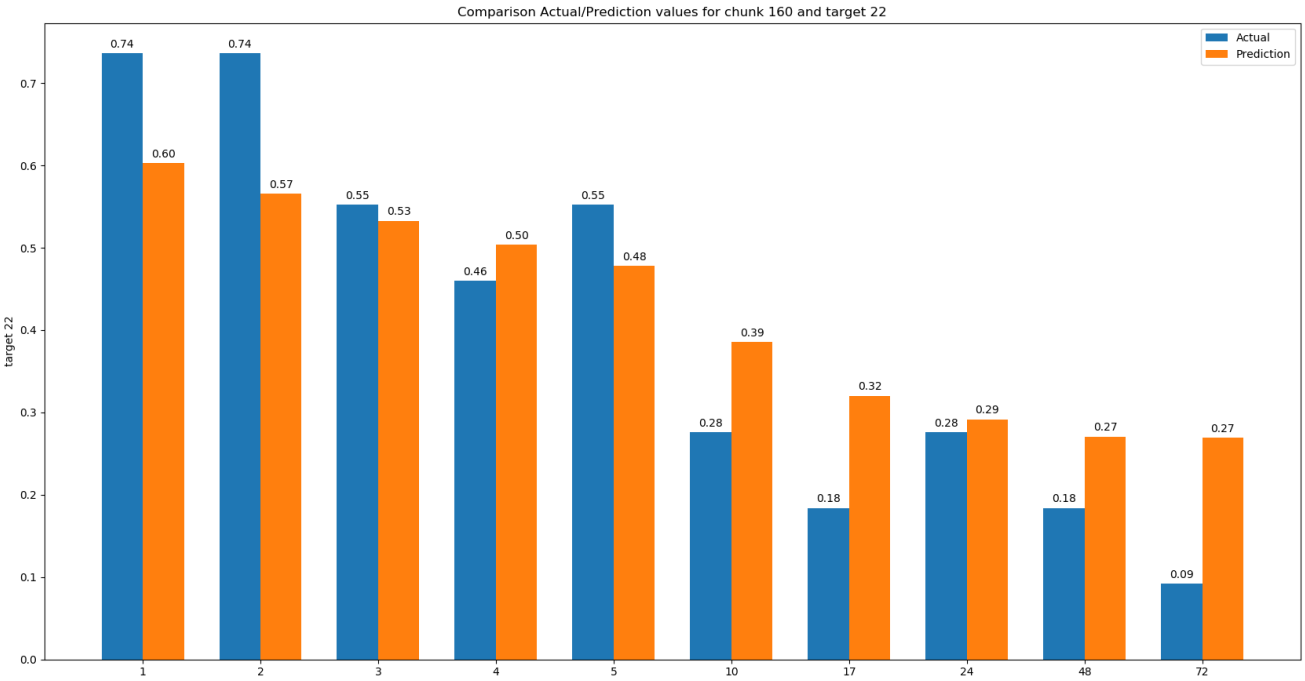
Εικόνα 90: Σύγκριση των πραγματικών και των προβλεπόμενων τιμών του βέλτιστου ARIMA μοντέλου για το chunk 200 και την 6η target μεταβλητή για τις επόμενες 72 ώρες



Εικόνα 91: Σύγκριση των πραγματικών και των προβλεπόμενων τιμών του βέλτιστου ARIMA μοντέλου για το chunk 12 και την 6η target μεταβλητή για τις επόμενες 72 ώρες



Εικόνα 92: Σύγκριση των πραγματικών και των προβλεπόμενων τιμών του βέλτιστου ARIMA μοντέλου για το chunk 206 και την 6η target μεταβλητή για τις επόμενες 72 ώρες



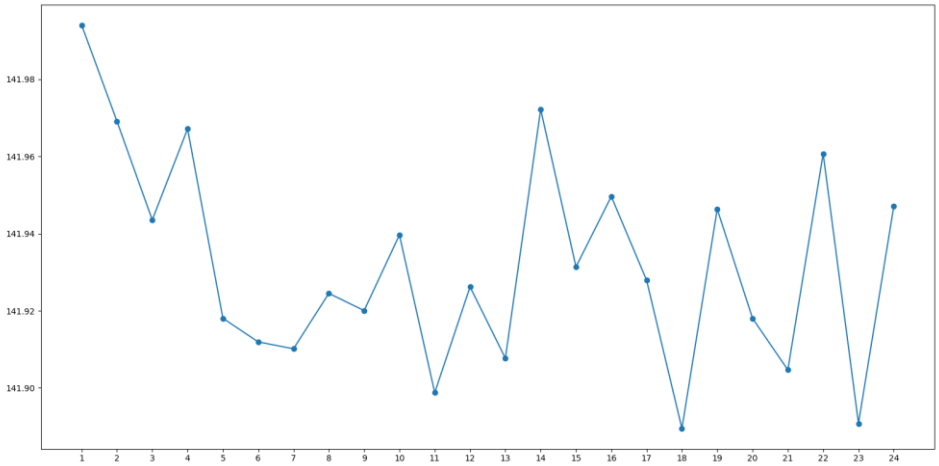
Εικόνα 93: Σύγκριση των πραγματικών και των προβλεπόμενων τιμών του βέλτιστου ARIMA μοντέλου για το chunk 160 και την 22η target μεταβλητή για τις επόμενες 72 ώρες

7.3.3 LSTMs

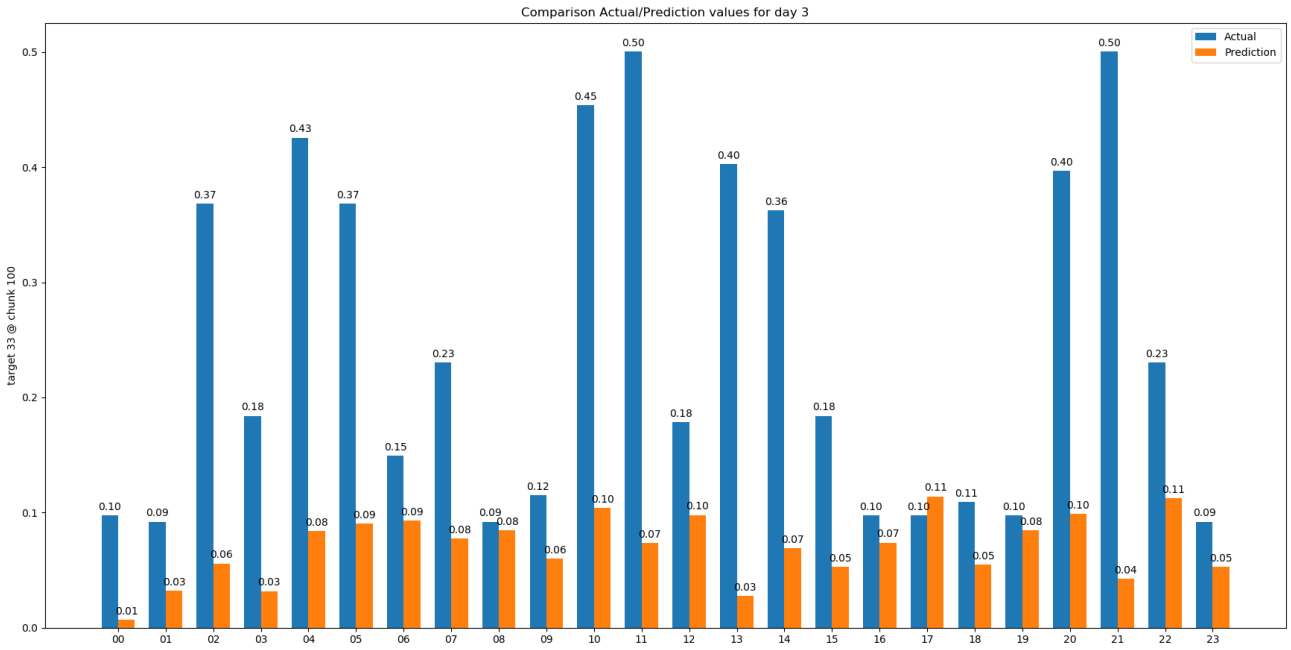
Τα πακέτα στα οποία είναι χωρισμένο το dataset δεν είναι συνεχή μεταξύ τους. Το πακέτο 1 αναφέρεται στο μήνα Οκτώβριο, το πακέτο 2 αναφέρεται στο μήνα Ιούλιο κοκ. Επομένως δεν μπορούμε να παραβλέψουμε το διαχωρισμό των πακέτων ώστε να λάβουμε όλο το dataset ως είσοδο στο LSTM μοντέλο μας. Και αυτό διότι το LSTM όπως έχουμε αναλύσει σε προηγούμενες ενότητες χρησιμοποιεί ως είσοδο δεδομένα από το προηγούμενο χρονικό βήμα πράγμα που σημαίνει ότι στις εναλλαγές μεταξύ των πακέτων δεν θα μπορούσαμε να έχουμε σωστές προβλέψεις και εισόδους στο μοντέλο μας.

Επομένως, αναπτύχθηκαν LSTM μοντέλα για κάθε πακέτο του dataset και διαμορφώθηκαν έτσι ώστε να γίνουν προβλέψεις τις επόμενες 24 ώρες. Ορίστηκαν epochs=70, batch size =16, δύο επίπεδα νευρώνων από 200 και 100 αντίστοιχα καθώς και άλλο ένα επίπεδο μέγεθος νευρώνων όσο ο αριθμός των εξόδων. Activation function='relu' και loss function='mse' ενώ και πάλι ο optimizer παραμένει ο 'adam'. Για τη δημιουργία του κατάλληλου shape σύγκρισης training και testing datasets, περιορίσαμε τα chunks ώστε να έχουν 152 τα δύο, 120 και 72 ωριαίες μετρήσεις αντίστοιχα και 42 target μεταβλητές.

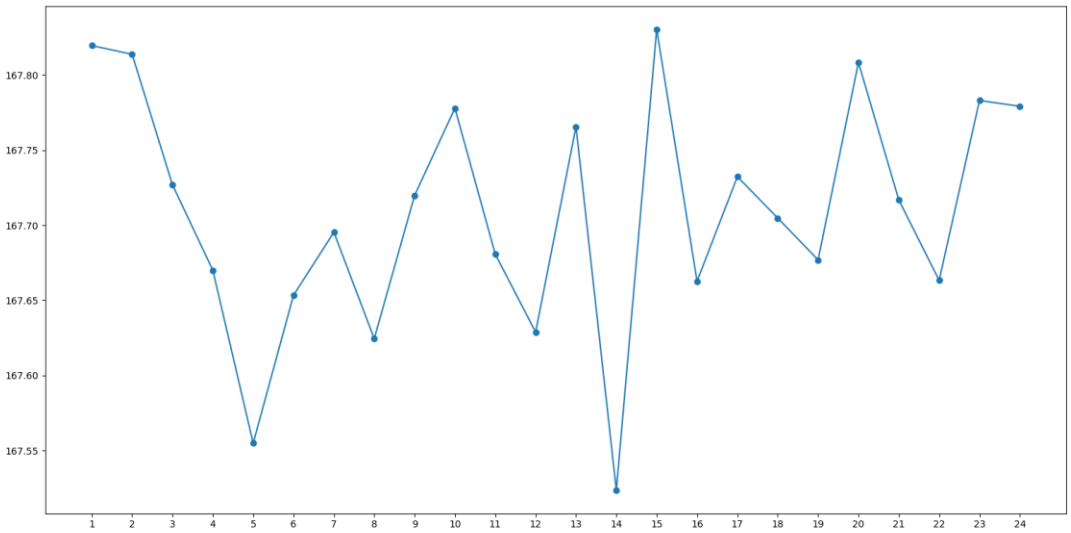
Στα περισσότερα target είχαμε τα σφάλματα να τείνουν στο μηδέν λόγω της ελάχιστης διακύμανσης τους, πιθανώς από τη διαδικασία συμπλήρωσης των κενών τιμών. Σε άλλες περιπτώσεις όπου διαθέταμε αποδεκτό αριθμό δεδομένων είχαμε αποτελέσματα ποικίλα, άλλα μεγαλύτερα άλλα μικρότερα και άλλα κοντά σε αυτά των ARIMA μοντέλων. Επιλεκτικά παρουσιάζουμε κάποιες από τις γραφικές αναπαραστάσεις των αποτελεσμάτων του μοντέλου μας. Βλέπουμε μια αναρχία στην απόδοση του LSTM.



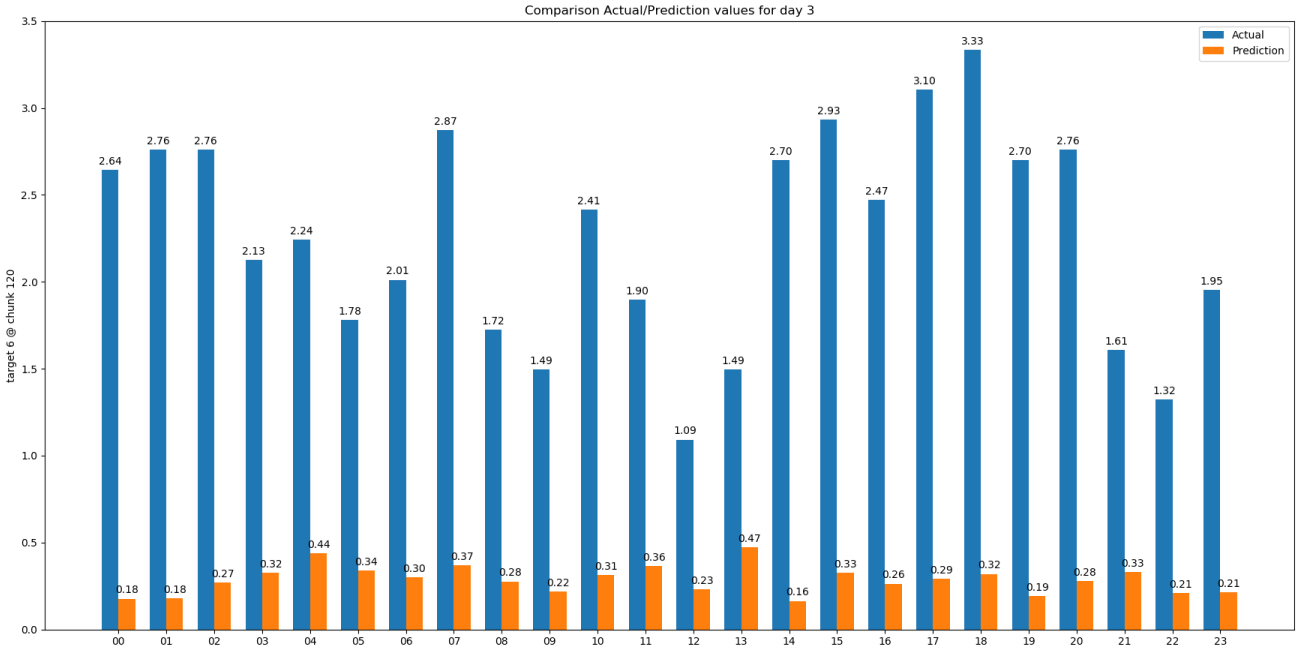
Εικόνα 94: Γραφική παράσταση των αποτελεσμάτων του LSTM για το chunk 100 και τη μεταβλητή 33 για την 3η ημέρα πρόβλεψης



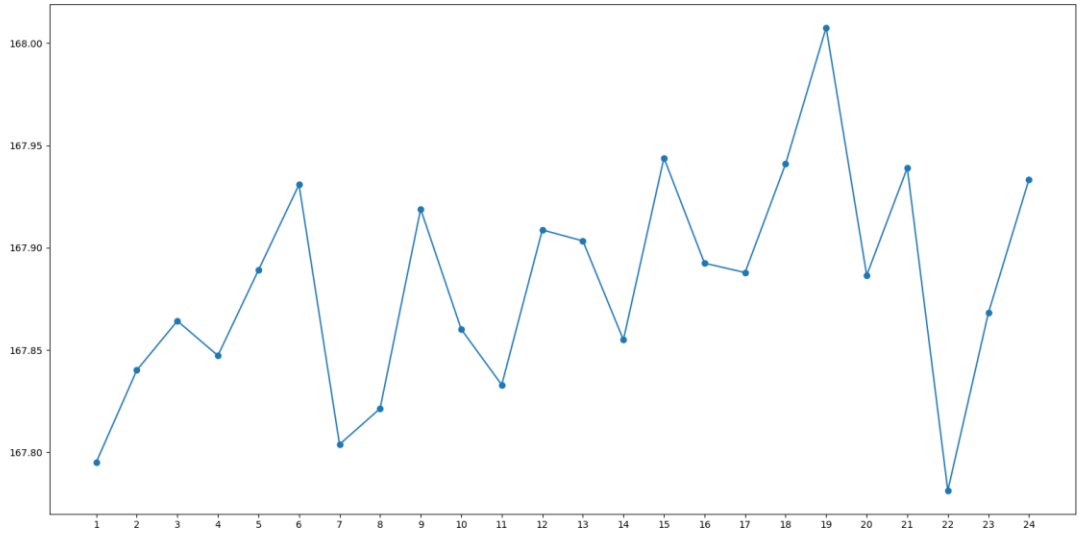
Εικόνα 95: Σύγκριση των πραγματικών και των προβλεπόμενων τιμών του LSTM μοντέλου για το chunk 100 και την 33η target μεταβλητή για την 3η ημέρα πρόβλεψης



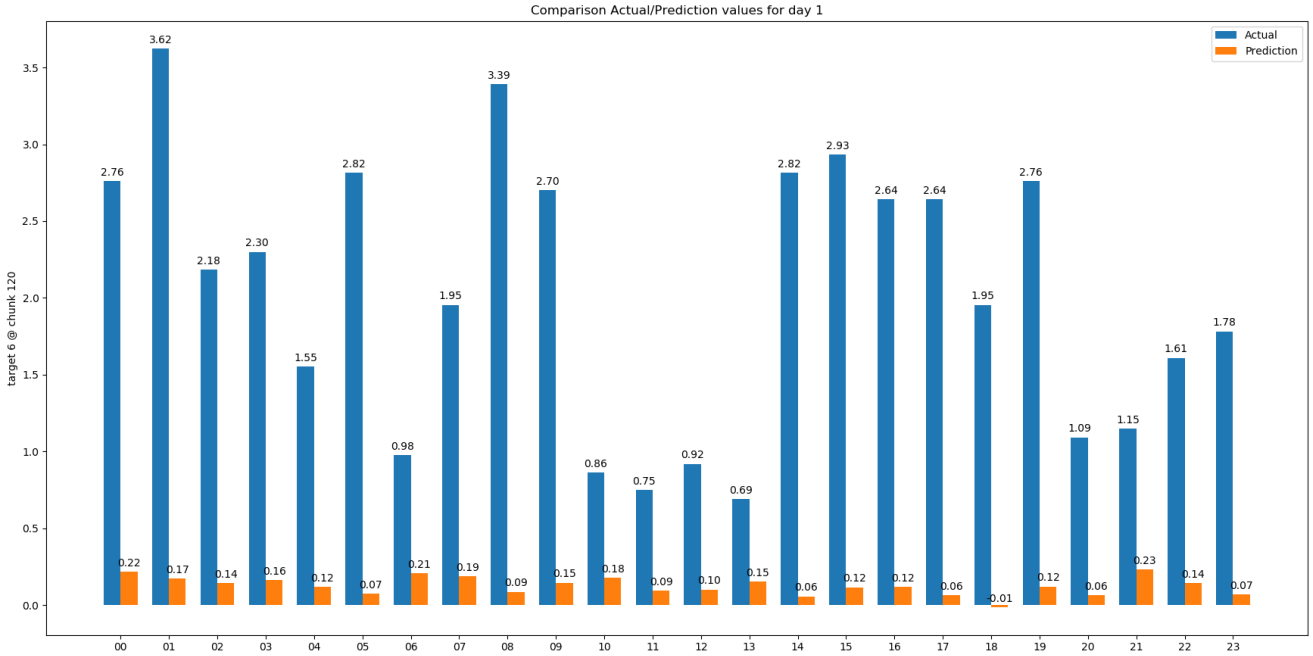
Εικόνα 96: Γραφική παράσταση των αποτελεσμάτων του LSTM για το chunk 120 και τη μεταβλητή 6 για την 3η ημέρα πρόβλεψης



Εικόνα 97: Σύγκριση των πραγματικών και των προβλεπόμενων τιμών του LSTM μοντέλου για το chunk 120 και την 6η target μεταβλητή για την 3η ημέρα πρόβλεψης

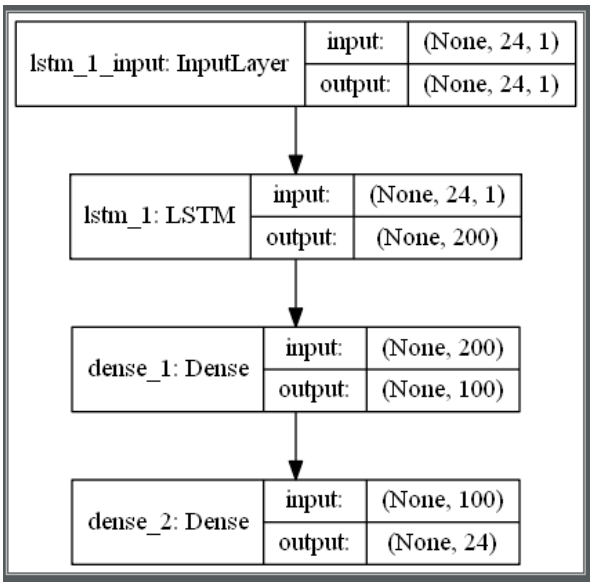


Εικόνα 98: Γραφική παράσταση των αποτελεσμάτων του LSTM για το chunk 120 και τη μεταβλητή 6 για την 1η ημέρα πρόβλεψης



Εικόνα 99: Σύγκριση των πραγματικών και των προβλεπόμενων τιμών του LSTM μοντέλου για το chunk 120 και την 6η target μεταβλητή για την 1η ημέρα πρόβλεψης

Τέλος, δημιουργούμε ένα σχήμα με την αποτύπωση της δομής του LSTM μοντέλου μας.



Εικόνα 100: Σχηματική αναπαράσταση του LSTM μοντέλου του προγράμματος (μέσω Keras)

8. ΣΥΜΠΕΡΑΣΜΑΤΑ

- Από την ανάπτυξη των μοντέλων ARIMA και LSTM στο πρώτο dataset εξετάστηκε η πρόβλεψη της ενεργειακής κατανάλωσης ενός νοικοκυριού για τις επόμενες επτά μέρες. Δοκιμάστηκαν διάφορες παραλλαγές στο LSTM μοντέλο με την απλή εκδοχή του να αποδίδει ελάχιστα καλύτερα.

Πίνακας 18. Συνολικές επιδόσεις των LSTM μοντέλων για το πρώτο dataset

LSTM	Mean RMSE
Univariate	369.620
Encoder-Decoder univariate	373.982
Encoder-Decoder multivariate	367.451
Encoder-Decoder CNN univariate	377.303
Encoder-Decoder Convolutional univariate	379.307

Σε ίδια κλίμακα κυμάνθηκε και το ARIMA μοντέλο ανάπτυξης αποδίδοντας ίδιες δυσκολίες πρόβλεψης με τις ίδιες κορυφές στην απόδοση του RMSE.

Πίνακας 19. Συνολική επίδοση του ARIMA μοντέλου για το πρώτο dataset

ARIMA	Mean RMSE
(6,0,0)	379.765

Το γενικό συμπέρασμα από τα παραπάνω μοντέλα είναι ότι το univariate LSTM μοντέλο αποδείχτηκε αποδοτικότερο για το συγκεκριμένο dataset. Ειδικότερα, όταν εξετάστηκε το dataset τροποποιημένο σε ώρες αντί για ημέρες, παρατηρούμε πολύ καλύτερες προβλέψεις σε σχέση με το ARIMA μοντέλο καθώς οι διαφορές πραγματικών και προβλεπόμενων τιμών είναι σημαντικά μικρότερες. Η Τρίτη και η Παρασκευή ήταν οι μέρες με την μεγαλύτερη ακρίβεια, ενώ το μεγαλύτερο σφάλμα στις προβλέψεις το είχαμε για την τελευταία μέρα, το Σάββατο. Ωστόσο, για καλύτερα αποτελέσματα απαιτούνται περισσότερα δεδομένα και η εξέτασή του αρχικού dataset σε λεπτά, που για υπολογιστικούς λόγους αποφεύχθηκε στη παρούσα εργασία.

- Από την ανάπτυξη των μοντέλων ARIMA και LSTM στο δεύτερο dataset εξετάστηκε η πρόβλεψη της συγκέντρωσης PM2.5 σωματιδίων τις επόμενες 24 ώρες.

Πίνακας 20. Τελικές επιδόσεις των ARIMA και LSTM μοντέλων για το δεύτερο dataset

Model	Mean RMSE
ARIMA (7,0,0)	72.207
Multivariate LSTM	26.229
Multivariate-multistep LSTM	26.250

Παρατηρούμε ότι έχουμε ένα εξαιρετικό χαμηλό RMSE στα αναπτυγμένα LSTM. Από τις γραφικές παραστάσεις τους παρατηρήσαμε ότι έχουμε απότομη πτώση στη από τη δεύτερη πρόβλεψη και μετά Εν αντιθέσει με το ARIMA μοντέλο, που παρόλο που έχει ένα σχετικά υψηλό σκορ συγκρίνοντάς το με το LSTM, από τη γραφική και από τα αριθμητικά αποτελέσματά του βλέπουμε μια αναμενόμενη συμπεριφορά, καθώς οι πρώτες προβλέψεις είναι πιο εύκολες να γίνουν και καθώς αυξάνεται ο χρόνος ανάλογα αυξάνεται και η αποτυχία του μοντέλου σε σωστή πρόβλεψη. Καταλήγουμε ότι το LSTM μοντέλο που αναπτύχθηκε έχει με διαφορά με καλύτερη επίδοση και προτείνεται για το δεύτερο dataset.

- Το τρίτο dataset επιζητούσαμε την πρόβλεψη 39 target μεταβλητών για συγκεκριμένες χρονικές στιγμές μέσα στις επόμενες τρεις ημέρες. Τα ARIMA μοντέλα είδαμε ότι απέδιδαν το ίδιο καλά. Από τη γραφική και από τα αριθμητικά αποτελέσματά του βλέπουμε μια αναμενόμενη συμπεριφορά, καθώς οι πρώτες προβλέψεις είναι πιο εύκολες να γίνουν και καθώς αυξάνεται ο χρόνος ανάλογα αυξάνεται και η αποτυχία του μοντέλου σε σωστή πρόβλεψη, όπως είχαμε δει και στο δεύτερο dataset.

Πίνακας 21. Τελική απόδοση του ARIMA μοντέλου για το τρίτο dataset

ARIMA	Total mean MAE
(2,0,0)	0.490

Ωστόσο, η δυσκολία αυτού του dataset ήταν ότι ήταν εξ αρχής χωρισμένο σε πακέτα, όπου καθένα από αυτά είχε περίπου 190 το μέγιστο τιμές ανά μεταβλητή ενώ το 40% των δεδομένων ήταν κενές τιμές. Ως αποτέλεσμα της ασυνέχειας των πακέτων, το LSTM αναπτύχθηκε για κάθε πακέτο χωριστά, με αποτέλεσμα να έχουμε ελάχιστε μετρήσεις για την ανάπτυξη κάποιου αποδοτικού LSTM μοντέλου πρόβλεψης. Στα περισσότερα target είχαμε τα σφάλματα να τείνουν στο μηδέν λόγω της ελάχιστης διακύμανσης τους, πιθανώς από τη διαδικασία συμπλήρωσης των κενών τιμών. Σε άλλες περιπτώσεις όπου διαθέταμε αποδεκτό αριθμό δεδομένων είχαμε αποτελέσματα ποικίλα, άλλα μεγαλύτερα άλλα μικρότερα και άλλα κοντά σε αυτά των ARIMA μοντέλων. Επομένως,

καταλήγουμε ότι το ARIMA μοντέλο είναι αρκετά αξιόπιστο και ότι η αρχιτεκτονική του LSTM είναι πιθανώς ακατάλληλη για το συγκεκριμένο dataset με τη δομή που δίνεται.

9. ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Rob J Hyndman, George Athanasopoulos. (2014). "Forecasting: principles and practice", ISBN: 9780987507105, OTexts

[2] John T. Mentzer Mark A. Moon (2004) "Sales Forecasting Management: A Demand Management Approach.", SAGE Publications

[3] Cleveland, Robert B., et al. "STL: A seasonal-trend decomposition procedure based on loess." Journal of Official Statistics 6.1 (1990): 3-73.

[5] Cortes, Corinna, and Vladimir Vapnik. "Support-vector networks." Machine learning 20.3 (1995): 273-297.

[6] "packtpub." [Online]. Available: https://www.packtpub.com/books/content/implementing-artificial-neural-networks-tensorflow

[7] "codeproject." [Online]. Available: https://www.codeproject.com/Articles/175777/Financial- predictor-via-neural-network

[8] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. MIT Press, 2016.

[9] R. Pascanu, C. Gulcehre, K. Cho, and Y. Bengio, "How to Construct Deep Recurrent Neural Networks," 2013. [Online]. Available: http://arxiv.org/abs/1312.6026

[10] A. Graves, A. r. Mohamed, and G. E. Hinton, "Speech recognition with deep recurrent neural networks," 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, no. 6, pp. 6645–6649, 2013.

[11] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Gated Feedback Recurrent Neural Networks," vol. 37, 2015. [Online]. Available: http://arxiv.org/abs/1502.02367

[12] Y. Bengio, Neural Networks for Speech and Sequence Recognition. International Thomson Com- puter Press, 1996. [Online]. Available: https://books.google.gr/books?id=mLVQAAAAMAAJ

[13] Y. Bengio, Yann LeCun, and H. Y, "Globally trained handwritten word recognizer using spatial representation, space displacement neural networks and hidden markov models," Advances in Neural Information Processing Systems, vol. 6, 1993.

[14] S. Aksoy and R. M. Haralick, "Feature normalization and likelihood-based similarity measures for image retrieval," Pattern Recognition Letters, vol. 22, no. 5, pp. 563–582, 2001.

[15] A. Ng, "Coursera." [Online]. Available: https://www.coursera.org/learn/machine-learning

[16] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," Neural Networks, vol. 2, no. 5, pp. 359–366, 1989.

[17] R. Maciejewsk, K. S. Candan, "Introduction to Time Series [Online]. Available: https://www.coursera.org/lecture/temporal-and-hierarchical-analysis/introduction-to-time-series-g6UjU

[18]Peter J. Brockwell, & Richard A. Davis, "Introduction to Time Series and Forecasting", 3rdedition, Springer International Publishing Switzerland, 2016

[19]C. Chatfield, "The analysis of time series: an introduction", 5fthedition, Chapman and Hall/CRC, 19963.George E. P. Box, & Gwilym M. Jenkins, & Gregory C. Reinsel, "Time Series Analysis Forecasting and Control", 3rdedition, Prentice-Hall, New Jersey, 1994

[20]Dr. N.D Lewis, "Deep TimeSeries Forecasting with Python: An Intuitive Introduction to Deep Learning for Applied Time Series Modeling", CreateSpace Independent Publishing Platform, 2016

[21]Avishek Pal, & PKS Prakash, "Practical Time Series Analysis: Master Time Series Data Processing, Visualization, and Modeling using Python", Packt Publishing, Birmingham, 2017

[22] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. MIT Press, 2016.

[23] Beijing PM2.5 Data Data Set, [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Beijing+PM2.5+Data#

[24] Individual household electric power consumption Data Set, [Online]. Available: https://archive.ics.uci.edu/ml/datasets/individual+household+electric+power+consumption

[25] EMC Data Science Global Hackathon (Air Quality Prediction), [Online]. Available: https://www.kaggle.com/c/dsg-hackathon/data

[26] https://datamarket.com/data/set/22ox/monthly-milk-production-pounds-per-cow-jan-62-dec-75#!ds=22ox&display=line

[27] https://www.datascienceblog.net/post/machine-learning/forecasting-an-introduction

[28] Phil Simon,Too Big to Ignore: The Business Case for Big Data, Wiley Publishing, 1stedition, 2013

[29] http://neuralnetworksanddeeplearning.com/chap1.html

[30] https://medium.com/shallow-thoughts-about-deep-learning/how-would-we-find-a-better-activation-function-than-relu-4409df217a5c

[31] https://medium.com/mlreview/understanding-lstm-and-its-diagrams-37e2f46f1714

[32] https://cedar.buffalo.edu/~srihari/CSE676/10.10%20LSTM.pdf

[33] https://www.cc.gatech.edu/~san37/post/dlhc-rnn/

[34] https://medium.com/ml-research-lab/under-fitting-over-fitting-and-its-solution-dc6191e34250

[35] https://cedar.buffalo.edu/~srihari/CSE676/5.2%20MLBasics-Capacity.pdf

[36] https://cedar.buffalo.edu/~srihari/CSE676/7.8%20EarlyStopping.pdf

[37] https://towardsdatascience.com/how-to-select-the-right-evaluation-metric-for-machine-learning-models-part-1-regrression-metrics-3606e25beae0

[38] https://keras.io/