ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

ΑΝΑΓΝΩΡΙΣΗ ΜΗ ΦΥΣΙΟΛΟΓΙΚΩΝ ΠΡΟΤΥΠΩΝ ΚΑΙ ΤΙΜΩΝ ΣΕ IoT ΣΥΣΤΗΜΑΤΑ ΓΙΑ ΑΝΙΧΝΕΥΣΗ ΒΛΑΒΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Χρύσα Ριζεάκου

Επιβλέπουσα: Βασιλική Καντερέ

Επίκουρη Καθηγήτρια Ε.Μ.Π.

Αθήνα, Ιούνιος 2023

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

ΑΝΑΓΝΩΡΙΣΗ ΜΗ ΦΥΣΙΟΛΟΓΙΚΩΝ ΠΡΟΤΥΠΩΝ ΚΑΙ ΤΙΜΩΝ ΣΕ IoT ΣΥΣΤΗΜΑΤΑ ΓΙΑ ΑΝΙΧΝΕΥΣΗ ΒΛΑΒΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Χρύσα Ριζεάκου

Επιβλέπουσα: Βασιλική Καντερέ

Επίκουρη Καθηγήτρια Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 4η Ιουλίου 2023.

.....	.....	.....
Β. Καντερέ	Δ.Τσουμάκος	Σ. Παπαβασιλείου
Επίκουρη Καθηγήτρια Ε.Μ.Π	Αναπληρωτής Καθηγητής Ε.Μ.Π	Καθηγητής Ε.Μ.Π

Αθήνα, Ιούνιος 2023

.....
Χρύσα Ριζεάκου
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Copyright © Χρύσα Ριζεάκου, 2023.
Με επιφύλαξη παντός δικαιώματος. All rights reserved.

Απαγορεύεται η αντιγραφή, αποθήκευση και διανομή της παρούσας εργασίας, εξ ολοκλήρου ή τμήματος αυτής, για εμπορικό σκοπό. Επιτρέπεται η ανατύπωση, αποθήκευση και διανομή για σκοπό μη κερδοσκοπικό, εκπαιδευτικής ή ερευνητικής φύσης, υπό την προϋπόθεση να αναφέρεται η πηγή προέλευσης και να διατηρείται το παρόν μήνυμα. Ερωτήματα που αφορούν τη χρήση της εργασίας για κερδοσκοπικό σκοπό πρέπει να απευθύνονται προς την συγγραφέα.

Οι απόψεις και τα συμπεράσματα που περιέχονται σε αυτό το έγγραφο εκφράζουν την συγγραφέα και δεν πρέπει να ερμηνευθεί ότι αντιπροσωπεύουν τις επίσημες θέσεις του Εθνικού Μετσόβιου Πολυτεχνείου.

Περίληψη

Το Διαδίκτυο των Πραγμάτων (Internet of Things - IoT) έχει αναδειχθεί ως ένα μετασχηματιστικό τεχνολογικό παράδειγμα, το οποίο συνδέει πολυάριθμες συσκευές και επιτρέπει την ανταλλαγή τεράστιων ποσοτήτων δεδομένων. Με την αυξανόμενη ανάπτυξη συστημάτων IoT σε διάφορους τομείς, η ανάγκη για αξιόπιστους μηχανισμούς ανίχνευσης σφαλμάτων καθίσταται υψίστης σημασίας. Η παρούσα διατριβή επικεντρώνεται στην ανάπτυξη τεχνικών για την αναγνώριση μη φυσιολογικών προτύπων και τιμών σε συστήματα IoT και πιο συγκεκριμένα σε βιομηχανικά συστήματα IoT, ώστε να διευκολύνεται η αποτελεσματική ανίχνευση σφαλμάτων.

Ο πρωταρχικός στόχος αυτής της εργασίας είναι να σχεδιάσει και να εφαρμόσει ένα ολοκληρωμένο πλαίσιο (framework) που μπορεί να εντοπίσει αποκλίσεις από την αναμενόμενη συμπεριφορά σε συστήματα IoT. Το προτεινόμενο πλαίσιο ενσωματώνει αλγορίθμους μηχανικής μάθησης, στατιστική ανάλυση και τεχνικές ανίχνευσης ανωμαλιών για την ακριβή διάκριση μεταξύ φυσιολογικών και μη φυσιολογικών μοτίβων ή τιμών. Το πλαίσιο στοχεύει στη βελτίωση της ακρίβειας και της αποτελεσματικότητας στην ανίχνευση σφαλμάτων, ελαχιστοποιώντας έτσι τον χρόνο διακοπής λειτουργίας του συστήματος και ενισχύοντας τη συνολική λειτουργική αξιοπιστία.

Για την επίτευξη αυτού του στόχου, πραγματοποιείται εκτενής ανάλυση διαφόρων αρχιτεκτονικών συστημάτων IoT, πρωτοκόλλων και μορφών δεδομένων για τον εντοπισμό πιθανών πηγών σφαλμάτων και των αντίστοιχων προτύπων τους. Διερευνάται ένα ευρύ φάσμα αλγορίθμων μηχανικής μάθησης, συμπεριλαμβανομένων τεχνικών με επίβλεψη, χωρίς επίβλεψη και με ημιεπίβλεψη, προκειμένου να προσδιοριστεί η καταλληλότητά τους για την αναγνώριση μη φυσιολογικών μοτίβων και τιμών σε ροές δεδομένων IoT. Επιπλέον, χρησιμοποιούνται μέθοδοι στατιστικής ανάλυσης για την καταγραφή των χρονικών εξαρτήσεων και των αλληλοσυσχετίσεων εντός των δεδομένων IoT.

Το προτεινόμενο πλαίσιο αξιολογείται με τη χρήση ενός πραγματικού συνόλου δεδομένων IoT που προέρχεται από ένα βιομηχανικό περιβάλλον, συγκεκριμένα από ένα εργοστάσιο κατασκευής πλαστικών. Χρησιμοποιούνται μετρικές επιδόσεων για την αξιολόγηση της αποτελεσματικότητας του πλαισίου όσον αφορά την ακριβή ανίχνευση μη φυσιολογικών προτύπων και τιμών, με παράλληλη ελαχιστοποίηση των ψευδώς θετικών και ψευδώς αρνητικών αποτελεσμάτων. Το πλαίσιο που προκύπτει προέρχεται από μια διαδικασία μαλακής/σκληρής ψηφοφορίας. Τέλος, σχεδιάζεται μια απλή αρχιτεκτονική IoT, η οποία αποτελείται από βασικά στοιχεία του IoT και μπορεί να ενσωματώσει το πλαίσιο και να αυτοματοποιήσει τη διαδικασία ανίχνευσης ανωμαλιών.

Εν κατακλείδι, η παρούσα διατριβή παρουσιάζει ένα ολοκληρωμένο πλαίσιο για την αναγνώριση μη φυσιολογικών προτύπων και τιμών σε συστήματα IoT, διευκολύνοντας την αποτελεσματική ανίχνευση σφαλμάτων και επίσης προτείνει μια απλή αρχιτεκτονική IoT στην οποία μπορεί να ενσωματωθεί το πλαίσιο. Τα προτεινόμενα αποτελέσματα συμβάλλουν στην προώθηση της αξιοπιστίας του IoT, επιτρέποντας την έγκαιρη παρέμβαση και τις ενέργειες συντήρησης για την ελαχιστοποίηση των διαταραχών του συστήματος, τη βελτίωση της λειτουργικής απόδοσης και την ενίσχυση της εμπειρίας των χρηστών στο ολοένα και πιο διασυνδεδεμένο τοπίο του IoT.

Λέξεις – κλειδιά: Διαδίκτυο των Πραγμάτων (Internet of Things - IoT), ανίχνευση σφαλμάτων, μη φυσιολογικά πρότυπα, βιομηχανικά συστήματα IoT, πλαίσιο (framework), μηχανική μάθηση, στατιστική ανάλυση, ανίχνευση ανωμαλιών, απλή αρχιτεκτονική IoT, λειτουργική αξιοπιστία

Abstract

The Internet of Things (IoT) has emerged as a transformative technological paradigm, connecting numerous devices and enabling the exchange of vast amounts of data. With the growing deployment of IoT systems in various domains, the need for reliable fault detection mechanisms becomes paramount. This thesis focuses on the development of techniques for recognizing abnormal patterns and values within IoT systems and more specifically Industrial IoT systems to facilitate efficient fault detection.

The primary objective of this paper is to design and implement a comprehensive framework that can identify deviations from expected behavior in IoT systems. The proposed framework incorporates machine learning algorithms, statistical analysis, and anomaly detection techniques to accurately distinguish between normal and abnormal patterns or values. The framework aims to improve the accuracy and efficiency of fault detection, thereby minimizing system downtime and enhancing overall operational reliability.

To achieve this goal, an extensive analysis of various IoT system architectures, protocols, and data formats is conducted to identify potential fault sources and their corresponding patterns. A wide range of machine learning algorithms, including supervised, unsupervised, and semi-supervised techniques, are explored to determine their suitability for recognizing abnormal patterns and values in IoT data streams. Moreover, statistical analysis methods are employed to capture temporal dependencies and interrelationships within IoT data.

The proposed framework is evaluated using a real-world IoT dataset obtained from an industrial setting, specifically a plastic manufacturing factory. Performance metrics are employed to assess the effectiveness of the framework in accurately detecting abnormal patterns and values, while minimizing false positives and false negatives. The resulting framework comes from a soft/hard voting process. At last, a simple IoT architecture is designed, which consists of basic IoT components and can incorporate the framework and automate the anomaly detection process.

In conclusion, this thesis presents a comprehensive framework for the recognition of abnormal patterns and values in IoT systems, facilitating effective fault detection and also suggest a simple IoT architecture in which the framework can be incorporated. The proposed results contribute to the advancement of IoT reliability, enabling timely intervention and maintenance actions to minimize system disruptions, improve operational efficiency, and enhance user experience in the increasingly interconnected IoT landscape.

Keywords: Internet of Things (IoT), fault detection mechanisms, abnormal patterns, Industrial IoT, framework, machine learning, statistical analysis, anomaly detection, simple IoT architecture, operational reliability

Ευχαριστίες

Αρχικά θα ήθελα να ευχαριστήσω την επιβλέπουσα καθηγήτρια Βασιλική Καντερέ για την εμπιστοσύνη που μου έδειξε αναθέτοντας μου τη συγκεκριμένη διπλωματική εργασία, η οποία μου έδωσε την ευκαιρία να έρθω σε επαφή με καινοτόμες τεχνολογίες και αρχιτεκτονικές σε έναν σημαντικό τομέα της σύγχρονης τεχνολογίας όπως είναι αυτός του Internet of Things.

Επιπλέον, θα ήθελα να ευχαριστήσω ιδιαίτερα τον υποψήφιο διδάκτορα του Ε.Μ.Π Πάρη Κερασιώτη για τη συνεχή καθοδήγηση και την ανεκτίμητη βοήθειά του. Είμαι ευγνώμων για την εξαιρετική συνεργασία που είχαμε και τον πολύτιμο χρόνο που αφιέρωσε καθόλη τη διάρκεια της παρούσας εργασίας.

Ένα μεγάλο ευχαριστώ στην οικογένεια μου για τη στήριξη και την υπομονή. Τέλος, ευχαριστώ τις φίλες μου, ειδικά τη Μαρίλη. Η κοινή φοιτητική μας πορεία ήταν για μένα πολύτιμη.

Χρύσα Ριζεάκου
Αθήνα, Ιούνιος 2023

Πίνακας Περιεχομένων

Ευρετήριο Εικόνων

Internet of Things

1.1 Εισαγωγή στο IoT

Το Διαδίκτυο των Πραγμάτων (IoT) αναφέρεται στο διασυνδεδεμένο δίκτυο φυσικών συσκευών, οχημάτων, οικιακών συσκευών και άλλων αντικειμένων με ηλεκτρονικά τμήματα, λογισμικό, αισθητήρες και συνδεσιμότητα που τους επιτρέπει να συνδέονται και να ανταλλάσσουν δεδομένα μεταξύ τους. Το IoT επιτρέπει σε αυτά τα αντικείμενα να ανιχνεύονται και να ελέγχονται εξ αποστάσεως μέσω της υπάρχουσας δικτυακής υποδομής, δημιουργώντας έτσι ευκαιρίες για πιο άμεση ενοποίηση μεταξύ του φυσικού κόσμου και των συστημάτων που βασίζονται σε υπολογιστές και οδηγώντας σε βελτιωμένη απόδοση, ακρίβεια και οικονομικό όφελος.

Με πιο απλά λόγια, το IoT αναφέρεται σε έναν κόσμο όπου τα καθημερινά αντικείμενα έχουν τη δυνατότητα να συλλέγουν και να μεταδίδουν δεδομένα μέσω του Διαδικτύου, καθιστώντας δυνατή την παρακολούθηση και τον έλεγχό τους εξ αποστάσεως. Αυτά τα αντικείμενα κυμαίνονται από απλές συσκευές, όπως "έξυπνους" θερμοστάτες και λαμπτήρες, έως πιο πολύπλοκα συστήματα, όπως βιομηχανικά μηχανήματα και ιατρικό εξοπλισμό. Ενσωματώνοντας αισθητήρες, ενεργοποιητές και δυνατότητες επικοινωνίας σε αυτές τις συσκευές, το IoT τους δίνει τη δυνατότητα να αλληλεπιδρούν μεταξύ τους και με εξωτερικά συστήματα, δημιουργώντας ένα δίκτυο συνδεδεμένων συσκευών.

Η τεχνολογία IoT είναι έτοιμη να μεταμορφώσει τον τρόπο που ζούμε, εργαζόμαστε και αλληλεπιδρούμε με τον κόσμο γύρω μας, προσφέροντας οφέλη όπως αυξημένη απόδοση, εξοικονόμηση κόστους και βελτιωμένη ασφάλεια και ευκολία. Ταυτόχρονα, εγείρει επίσης νέες προκλήσεις και ανησυχίες, όπως το απόρρητο και την ασφάλεια των δεδομένων, καθώς και την ανάγκη για πρότυπα και κανονισμούς που θα διέπουν την ανάπτυξη και τη χρήση των συστημάτων IoT.

Το Διαδίκτυο των Πραγμάτων (IoT) έχει ένα ευρύ φάσμα εφαρμογών σε διάφορους κλάδους και τομείς, όπως:

- Έξυπνα σπίτια: Η τεχνολογία IoT μπορεί να χρησιμοποιηθεί για τον έλεγχο και την αυτοματοποίηση διαφόρων συστημάτων σε ένα σπίτι, όπως ο φωτισμός, η θέρμανση και η ψύξη και η ασφάλεια του σπιτιού. Για παράδειγμα, ένας έξυπνος θερμοστάτης μπορεί να προγραμματιστεί ώστε να προσαρμόζει τη θερμοκρασία με βάση τα προγράμματα και τις προτιμήσεις των κατοίκων και ένα έξυπνο σύστημα ασφαλείας μπορεί να ειδοποιεί τους ιδιοκτήτες σπιτιού για οποιαδήποτε ασυνήθιστη δραστηριότητα.
- Υγειονομική περίθαλψη: Οι συσκευές IoT και οι φορητές τεχνολογίες μπορούν να βοηθήσουν στην παρακολούθηση της υγείας ενός ατόμου, στη συλλογή δεδομένων για τις ζωτικές του ενδείξεις και στην ειδοποίηση των παρόχων υγειονομικής περίθαλψης εάν υπάρχουν ανωμαλίες. Αυτό μπορεί να βελτιώσει την ποιότητα της περίθαλψης και να μειώσει τον κίνδυνο νοσηλείας.
- Γεωργία: Οι αισθητήρες IoT μπορούν να χρησιμοποιηθούν για την παρακολούθηση της υγρασίας του εδάφους, της θερμοκρασίας και των επιπέδων θρεπτικών ουσιών και παρέχουν στους αγρότες πληροφορίες σε πραγματικό χρόνο για την υγεία των καλλιεργειών τους. Αυτό μπορεί να βοηθήσει στη βελτιστοποίηση των αποδόσεων των καλλιεργειών και στη μείωση των απωλειών.
- Μεταφορές: Η τεχνολογία IoT μπορεί να χρησιμοποιηθεί για τη βελτίωση των συστημάτων μεταφορών, όπως η διαχείριση της κυκλοφορίας, η ασφάλεια των οχημάτων και η βελτιστοποίηση διαδρομής. Για παράδειγμα, τα συνδεδεμένα αυτοκίνητα μπορούν να επικοινωνούν μεταξύ τους και

με τις οδικές υποδομές για τη μείωση της κυκλοφοριακής συμφόρησης και τη βελτίωση της οδικής ασφάλειας.

- Κατασκευαστικός τομέας: Η τεχνολογία IoT μπορεί να χρησιμοποιηθεί για την παρακολούθηση και τον έλεγχο των διαδικασιών παραγωγής σε εργοστάσια, διασφαλίζοντας ότι χρησιμοποιείται η σωστή ποσότητα πρώτων υλών και μειώνοντας τα απόβλητα. Οι αισθητήρες IoT μπορούν επίσης να χρησιμοποιηθούν για την παρακολούθηση της απόδοσης των μηχανημάτων, επιτρέποντας τη συντήρηση πριν από την εμφάνιση βλαβών.
- Διαχείριση ενέργειας: Οι συσκευές IoT μπορούν να χρησιμοποιηθούν για την παρακολούθηση και τον έλεγχο της χρήσης ενέργειας σε σπίτια και κτίρια, μειώνοντας την κατανάλωση ενέργειας και το κόστος. Για παράδειγμα, οι έξυπνοι μετρητές μπορούν να παρακολουθούν τη χρήση ενέργειας σε πραγματικό χρόνο και τα έξυπνα συστήματα φωτισμού μπορούν να σβήσουν αυτόματα τα φώτα όταν ένα δωμάτιο δεν χρησιμοποιείται.

Αυτά είναι μόνο μερικά παραδείγματα από τις πολλές εφαρμογές της τεχνολογίας IoT. Με την ταχεία ανάπτυξη των συνδεδεμένων συσκευών και την αυξανόμενη διαθεσιμότητα της τεχνολογίας IoT, οι δυνατότητες για εφαρμογές IoT είναι ουσιαστικά ατελείωτες.

Για να γίνει πραγματικά αντιληπτή η έκταση του IoT ακολουθούν μερικοί τρόποι μέτρησης του μεγέθους και του εύρους του (Ranger, 2020):

- Αριθμός συνδεδεμένων συσκευών: Σύμφωνα με πρόσφατους υπολογισμούς, υπάρχουν επί του παρόντος πάνω από 27 δισεκατομμύρια συσκευές IoT σε χρήση παγκοσμίως και ο αριθμός αυτός προβλέπεται να αυξηθεί σε πάνω από 75 δισεκατομμύρια έως το 2025. Αυτό περιλαμβάνει ένα ευρύ φάσμα συσκευών, από έξυπνες οικιακές συσκευές και φορητές συσκευές μέχρι βιομηχανικά μηχανήματα και ιατρικό εξοπλισμό.
- Μέγεθος αγοράς: Η παγκόσμια αγορά για την τεχνολογία IoT εκτιμάται ότι αξίζει τρισεκατομμύρια δολάρια και αυξάνεται ραγδαία. Η αγορά περιλαμβάνει όχι μόνο τα στοιχεία υλικού και λογισμικού των συσκευών IoT, αλλά και τις υπηρεσίες και τις λύσεις που υποστηρίζουν την ανάπτυξη και χρήση συστημάτων IoT.
- Οικονομικός αντίκτυπος: Το IoT έχει σημαντικό αντίκτυπο σε διάφορες βιομηχανίες και οικονομίες, οδηγώντας την καινοτομία, βελτιώνοντας την αποτελεσματικότητα και δημιουργώντας νέες επιχειρηματικές ευκαιρίες. Για παράδειγμα, η τεχνολογία IoT συμβάλλει στη βελτιστοποίηση των αλυσίδων εφοδιασμού, στη μείωση των απορριμμάτων και στη βελτίωση της παραγωγικότητας στη μεταποιητική βιομηχανία, βελτιώνοντας παράλληλα την ενεργειακή απόδοση και μειώνοντας το κόστος στον ενεργειακό τομέα.
- Δεδομένα που δημιουργούνται: Οι συσκευές IoT παράγουν τεράστιο όγκο δεδομένων, με δισεκατομμύρια σημεία δεδομένων να συλλέγονται και να μεταδίδονται καθημερινά. Αυτά τα δεδομένα μπορούν να χρησιμοποιηθούν για διάφορους σκοπούς, όπως προγνωστική συντήρηση, διαχείριση κυκλοφορίας και εξατομικευμένη υγειονομική περίθαλψη.
- Κοινωνικός αντίκτυπος: Το IoT έχει επίσης σημαντικό αντίκτυπο στην κοινωνία, αλλάζοντας τον τρόπο που ζούμε, εργαζόμαστε και αλληλεπιδρούμε με τον κόσμο γύρω μας. Για παράδειγμα, η τεχνολογία IoT επιτρέπει νέες μορφές απομακρυσμένης εργασίας, διευκολύνοντας τους ανθρώπους να ελέγχουν και να παρακολουθούν τα σπίτια τους από απόσταση και βελτιώνοντας την πρόσβαση στην υγειονομική περίθαλψη σε απομακρυσμένες και υποεξυπηρετούμενες περιοχές.

Αυτοί είναι μερικοί μόνο τρόποι για να μετρηθεί το μέγεθος και ο αντίκτυπος του IoT. Η τεχνολογία βρίσκεται ακόμη στα πρώτα της στάδια ανάπτυξής της και είναι πιθανό να συνεχίσει να αναπτύσσεται και να εξελίσσεται τα επόμενα χρόνια, με νέες εφαρμογές και νέες ευκαιρίες για καινοτομία.

1.1.1 Βασικοί Μηχανισμοί IoT Συστημάτων

Το Διαδίκτυο των Πραγμάτων (IoT) λειτουργεί συνδέοντας φυσικές συσκευές και αντικείμενα στο διαδίκτυο, επιτρέποντάς τους να συλλέγουν και να ανταλλάσσουν δεδομένα και να ελέγχονται και να παρακολουθούνται εξ αποστάσεως. Ακολουθεί μια βασική επεξήγηση της λειτουργίας του IoT (Atzori, et al., 2010):

1. Συσκευές και αισθητήρες: Οι συσκευές IoT είναι συνήθως εξοπλισμένες με αισθητήρες, οι οποίοι συλλέγουν δεδομένα σχετικά με το περιβάλλον, την ίδια τη συσκευή ή άλλους παράγοντες. Για παράδειγμα, ένα έξυπνο θερμόμετρο μπορεί να συλλέγει δεδομένα για τη θερμοκρασία και την υγρασία, ενώ μία έξυπνη κλειδαριά μπορεί να συλλέγει δεδομένα για το αν είναι κλειδωμένη ή ξεκλείδωτη.
2. Συλλογή και μετάδοση δεδομένων: Τα δεδομένα που συλλέγονται από τους αισθητήρες μεταδίδονται σε μια κεντρική τοποθεσία, όπως έναν διακομιστή που βασίζεται στο υπολογιστικό νέφος (cloud) ή έναν τοπικό διανομέα (local hub), χρησιμοποιώντας ασύρματες τεχνολογίες όπως Wi-Fi, Bluetooth ή δίκτυα κινητής τηλεφωνίας. Στη συνέχεια, τα δεδομένα δέχονται επεξεργασία και αποθηκεύονται, έτοιμα για ανάλυση και δράση.
3. Επεξεργασία δεδομένων: Τα δεδομένα που συλλέγονται από συσκευές IoT υποβάλλονται σε επεξεργασία και αναλύονται προκειμένου να εξαχθούν σημαντικές πληροφορίες και να ληφθούν αποφάσεις. Αυτή η επεξεργασία μπορεί να πραγματοποιηθεί τοπικά, στην ίδια τη συσκευή ή στο cloud, όπου μπορεί να αναλυθεί χρησιμοποιώντας ισχυρότερους υπολογιστικούς πόρους.
4. Ενεργοποίηση: Με βάση τις πληροφορίες που προέρχονται από τα δεδομένα, οι συσκευές IoT μπορούν να προβούν σε ενέργειες, όπως να προσαρμόσουν τις ρυθμίσεις τους, να στείλουν ειδοποιήσεις ή να ενεργοποιήσουν άλλες συσκευές να αναλάβουν δράση. Για παράδειγμα, ένας έξυπνος θερμοστάτης μπορεί να προσαρμόσει τη θερμοκρασία με βάση τα δεδομένα που έχει συλλέξει ή μια έξυπνη κλειδαριά μπορεί να ξεκλειδώσει αυτόματα όταν πλησιάσει ένα άτομο με τα σωστά διαπιστευτήρια.
5. Διαλειτουργικότητα: Οι συσκευές IoT συχνά χρειάζεται να επικοινωνούν και να αλληλεπιδρούν μεταξύ τους, ακόμα κι αν έχουν κατασκευαστεί από διαφορετικούς κατασκευαστές. Αυτό επιτυγχάνεται με τη χρήση τυπικών πρωτοκόλλων, όπως τα MQTT, CoAP και HTTP, τα οποία επιτρέπουν στις συσκευές να ανταλλάσσουν δεδομένα και να αλληλεπιδρούν μεταξύ τους.
6. Ασφάλεια: Η ασφάλεια είναι μια κρίσιμη πτυχή του IoT, καθώς οι συνδεδεμένες συσκευές συχνά αποθηκεύουν και μεταδίδουν ευαίσθητα δεδομένα και μπορεί να είναι ευάλωτες σε επιθέσεις στον κυβερνοχώρο. Για να διασφαλίσουν την ασφάλεια των συστημάτων IoT, οι κατασκευαστές και οι προγραμματιστές χρησιμοποιούν μια σειρά μέτρων ασφαλείας, όπως κρυπτογράφηση, ασφαλής εκκίνηση και ασφαλείς ενημερώσεις λογισμικού, καθώς και τακτικές ενημερώσεις κώδικα λογισμικού και ελέγχους ασφαλείας.

Αυτοί είναι οι βασικοί μηχανισμοί του IoT και συνεργάζονται για να επιτρέπουν στις συσκευές να συλλέγουν και να ανταλλάσσουν δεδομένα, να ελέγχονται και να παρακολουθούνται εξ αποστάσεως και να αλληλεπιδρούν με άλλες συσκευές και συστήματα. Η ακριβής εφαρμογή αυτών των μηχανισμών μπορεί να ποικίλλει ανάλογα με τη συγκεκριμένη εφαρμογή IoT και τις συσκευές και την τεχνολογία που χρησιμοποιούνται, αλλά οι βασικές αρχές παραμένουν οι ίδιες.

1.1.2 Big Data και IoT

Τα μεγάλα δεδομένα αναφέρονται σε εξαιρετικά μεγάλα και πολύπλοκα σύνολα δεδομένων που είναι δύσκολο να τα επεξεργαστεί και να τα αναλύσει κανείς χρησιμοποιώντας παραδοσιακές τεχνικές και τεχνολογίες διαχείρισης δεδομένων. Αυτά τα σύνολα δεδομένων προέρχονται συχνά από διάφορες πηγές, συμπεριλαμβανομένων των μέσων κοινωνικής δικτύωσης, των δικτύων αισθητήρων, των οικονομικών συναλλαγών και άλλων.

Το Διαδίκτυο των Πραγμάτων (IoT) είναι μια βασική πηγή μεγάλων δεδομένων, καθώς οι συσκευές IoT παράγουν τεράστιες ποσότητες δεδομένων όπως έχουμε ήδη επισημάνει. Ο τεράστιος όγκος δεδομένων που παράγονται από συσκευές IoT είναι τόσο δύσκολα διαχειρίσιμος, ώστε καθίσταται αναγκαίο να εφαρμοστούν ειδικές τεχνικές ανάλυσης.

Οι τεχνολογίες big data λοιπόν, παρέχουν έναν τρόπο διαχείρισης των ποσοτήτων δεδομένων που παράγονται από τις συσκευές IoT, επιτρέποντας στους οργανισμούς να επεξεργάζονται, να αποθηκεύουν και να αναλύουν τα δεδομένα σε κλίμακα (at scale). Συνδυάζοντας μεγάλα δεδομένα και IoT, οι οργανισμοί μπορούν να μετατρέψουν τις τεράστιες ποσότητες δεδομένων που παράγονται από συσκευές IoT σε πολύτιμες γνώσεις και αποτελέσματα που μπορούν να αξιοποιηθούν. Στη συνέχεια θα δούμε αναλυτικά πώς ακριβώς συμβαίνει αυτό.

1.2 Αρχιτεκτονικές σε IoT

Το cloud computing και το edge computing είναι δύο ξεχωριστά υπολογιστικά μοντέλα που έχουν σχεδιαστεί για να υποστηρίζουν διαφορετικούς τύπους υπολογιστικών απαιτήσεων.

Το cloud computing (Erl, et al., 2013) αναφέρεται στην παροχή υπολογιστικών υπηρεσιών — συμπεριλαμβανομένων διακομιστών, αποθήκευσης, βάσεων δεδομένων, δικτύωσης, λογισμικού και analytics — μέσω του Διαδικτύου για ταχύτερη καινοτομία, ευέλικτους πόρους και οικονομίες κλίμακας. Το cloud computing παρέχει στους χρήστες πρόσβαση σε κοινόχρηστους υπολογιστικούς πόρους και επιτρέπει στους οργανισμούς να αποθηκεύουν, να επεξεργάζονται και να αναλύουν δεδομένα σε μια κεντρική τοποθεσία, συνήθως σε ένα κέντρο δεδομένων ή μια ομάδα κέντρων δεδομένων.

Το Edge computing (Hofstee, 2020), από την άλλη πλευρά, αναφέρεται σε ένα αποκεντρωμένο υπολογιστικό μοντέλο στο οποίο η επεξεργασία δεδομένων πραγματοποιείται στην άκρη του δικτύου, πιο κοντά στο σημείο όπου παράγονται τα δεδομένα, παρά σε μια κεντρική τοποθεσία. Το Edge computing έχει σχεδιαστεί για να υποστηρίζει την αυξανόμενη ζήτηση για εφαρμογές χαμηλής καθυστέρησης και υψηλού εύρους ζώνης. Με την επεξεργασία δεδομένων στο άκρο, ο υπολογισμός ακμών μειώνει την ποσότητα των δεδομένων που πρέπει να μεταδοθούν στο cloud ή σε ένα κεντρικό κέντρο δεδομένων, βελτιώνοντας την ανταπόκριση και την αποτελεσματικότητα του συστήματος.

Οι κύριες διαφορές μεταξύ του cloud computing και του edge computing είναι η τοποθεσία όπου γίνεται επεξεργασία και αποθήκευση δεδομένων και οι τύποι των εφαρμογών που υποστηρίζονται. Συνοπτικά, το cloud computing παρέχει κεντρικές υπολογιστικές υπηρεσίες, ενώ το edge computing παρέχει αποκεντρωμένες υπολογιστικές υπηρεσίες πιο κοντά στην άκρη του δικτύου.

1.2.1 Cloud computing

- Αρχιτεκτονική λ (Lambda Architecture)

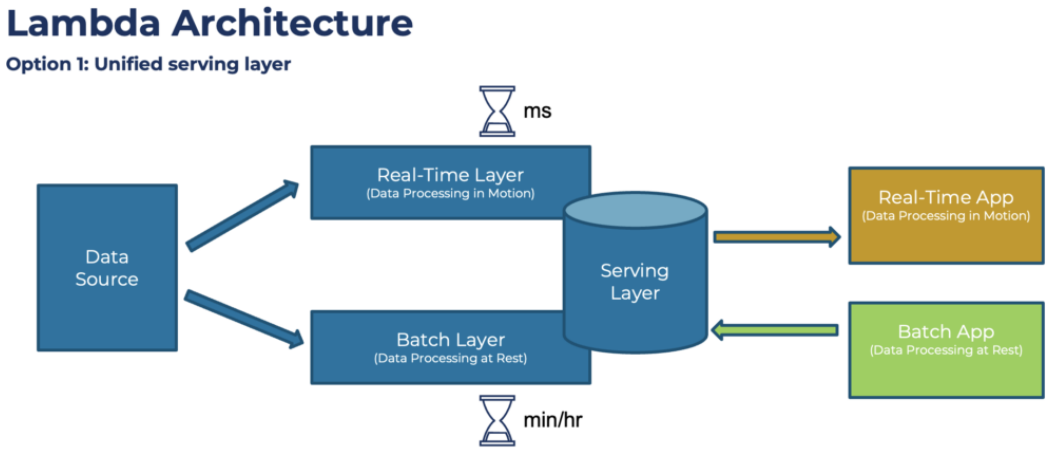
Η αρχιτεκτονική λ είναι μια αρχιτεκτονική επεξεργασίας δεδομένων που έχει σχεδιαστεί για να χειρίζεται μεγάλους όγκους δεδομένων σε πραγματικό χρόνο. Αναπτύχθηκε από τον Nathan Marz και συνδυάζει την επεξεργασία κατά παρτίδες και την επεξεργασία ροής σε ένα ενιαίο σύστημα. Η αρχιτεκτονική έχει σχεδιαστεί για να χειρίζεται τρία κύρια στοιχεία της επεξεργασίας δεδομένων: batch layer, speed layer και serving layer.

Batch Layer (Επίπεδο παρτίδας): Το επίπεδο αυτό είναι υπεύθυνο για το χειρισμό μεγάλων όγκων δεδομένων σε batch mode. Αποθηκεύει όλα τα δεδομένα σε ένα κατανεμημένο σύστημα αρχείων, όπως το Hadoop HDFS, και επεξεργάζεται δεδομένα χρησιμοποιώντας εργαλεία επεξεργασίας παρτίδας, όπως το Apache Spark ή το Apache Hadoop MapReduce. Το batch layer δημιουργεί ένα κύριο σύνολο δεδομένων, το οποίο είναι αμετάβλητο και περιέχει όλα τα ιστορικά δεδομένα.

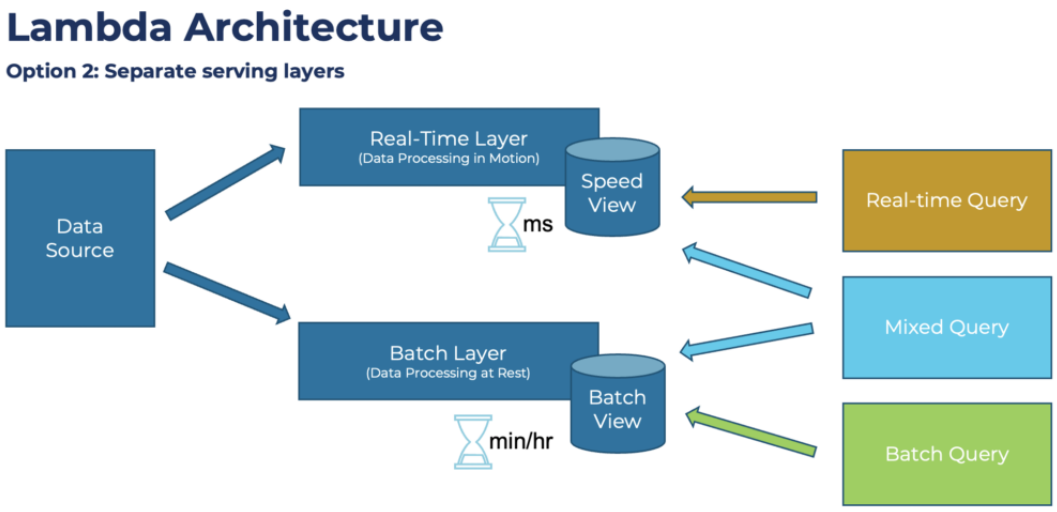
Speed Layer (Real-Time Layer) (Επίπεδο ταχύτητας): Το επίπεδο ταχύτητας είναι υπεύθυνο για την επεξεργασία ροών δεδομένων σε πραγματικό χρόνο. Χρησιμοποιεί ένα σύστημα επεξεργασίας ροής, όπως το Apache Kafka ή το Apache Storm, για την επεξεργασία δεδομένων σε πραγματικό χρόνο. Το επίπεδο ταχύτητας δημιουργεί μια προβολή των πιο πρόσφατων δεδομένων, η οποία είναι προσωρινή και ενημερώνεται συνεχώς.

Serving Layer (Επίπεδο εξυπηρέτησης): Το επίπεδο εξυπηρέτησης είναι υπεύθυνο για την παροχή γρήγορης πρόσβασης στα αποτελέσματα της επεξεργασίας κατά παρτίδες και ταχύτητας. Αποθηκεύει τα αποτελέσματα της επεξεργασίας κατά παρτίδες και ταχύτητας σε μια κατανεμημένη βάση δεδομένων, όπως το Apache Cassandra, και παρέχει πρόσβαση χωρίς καθυστέρηση (low-latency) στα αποτελέσματα μέσω ενός query API.

Η αρχιτεκτονική Lambda λειτουργεί συνδυάζοντας τα αποτελέσματα της επεξεργασίας κατά παρτίδες και ταχύτητας σε μια ενιαία, ενοποιημένη προβολή των δεδομένων. Έχει σχεδιαστεί για να χειρίζεται μεγάλους όγκους δεδομένων και να παρέχει γρήγορη, σε πραγματικό χρόνο πρόσβαση στα αποτελέσματα της επεξεργασίας.



Εικόνα 1-1 Αρχιτεκτονική λ με ενοποιημένο επίπεδο εξυπηρέτησης



Εικόνα 1-2 Αρχιτεκτονική λ με διαχωρισμένο επίπεδο εξυπηρέτησης

Οι εφαρμογές της αρχιτεκτονικής Lambda περιλαμβάνουν την επεξεργασία δεδομένων σε πραγματικό χρόνο, όπως στην ανάλυση των μέσων κοινωνικής δικτύωσης, στην επεξεργασία δεδομένων αισθητήρων και στην ανάλυση οικονομικών δεδομένων. Είναι ιδιαίτερα χρήσιμη σε εφαρμογές στις οποίες η πρόσβαση με χαμηλή καθυστέρηση σε δεδομένα είναι κρίσιμη.

Τα κύρια πλεονεκτήματα της αρχιτεκτονικής Lambda είναι η ικανότητά της να χειρίζεται μεγάλους όγκους δεδομένων, ο σχεδιασμός της με ανοχή σε σφάλματα και η ικανότητά της να χειρίζεται την επεξεργασία δεδομένων σε πραγματικό χρόνο. Η αρχιτεκτονική είναι επίσης εξαιρετικά επεκτάσιμη (highly scalable) και μπορεί να προσαρμοστεί για να χειρίζεται τις μεταβαλλόμενες απαιτήσεις δεδομένων.

Ωστόσο, η αρχιτεκτονική λ έχει ορισμένους περιορισμούς. Απαιτεί εξειδικευμένες δεξιότητες και τεχνογνωσία για να σχεδιαστεί και να εφαρμοστεί και μπορεί να είναι περίπλοκο και δύσκολο να συντηρηθεί μιας και οι τεχνολογίες που απαιτούνται για να τρέξουν τα 3 επίπεδα είναι σύνθετες. Μπορεί επίσης να οδηγήσει σε επικάλυψη δεδομένων, η οποία κατ' επέκταση οδηγεί σε ασυνέπειες στα δεδομένα.

- Αρχιτεκτονική κ (Kappa Architecture)

  Η αρχιτεκτονική Kappa είναι μια αρχιτεκτονική επεξεργασίας δεδομένων που έχει σχεδιαστεί για να χειρίζεται μεγάλους όγκους δεδομένων με τρόπο ανεκτικό σε σφάλματα, επεκτάσιμο και σε πραγματικό χρόνο. Εισήχθη από τον Jay Kreps το 2014 ως εξέλιξη της αρχιτεκτονικής Lambda. Η αρχιτεκτονική Kappa έχει σχεδιαστεί για να απλοποιεί την πολυπλοκότητα της αρχιτεκτονικής Lambda εξαλείφοντας το batch layer.

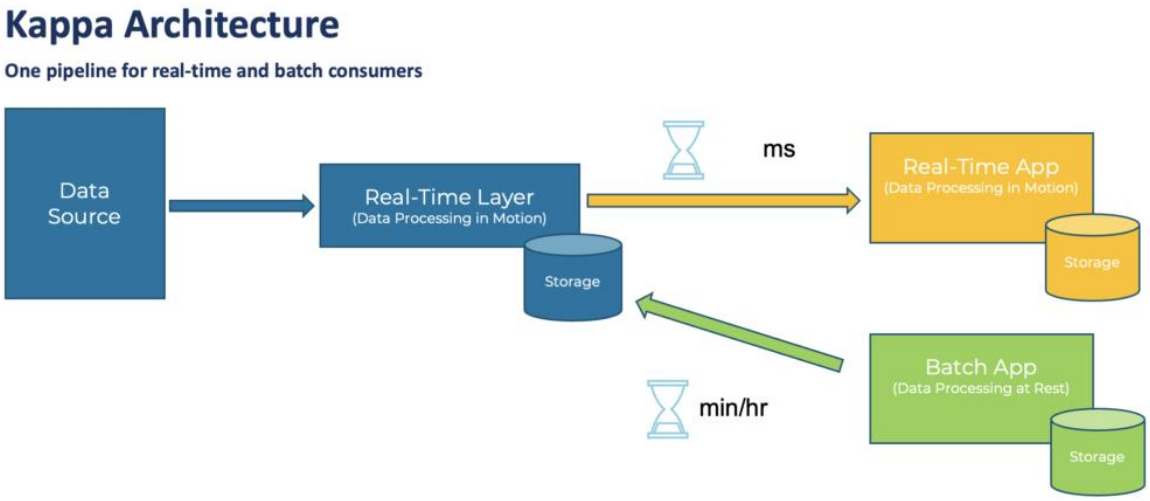
  Αποτελείται από δύο βασικά στοιχεία:

  Stream Processing Layer (Επίπεδο επεξεργασίας ροής): Το επίπεδο επεξεργασίας ροής είναι υπεύθυνο για την επεξεργασία ροών δεδομένων σε πραγματικό χρόνο χρησιμοποιώντας ένα

σύστημα επεξεργασίας ροής, όπως το Apache Kafka Streams, το Apache Flink ή το Apache Storm. Το επίπεδο επεξεργασίας ροής καταναλώνει δεδομένα από τη ροή εισόδου, τα επεξεργάζεται σε πραγματικό χρόνο και εξάγει τα επεξεργασμένα δεδομένα σε μια νέα ροή.

Serving Layer (Επίπεδο εξυπηρέτησης): Το επίπεδο εξυπηρέτησης είναι υπεύθυνο για την εξυπηρέτηση των ερωτημάτων (queries) και την παροχή πρόσβασης στα επεξεργασμένα δεδομένα. Αποθηκεύει τα επεξεργασμένα δεδομένα σε μια κατανεμημένη βάση δεδομένων, όπως το Apache Cassandra ή το Apache HBase, και παρέχει ένα query API που επιτρέπει στους χρήστες να έχουν πρόσβαση στα δεδομένα.

Στην αρχιτεκτονική Kappa η επεξεργασία δεδομένων γίνεται σε πραγματικό χρόνο καθώς φτάνουν στο επίπεδο επεξεργασίας ροής. Τα επεξεργασμένα δεδομένα γίνονται στη συνέχεια άμεσα διαθέσιμα για αναζήτηση στο επίπεδο εξυπηρέτησης. Η αρχιτεκτονική Kappa εξαλείφει το batch layer που βρίσκεται στην αρχιτεκτονική Lambda, απλοποιώντας το συνολικό σύστημα και μειώνωντας την καθυστέρηση.



Εικόνα 1-3 Αρχιτεκτονική κ

Οι εφαρμογές της αρχιτεκτονικής Kappa περιλαμβάνουν επεξεργασία δεδομένων σε πραγματικό χρόνο, όπως σε εφαρμογές IoT, ανίχνευση απάτης σε πραγματικό χρόνο και αναλύσεις μέσων κοινωνικής δικτύωσης. Είναι ιδιαίτερα χρήσιμη αρχιτεκτονική σε εφαρμογές στις οποίες η γρήγορη πρόσβαση σε δεδομένα είναι κρίσιμη.

Τα κύρια πλεονεκτήματα της αρχιτεκτονικής Kappa είναι η απλότητα, η επεκτασιμότητα (scalability) και οι δυνατότητες επεξεργασίας δεδομένων σε πραγματικό χρόνο. Η αρχιτεκτονική είναι επίσης ανεκτική σε σφάλματα (fault-tolerant), καθώς μπορεί να χειριστεί τις αστοχίες και να ανακάμψει αυτόματα από αυτές μέσω της απαίτησης επανεπεξεργασίας δεδομένων. Είναι απλούστερη στην υλοποίηση σε σχέση με τη λ και ως εκ τούτου η διαδικασία ανάπτυξης, debugging και συντήρησης είναι πιο απλή.

Ωστόσο, η αρχιτεκτονική Kappa έχει ορισμένους περιορισμούς. Ενδέχεται να μην είναι κατάλληλη για εφαρμογές που απαιτούν πολύπλοκη μαζική επεξεργασία, όπως η μηχανική εκμάθηση ή η εξόρυξη δεδομένων λόγω της απουσίας του batch layer. Επιπλέον, η αρχιτεκτονική Kappa μπορεί να

απαιτεί εξειδικευμένες δεξιότητες για το σχεδιασμό και την εφαρμογή, ειδικά όταν πρόκειται για μεγάλους όγκους δεδομένων.

1.2.2 Edge Computing

- Cloudlets

Το Cloudlet είναι ένας όρος που χρησιμοποιείται για να περιγράψει ένα κέντρο δεδομένων μικρής κλίμακας που παρέχει υπηρεσίες υπολογιστικού νέφους σε κοντινές κινητές συσκευές. Λειτουργεί ως γέφυρα μεταξύ των φορητών συσκευών και του cloud, παρέχοντας υπηρεσίες επεξεργασίας δεδομένων χαμηλής καθυστέρησης, ενεργειακής απόδοσης και σε πραγματικό χρόνο. Η αρχιτεκτονική του cloud αποτελείται από ένα σύμπλεγμα διακομιστών που είναι γεωγραφικά κατανεμημένοι για να είναι πιο κοντά στους τελικούς χρήστες, διασυνδέονται με συνδέσεις δικτύου υψηλής ταχύτητας και διαχειρίζονται από ένα στρώμα ενδιάμεσου λογισμικού cloud. Αυτή η αρχιτεκτονική έχει τέσσερα επίπεδα: υλικό (hardware), εικονικοποίηση (virtualization), ενδιάμεσο λογισμικό (middleware) και εφαρμογή (application) .

Το hardware layer αποτελείται από φυσικούς διακομιστές, συσκευές αποθήκευσης και εξοπλισμό δικτύωσης. Το virtualization layer παρέχει δυνατότητες εικονικοποίησης που επιτρέπουν σε πολλαπλές εικονικές μηχανές να εκτελούνται στον ίδιο φυσικό διακομιστή. Το επίπεδο ενδιάμεσου λογισμικού παρέχει ένα σύνολο υπηρεσιών και API που επιτρέπουν στις κινητές συσκευές να εκφορτώνουν εργασίες υπολογισμού στο cloudlet. Τέλος, το application layer αποτελείται από εφαρμογές που αναπτύσσονται στο cloudlet.

Η αρχιτεκτονική cloudlet έχει πολλές εφαρμογές, συμπεριλαμβανομένων της επαυξημένης πραγματικότητας για κινητά, του παιχνιδιού για κινητά και των υπηρεσιών που βασίζονται στην τοποθεσία. Στην επαυξημένη πραγματικότητα για κινητά, τα cloudlet παρέχουν επεξεργασία δεδομένων αισθητήρων σε πραγματικό χρόνο και τρισδιάστατη απόδοση, κάτι που είναι κρίσιμο για μια εμπειρία χρήστη υψηλής ποιότητας. Στα παιχνίδια για κινητά, τα cloudlets μπορούν να παρέχουν μείωση της καθυστέρησης στη ροή παιχνιδιών και επεξεργασία δεδομένων παιχνιδιού σε πραγματικό χρόνο. Στις υπηρεσίες που βασίζονται σε τοποθεσία, τα cloudlets μπορούν να παρέχουν επεξεργασία δεδομένων αισθητήρων και πληροφοριών τοποθεσίας σε πραγματικό χρόνο.

Τα πλεονεκτήματα της αρχιτεκτονικής cloudlet περιλαμβάνουν μειωμένη καθυστέρηση, ενεργειακή απόδοση, επεκτασιμότητα και βελτιωμένη εμπειρία χρήστη. Τα Cloudlet παρέχουν υπηρεσίες χαμηλής καθυστέρησης λόγω της εγγύτητάς τους με τους τελικούς χρήστες. Με τη μεταφόρτωση εργασιών υπολογισμού στο cloudlet, οι κινητές συσκευές μπορούν να εξοικονομήσουν ενέργεια. Τα Cloudlet μπορούν εύκολα να κλιμακωθούν προς τα πάνω ή προς τα κάτω με βάση τη ζήτηση για τις υπηρεσίες τους.

Ωστόσο, υπάρχουν και ορισμένα μειονεκτήματα της αρχιτεκτονικής cloudlet. Η εγκατάσταση και η συντήρηση μιας υποδομής cloudlet μπορεί να είναι δαπανηρή. Τα δεδομένα που μεταδίδονται μεταξύ κινητών συσκευών και cloudlet είναι ευάλωτα σε απειλές ασφαλείας. Τέλος, η αρχιτεκτονική αυτή είναι πολύπλοκη και απαιτεί εξειδικευμένες δεξιότητες για τη διαχείριση και τη συντήρηση.

Συμπερασματικά, η αρχιτεκτονική cloudlet είναι μια πολλά υποσχόμενη προσέγγιση για την παροχή υπηρεσιών υπολογιστικού νέφους σε κοντινές κινητές συσκευές. Οι εφαρμογές της είναι ποικίλες όπως και τα πλεονεκτήματα της, λόγου χάρη μειωμένη καθυστέρηση, ενεργειακή απόδοση,

επεκτασιμότητα και βελτιωμένη εμπειρία χρήστη. Ωστόσο, έχει επίσης ορισμένα μειονεκτήματα, συμπεριλαμβανομένου του κόστους, των ανησυχιών για την ασφάλεια των δεδομένων των χρηστών και της πολυπλοκότητας.

- Fog computing

  Το Fog Computing είναι μια κατανεμημένη υπολογιστική αρχιτεκτονική που επεκτείνει τις υπηρεσίες υπολογιστικού νέφους στην άκρη του δικτύου, πιο κοντά στους τελικούς χρήστες και τις συσκευές. Στοχεύει στη μείωση του λανθάνοντος χρόνου (latency), της χρήσης εύρους ζώνης (bandwidth usage) και του χρόνου απόκρισης (response time) μέσω της τοπικής επεξεργασίας δεδομένων, πιο κοντά στην πηγή. Η αρχιτεκτονική fog computing έχει σχεδιαστεί για να υποστηρίζει εφαρμογές που απαιτούν επεξεργασία δεδομένων χαμηλής καθυστέρησης και σε πραγματικό χρόνο, όπως το Internet of Things (IoT), το Industrial Internet of Things (IIoT) και οι έξυπνες πόλεις.

  Η αρχιτεκτονική fog computing αποτελείται από τρία επίπεδα: το στρώμα ακμών (edge layer), το στρώμα ομίχλης (fog layer) και το στρώμα νέφους (cloud layer). Το edge layer αποτελείται από συσκευές όπως αισθητήρες, ενεργοποιητές και πύλες. Το fog layer αποτελείται από ένα σύνολο διασυνδεδεμένων κόμβων ομίχλης (fog nodes) που παρέχουν υπηρεσίες υπολογισμού, αποθήκευσης και δικτύωσης. Το στρώμα cloud παρέχει υπηρεσίες υπολογιστικού νέφους, όπως αποθήκευση δεδομένων, analytics και διαχείριση.

  Το edge layer είναι υπεύθυνο για τη συλλογή δεδομένων από αισθητήρες και συσκευές και την προώθηση τους στο fog layer για περαιτέρω επεξεργασία. Το fog layer είναι υπεύθυνο για την επεξεργασία δεδομένων σε πραγματικό χρόνο, την παροχή αναλυτικών στοιχείων δεδομένων και τη διαχείριση πόρων. Τέλος, το στρώμα cloud είναι υπεύθυνο για τη μακροπρόθεσμη αποθήκευση δεδομένων, σύνθετες αναλύσεις και διαχείριση.

  Η αρχιτεκτονική fog computing έχει πολλές εφαρμογές, συμπεριλαμβανομένων των έξυπνων μεταφορών, του έξυπνου δικτύου και της υγειονομικής περίθαλψης. Στις έξυπνες μεταφορές, το fog computing μπορεί να προσφέρει διαχείριση και βελτιστοποίηση της κυκλοφορίας σε πραγματικό χρόνο. Στο έξυπνο δίκτυο, μπορεί να παρέχει παρακολούθηση και διαχείριση του ηλεκτρικού δικτύου σε πραγματικό χρόνο και στην υγειονομική περίθαλψη παρακολούθηση ασθενών σε πραγματικό χρόνο και απόκριση έκτακτης ανάγκης.

  Τα πλεονεκτήματα της αρχιτεκτονικής fog computing περιλαμβάνουν μειωμένη καθυστέρηση, βελτιωμένη απόδοση, επεκτασιμότητα και βελτιωμένη ασφάλεια. Με την τοπική επεξεργασία δεδομένων, πιο κοντά στην πηγή, το fog computing μειώνει τον λανθάνοντα χρόνο και βελτιώνει την απόδοση. Η αρχιτεκτονική είναι επεκτάσιμη, καθιστώντας την κατάλληλη για μεγάλης κλίμακας αναπτύξεις IoT και IIoT. Τέλος, παρέχει βελτιωμένη ασφάλεια με την τοπική επεξεργασία ευαίσθητων δεδομένων.

  Ωστόσο, υπάρχουν και ορισμένα μειονεκτήματα. Απαιτεί υψηλό βαθμό συντονισμού και επικοινωνίας μεταξύ των fog nodes, γεγονός που μπορεί να αυξήσει την πολυπλοκότητα και το κόστος. Επιπλέον, ο υπολογισμός ομίχλης μπορεί να αυξήσει την κατανάλωση ενέργειας και την απαγωγή θερμότητας, κάτι που μπορεί να είναι ανησυχητικό σε ορισμένες ρυθμίσεις.

- Mobile edge computing

  Το Mobile Edge Computing (MEC) είναι ένα πολλά υποσχόμενο υπολογιστικό παράδειγμα που φέρνει τους υπολογιστικούς πόρους και τις υπηρεσίες πιο κοντά στους χρήστες και τις φορητές συσκευές. Η αρχιτεκτονική MEC επιτρέπει την εκτέλεση υπολογιστικών εργασιών και την παροχή υπηρεσιών στην άκρη των δικτύων, μειώνοντας τη συμφόρηση του δικτύου και βελτιώνοντας την απόδοση των εφαρμογών. Έχει σχεδιαστεί για να συμπληρώνει το cloud computing παρέχοντας υπολογιστικούς πόρους και υπηρεσίες πιο κοντά στον χρήστη, γεγονός που μειώνει τον λανθάνοντα χρόνο και βελτιώνει την ποιότητα της υπηρεσίας.

  Η αρχιτεκτονική MEC αποτελείται από τρία κύρια στοιχεία: τον εξοπλισμό χρήστη (UE – user equipment), το edge cloud και την υποδομή δικτύου (network infrastructure). Ο εξοπλισμός χρήστη αποτελείται από κινητές συσκευές όπως smartphone, tablet και φορητές συσκευές που απαιτούν πρόσβαση σε υπολογιστικούς πόρους και υπηρεσίες. Το edge cloud είναι μια κατανεμημένη υπολογιστική πλατφόρμα που παρέχει υπολογιστικούς πόρους και υπηρεσίες κοντά στους χρήστες. Η υποδομή δικτύου είναι υπεύθυνη για τη σύνδεση του εξοπλισμού χρήστη και του edge cloud.

  Το edge cloud περιλαμβάνει διακομιστές άκρων, συσκευές αποθήκευσης και εξοπλισμό δικτύωσης. Αυτοί οι πόροι επιτρέπουν την εκτέλεση υπολογιστικών εργασιών και την παροχή υπηρεσιών όπως η προσωρινή αποθήκευση περιεχομένου, η επεξεργασία βίντεο και οι εφαρμογές επαυξημένης πραγματικότητας. Το edge cloud μπορεί επίσης να ενσωματωθεί σε υπηρεσίες υπολογιστικού νέφους για να παρέχει απρόσκοπτους και επεκτάσιμους υπολογιστικούς πόρους.

  Η αρχιτεκτονική MEC έχει πολλές εφαρμογές σε διάφορους τομείς όπως η υγειονομική περίθαλψη, οι μεταφορές και η ψυχαγωγία. Στην υγειονομική περίθαλψη, το MEC μπορεί να χρησιμοποιηθεί για απομακρυσμένη παρακολούθηση ασθενών, ανάλυση δεδομένων σε πραγματικό χρόνο και τηλεϊατρική. Στις μεταφορές, το MEC μπορεί να χρησιμοποιηθεί για τη διαχείριση της κυκλοφορίας, τα έξυπνα συστήματα μεταφοράς και την επικοινωνία από όχημα σε όχημα. Στην ψυχαγωγία, το MEC μπορεί να χρησιμοποιηθεί για παιχνίδια, ροή βίντεο και εφαρμογές επαυξημένης πραγματικότητας.

  Τα πλεονεκτήματα της αρχιτεκτονικής περιλαμβάνουν μειωμένη καθυστέρηση, βελτιωμένη απόδοση, βελτιωμένη ιδιωτικότητα και ασφάλεια και μειωμένη συμφόρηση δικτύου. Φέρνοντας τους υπολογιστικούς πόρους και τις υπηρεσίες πιο κοντά στους χρήστες, η αρχιτεκτονική MEC μειώνει τον λανθάνοντα χρόνο και βελτιώνει την απόδοση των εφαρμογών. Παρέχει επίσης ενισχυμένη ιδιωτικότητα και ασφάλεια με την επεξεργασία ευαίσθητων δεδομένων πιο κοντά στην πηγή, γεγονός που μειώνει τον κίνδυνο παραβίασης δεδομένων. Μπορεί επίσης να μειώσει τη συμφόρηση του δικτύου εκφορτώνοντας υπολογιστικές εργασίες από το κεντρικό δίκτυο.

  Ωστόσο, υπάρχουν επίσης ορισμένες προκλήσεις και περιορισμοί της αρχιτεκτονικής MEC. Μία από τις κύριες προκλήσεις είναι η διαχείριση πόρων και εφαρμογών στο edge cloud, η οποία απαιτεί αποτελεσματικούς μηχανισμούς κατανομής πόρων και εξισορρόπησης φορτίου. Μια άλλη πρόκληση είναι η ενοποίηση με τις υπάρχουσες υποδομές δικτύου και τις υπηρεσίες υπολογιστικού νέφους. Αυτό απαιτεί διαλειτουργικότητα και τυποποίηση. Επιπλέον, η αρχιτεκτονική MEC χρειάζεται μια αξιόπιστη και υψηλού εύρους ζώνης υποδομή δικτύου για να παρέχει απρόσκοπτη και αξιόπιστη συνδεσιμότητα.

1.3 Συστατικά Big Data Αρχιτεκτονικών και IoT Συστημάτων

1.3.1 Πρωτόκολλα επικοινωνίας και ανταλλαγής δεδομένων

Τα πρωτόκολλα επικοινωνίας είναι ένα σύνολο κανόνων και προτύπων που διέπουν την ανταλλαγή δεδομένων μεταξύ συσκευών ή συστημάτων. Αυτά τα πρωτόκολλα είναι απαραίτητα για τη διασφάλιση ότι τα δεδομένα που μεταδίδονται μεταξύ των συσκευών είναι ακριβή, αξιόπιστα και ασφαλή.

Στο πλαίσιο του IoT (Internet of Things), τα πρωτόκολλα επικοινωνίας διαδραματίζουν κρίσιμο ρόλο στο να επιτρέπουν στις συσκευές να επικοινωνούν μεταξύ τους και με τις εφαρμογές που βασίζονται σε cloud οι οποίες συλλέγουν και αναλύουν τα δεδομένα που δημιουργούνται από αυτές τις συσκευές. Μερικά από τα πιο κοινά πρωτόκολλα επικοινωνίας IoT περιλαμβάνουν τα MQTT, CoAP, AMQP και HTTP. (Borgia, 2014)

Στις αρχιτεκτονικές μεγάλων δεδομένων, τα πρωτόκολλα επικοινωνίας χρησιμοποιούνται για τη διευκόλυνση της ανταλλαγής δεδομένων μεταξύ διαφόρων στοιχείων της αρχιτεκτονικής, όπως συστήματα αποθήκευσης δεδομένων, μηχανές επεξεργασίας δεδομένων και εργαλεία ανάλυσης. Τα πιο συχνά χρησιμοποιούμενα πρωτόκολλα επικοινωνίας μεγάλων δεδομένων περιλαμβάνουν τα TCP/IP, HTTP και HTTPS.

Τα αποτελεσματικά πρωτόκολλα επικοινωνίας είναι απαραίτητα τόσο στην αρχιτεκτονική IoT όσο και στις αρχιτεκτονικές μεγάλων δεδομένων, επειδή επιτρέπουν την απρόσκοπτη ενοποίηση μεταξύ διαφορετικών components, διασφαλίζοντας ότι τα δεδομένα μεταδίδονται με ακρίβεια, ασφάλεια και αποτελεσματικότητα.

- MQTT
  Το MQTT (Message Queuing Telemetry Transport) είναι ένα ελαφρύ πρωτόκολλο ανταλλαγής μηνυμάτων ανοιχτού κώδικα σχεδιασμένο για συσκευές IoT (Internet of Things) και δίκτυα χαμηλού εύρους ζώνης και υψηλής καθυστέρησης. Το MQTT βασίζεται στο μοτίβο δημοσίευσης/εγγραφής (publish/subscribe) μηνυμάτων και χρησιμοποιείται ευρέως σε διάφορες εφαρμογές IoT.

  Το MQTT αποτελείται από τρία στοιχεία: έναν broker, τους clients και τα topics. Ο broker ενεργεί ως μεσάζων μεταξύ των clients και των topics. Οι clients μπορεί να είναι είτε publishers είτε subscribers. Οι publishers στέλνουν μηνύματα σε ένα συγκεκριμένο topic και οι subscribers λαμβάνουν μηνύματα από το topic στο οποίο έχουν εγγραφεί.

  Το MQTT λειτουργεί χρησιμοποιώντας ένα απλό μοντέλο δημοσίευσης/εγγραφής, με βάση το οποίο τα μηνύματα δημοσιεύονται σε ένα συγκεκριμένο topic και στη συνέχεια παραδίδονται σε όλους τους πελάτες που είναι εγγεγραμμένοι σε αυτό. Ο broker είναι υπεύθυνος για τη δρομολόγηση μηνυμάτων μεταξύ publishers και subscribers. Το πρωτόκολλο έχει σχεδιαστεί για να είναι ελαφρύ, με ελάχιστη επιβάρυνση και αποτελεσματική χρήση του εύρους ζώνης. Το MQTT χρησιμοποιεί το TCP/IP ως πρωτόκολλο μεταφοράς και υποστηρίζει επίπεδα QoS (Quality of Service) για να διασφαλίσει την αξιόπιστη παράδοση των μηνυμάτων.

  Ένα από τα κύρια πλεονεκτήματα του MQTT είναι ο ελαφρύς σχεδιασμός του, που το καθιστά ιδανικό για συσκευές IoT με περιορισμένους πόρους, όπως μνήμη και επεξεργαστική ισχύ. Υποστηρίζει επίσης επίπεδα QoS για να εξασφαλίσει αξιόπιστη παράδοση μηνυμάτων σε αναξιόπιστα δίκτυα.

Επιπλέον, το MQTT υποστηρίζει μια σειρά από πλατφόρμες και γλώσσες προγραμματισμού, καθιστώντας εύκολη την εφαρμογή του σε μια ποικιλία εφαρμογών IoT.

Ωστόσο, υπάρχουν και ορισμένοι περιορισμοί του MQTT. Δεν έχει σχεδιαστεί για επικοινωνία σε πραγματικό χρόνο και ενδέχεται να υπάρχουν καθυστερήσεις στην παράδοση μηνυμάτων ανάλογα με τις συνθήκες του δικτύου. Επιπλέον, το MQTT δεν διαθέτει ενσωματωμένες δυνατότητες ασφαλείας και εναπόκειται σε όποιον το χρησιμοποιεί στην υλοποίηση να διασφαλίσει ότι το πρωτόκολλο χρησιμοποιείται με ασφάλεια.

- CoAP
  Το CoAP (Constrained Application Protocol) είναι ένα ελαφρύ πρωτόκολλο επιπέδου εφαρμογής ανοιχτού κώδικα σχεδιασμένο για συσκευές IoT και δίκτυα χαμηλής κατανάλωσης και χαμηλού εύρους ζώνης. Το CoAP βασίζεται στις αρχές RESTful και χρησιμοποιείται ευρέως σε διάφορες εφαρμογές IoT.

  Το CoAP αποτελείται από τέσσερα στοιχεία: πελάτες, διακομιστές, μηνύματα και πόρους (clients, servers, messages, and resources). Οι πελάτες στέλνουν αιτήματα σε διακομιστές για πρόσβαση σε πόρους και οι διακομιστές απαντούν σε αυτά τα αιτήματα. Τα μηνύματα ανταλλάσσονται μεταξύ πελατών και διακομιστών και περιέχουν πληροφορίες σχετικά με το αίτημα ή την απάντηση. Οι πόροι είναι τα πράγματα με τα οποία μπορούν να αλληλεπιδράσουν οι πελάτες, όπως αισθητήρες ή ενεργοποιητές.

  Το CoAP λειτουργεί χρησιμοποιώντας ένα μοντέλο πελάτη-διακομιστή, όπου οι πελάτες στέλνουν αιτήματα σε διακομιστές για πρόσβαση σε πόρους. Το CoAP έχει σχεδιαστεί για να είναι ελαφρύ, με ελάχιστη επιβάρυνση και αποτελεσματική χρήση του εύρους ζώνης. Το CoAP χρησιμοποιεί το UDP (User Datagram Protocol) ως πρωτόκολλο μεταφοράς, σε αντίθεση με το MQTT που χρησιμοποιεί TCP/IP και υποστηρίζει διάφορες μεθόδους, όπως GET, POST, PUT και DELETE, για αλληλεπίδραση με πόρους.

  Ένα από τα κύρια πλεονεκτήματα του CoAP είναι ο ελαφρύς σχεδιασμός του, που το καθιστά ιδανικό για συσκευές IoT με περιορισμένους πόρους. Επιπλέον, το CoAP χρησιμοποιεί UDP, το οποίο απαιτεί λιγότερους πόρους σε σύγκριση με το TCP, καθιστώντας το πιο κατάλληλο για δίκτυα χαμηλής κατανάλωσης και χαμηλού εύρους ζώνης. Το CoAP υποστηρίζει επίσης τη μέθοδο παρατήρησης, η οποία επιτρέπει στους πελάτες να λαμβάνουν ειδοποιήσεις όταν αλλάζει ένας πόρος.

  Ωστόσο, υπάρχουν και ορισμένοι περιορισμοί του CoAP. Δεν έχει σχεδιαστεί για επικοινωνία σε πραγματικό χρόνο και ενδέχεται να υπάρχουν καθυστερήσεις στην παράδοση μηνυμάτων ανάλογα με τις συνθήκες του δικτύου. Επιπλέον, το CoAP δεν διαθέτει ενσωματωμένα χαρακτηριστικά ασφαλείας και εναπόκειται στον υλοποιητή να διασφαλίσει ότι το πρωτόκολλο χρησιμοποιείται με ασφάλεια.

- DDS
  Το DDS (Data Distribution Service) είναι ένα πρωτόκολλο ενδιάμεσου λογισμικού σχεδιασμένο για κατανεμημένα συστήματα, συμπεριλαμβανομένων των εφαρμογών IoT. Το DDS επιτρέπει την

αποτελεσματική και αξιόπιστη κοινή χρήση δεδομένων μεταξύ διαφορετικών συσκευών και εφαρμογών και βασίζεται σε μοντέλο publish/subscribe.

Το DDS αποτελείται από τρία κύρια στοιχεία: DataWriters, DataReaders και Topics. Οι DataWriters είναι υπεύθυνοι για τη δημοσίευση δεδομένων στο σύστημα, ενώ οι DataReaders είναι υπεύθυνοι για τη λήψη αυτών των δεδομένων. Τα topics καθορίζουν τα δεδομένα τα οποία μπορούν να δημοσιευτούν και να εγγραφούν.

Το DDS λειτουργεί χρησιμοποιώντας ένα μοντέλο publish/subscribe, κατά το οποίο οι publishers στέλνουν δεδομένα σε συγκεκριμένα topics και οι subscribers λαμβάνουν αυτά τα δεδομένα από τα topics. Το DDS χρησιμοποιεί ένα κοινό μοντέλο δεδομένων, που σημαίνει ότι όλες οι συσκευές και οι εφαρμογές έχουν πρόσβαση στα ίδια δεδομένα ταυτόχρονα. Το DDS υποστηρίζει επίσης πολιτικές Ποιότητας Υπηρεσίας (QoS), οι οποίες επιτρέπουν στο σύστημα να ιεραρχεί διαφορετικούς τύπους δεδομένων με βάση τη σημασία τους.

Ένα από τα κύρια πλεονεκτήματα του DDS είναι η ικανότητά του να χειρίζεται μεγάλες ποσότητες δεδομένων σε πραγματικό χρόνο. Το DDS έχει σχεδιαστεί για να είναι εξαιρετικά αποδοτικό και μπορεί να χειριστεί μεγάλους όγκους δεδομένων με ελάχιστο λανθάνοντα χρόνο (latency). Επιπλέον, το DDS υποστηρίζει προηγμένες πολιτικές QoS, οι οποίες επιτρέπουν στο σύστημα να δίνει προτεραιότητα στα δεδομένα βάσει παραγόντων όπως η αξιοπιστία, το εύρος ζώνης και ο λανθάνοντας χρόνος.

Ωστόσο, υπάρχουν και ορισμένοι περιορισμοί του DDS. Μπορεί να είναι πιο περίπλοκο στην εφαρμογή του σε σύγκριση με άλλα πρωτόκολλα ενδιάμεσου λογισμικού και απαιτεί περισσότερους πόρους για να λειτουργήσει αποτελεσματικά. Επιπλέον, το DDS δεν χρησιμοποιείται τόσο ευρέως όσο άλλα πρωτόκολλα ενδιάμεσου λογισμικού, γεγονός που μπορεί να κάνει πιο δύσκολη την εύρεση προγραμματιστών με εμπειρία στην υλοποίηση συστημάτων που βασίζονται σε DDS.

- AMQP
  Το AMQP (Advanced Message Queuing Protocol) είναι ένα πρωτόκολλο ανταλλαγής μηνυμάτων ανοιχτού κώδικα που έχει σχεδιαστεί για message-oriented middleware (MOM) εφαρμογές. Το AMQP επιτρέπει την αξιόπιστη, ασφαλή και αποτελεσματική ανταλλαγή μηνυμάτων μεταξύ διαφορετικών εφαρμογών και συσκευών.

  Το AMQP αποτελείται από δύο κύρια στοιχεία: τους message brokers και τους clients. Οι brokers μηνυμάτων είναι υπεύθυνοι για την αποθήκευση και τη δρομολόγηση των μηνυμάτων μεταξύ των clients, ενώ οι clients είναι υπεύθυνοι για την αποστολή και τη λήψη μηνυμάτων. Το AMQP περιλαμβάνει επίσης πολλά άλλα στοιχεία, όπως ανταλλαγές (exchanges), ουρές (queues) και δεσμεύσεις (bindings), τα οποία χρησιμοποιούνται για τη δρομολόγηση μηνυμάτων μεταξύ διαφορετικών πελατών.

  Το AMQP λειτουργεί χρησιμοποιώντας μια message-oriented προσέγγιση, στην οποία τα μηνύματα αποστέλλονται μεταξύ διαφορετικών πελατών χρησιμοποιώντας έναν message broker. Το AMQP υποστηρίζει πολλά μοτίβα ανταλλαγής μηνυμάτων, όπως publish/subscribe, request/response και

point to point. Το AMQP περιλαμβάνει επίσης προηγμένες λειτουργίες, όπως συναλλαγές, οι οποίες επιτρέπουν την ομαδοποίηση πολλαπλών μηνυμάτων σε μία μόνο συναλλαγή και την ατομική επεξεργασία τους.

Ένα από τα κύρια πλεονεκτήματα του AMQP είναι η διαλειτουργικότητά του με διαφορετικές γλώσσες προγραμματισμού και πλατφόρμες. Έχει σχεδιαστεί για να είναι αγνωστικό στη γλώσσα, πράγμα που σημαίνει ότι μπορεί να χρησιμοποιηθεί με διαφορετικές γλώσσες προγραμματισμού και πλατφόρμες. Επιπλέον, περιλαμβάνει προηγμένες λειτουργίες, όπως συναλλαγές (transactions) και έλεγχο ροής μηνυμάτων (message flow control), που επιτρέπουν αποτελεσματική και αξιόπιστη ανταλλαγή μηνυμάτων μεταξύ διαφορετικών εφαρμογών και συσκευών.

Ωστόσο, υπάρχουν και ορισμένοι περιορισμοί του AMQP. Μπορεί να είναι πιο περίπλοκο στην εφαρμογή του σε σύγκριση με άλλα πρωτόκολλα ανταλλαγής μηνυμάτων και απαιτεί περισσότερους πόρους για να λειτουργήσει αποτελεσματικά. Επιπλέον, δεν χρησιμοποιείται τόσο ευρέως όσο άλλα πρωτόκολλα ανταλλαγής μηνυμάτων, γεγονός που μπορεί να κάνει πιο δύσκολη την εύρεση προγραμματιστών με εμπειρία στην εφαρμογή συστημάτων που βασίζονται σε AMQP.

- XMPP
  Το XMPP (Extensible Messaging and Presence Protocol) είναι ένα πρωτόκολλο ανταλλαγής μηνυμάτων ανοιχτού κώδικα σχεδιασμένο για επικοινωνία σε πραγματικό χρόνο μεταξύ διαφορετικών εφαρμογών και συσκευών. Το XMPP επιτρέπει την ασφαλή και αποτελεσματική ανταλλαγή μηνυμάτων μεταξύ διαφορετικών πελατών και υποστηρίζει ένα ευρύ φάσμα λειτουργιών όπως η παρουσία (presence), η μεταφορά αρχείων (file transfer) και η συνομιλία πολλών χρηστών (multi-user chat).

  Το XMPP αποτελείται από τρία κύρια στοιχεία: πελάτες (clients), διακομιστές (servers) και επεκτάσεις (extensions). Οι clients είναι εφαρμογές που χρησιμοποιούν XMPP για την αποστολή και λήψη μηνυμάτων, οι διακομιστές είναι υπεύθυνοι για την αποθήκευση και τη δρομολόγηση μηνυμάτων μεταξύ των πελατών και οι επεκτάσεις είναι πρόσθετες δυνατότητες που μπορούν να προστεθούν στο XMPP.

  Το XMPP λειτουργεί χρησιμοποιώντας μια αρχιτεκτονική client/server, στην οποία οι πελάτες επικοινωνούν με διακομιστές για να στείλουν και να λάβουν μηνύματα. Το XMPP περιλαμβάνει επίσης πολλές προηγμένες λειτουργίες, όπως η παρουσία, η οποία επιτρέπει στους χρήστες να βλέπουν την online κατάσταση άλλων χρηστών και τη μεταφορά αρχείων, η οποία επιτρέπει στους χρήστες να μοιράζονται αρχεία μεταξύ τους.

  Ένα από τα κύρια πλεονεκτήματα του XMPP είναι η ευελιξία (flexibility) και η επεκτασιμότητα (extensibility) του. Το XMPP έχει σχεδιαστεί για να είναι εύκολα επεκτάσιμο, πράγμα που σημαίνει ότι μπορούν να προστεθούν νέες δυνατότητες και λειτουργίες στο πρωτόκολλο χωρίς να επηρεαστούν οι υπάρχουσες εφαρμογές και συσκευές. Επιπλέον, το XMPP χρησιμοποιείται ευρέως σε διάφορους κλάδους και έχει μια μεγάλη και ενεργή κοινότητα προγραμματιστών.

Ωστόσο, υπάρχουν και ορισμένοι περιορισμοί του XMPP. Μπορεί να είναι πιο περίπλοκο στην εφαρμογή του σε σύγκριση με άλλα πρωτόκολλα ανταλλαγής μηνυμάτων και μπορεί να απαιτεί περισσότερους πόρους για να λειτουργήσει αποτελεσματικά. Επιπλέον, το XMPP δεν χρησιμοποιείται τόσο ευρέως όσο άλλα πρωτόκολλα ανταλλαγής μηνυμάτων όπως το MQTT ή το AMQP, γεγονός που μπορεί να κάνει πιο δύσκολη την εύρεση προγραμματιστών με εμπειρία στην εφαρμογή συστημάτων που βασίζονται σε XMPP.

1.3.2 Τεχνικές αποθήκευσης δεδομένων μεγάλης κλίμακας

Η αποθήκευση μεγάλων δεδομένων στο πλαίσιο του IoT (Internet of Things) μπορεί να είναι δύσκολη λόγω του μεγάλου όγκου, της ποικιλίας και της ταχύτητας των δεδομένων που παράγονται από συσκευές IoT.

Ορισμένες από τις δυσκολίες τις οποίες συνεπάγεται η αποθήκευση μεγάλων δεδομένων είναι:

Όγκος δεδομένων (data volume): Οι συσκευές IoT παράγουν τεράστιες ποσότητες δεδομένων, τα οποία μπορούν γρήγορα να κατακλύσουν τα παραδοσιακά συστήματα αποθήκευσης. Η αποθήκευση αυτών των δεδομένων απαιτεί επεκτάσιμες και κατανεμημένες λύσεις αποθήκευσης που μπορούν να χειριστούν τον μεγάλο όγκο δεδομένων.

Ποικιλία δεδομένων (data variety): Οι συσκευές IoT δημιουργούν δεδομένα σε διάφορες μορφές, συμπεριλαμβανομένων των δομημένων, ημιδομημένων και μη δομημένων δεδομένων. Αυτό καθιστά δύσκολη την αποθήκευση των δεδομένων σε μια παραδοσιακή σχεσιακή βάση δεδομένων, η οποία έχει σχεδιαστεί για να χειρίζεται δομημένα δεδομένα.

Ταχύτητα δεδομένων (data velocity) : Οι συσκευές IoT δημιουργούν δεδομένα σε πραγματικό χρόνο ή σχεδόν σε πραγματικό χρόνο, πράγμα που σημαίνει ότι το σύστημα αποθήκευσης πρέπει να είναι σε θέση να χειρίζεται υψηλό ποσοστό εισερχόμενων δεδομένων. Αυτό απαιτεί ένα σύστημα αποθήκευσης που είναι βελτιστοποιημένο για λειτουργίες γρήγορης εγγραφής και μπορεί να αποθηκεύει και να ανακτά γρήγορα δεδομένα.

Ασφάλεια δεδομένων (data security): Οι συσκευές IoT συχνά συλλέγουν ευαίσθητα δεδομένα, όπως προσωπικές πληροφορίες και επιχειρηματικά δεδομένα. Αυτά τα δεδομένα πρέπει να αποθηκεύονται με ασφάλεια για να αποτρέπεται η μη εξουσιοδοτημένη πρόσβαση και να διασφαλίζεται το απόρρητο.

Ενοποίηση δεδομένων (data integration): Τα δεδομένα IoT δημιουργούνται συχνά από πολλαπλές πηγές και πρέπει να ενσωματωθούν και να αποθηκευτούν με τρόπο που να επιτρέπει την εύκολη ανάλυση και ανάκτηση. Αυτό απαιτεί ένα σύστημα αποθήκευσης που μπορεί να ενσωματώσει δεδομένα από διάφορες πηγές και να παρέχει μια ενοποιημένη προβολή των δεδομένων.

Επεξεργασία δεδομένων (data processing): Η αποθήκευση μεγάλων δεδομένων στο IoT δεν αφορά μόνο την αποθήκευση αλλά και την επεξεργασία των δεδομένων σε πραγματικό χρόνο για την απόκτηση πληροφοριών. Αυτό απαιτεί ένα σύστημα αποθήκευσης με ενσωματωμένα εργαλεία ανάλυσης και δυνατότητα επεξεργασίας δεδομένων σε πραγματικό χρόνο.

Συνολικά, η αποθήκευση μεγάλων δεδομένων στο IoT απαιτεί μια λύση αποθήκευσης που είναι επεκτάσιμη, ευέλικτη, ασφαλής και μπορεί να χειριστεί μεγάλο όγκο δεδομένων. Είναι σημαντικό να

επιλέγεται κάθε φορά μια αποθηκευτική λύση βελτιστοποιημένη για τις συγκεκριμένες ανάγκες της εφαρμογής IoT, η οποία να μπορεί να υποστηρίξει την επεξεργασία και ανάλυση δεδομένων σε πραγματικό χρόνο.

Ακολουθούν μερικοί από τους πιο συνήθεις τρόπους αποθήκευσης big data.

- HDFS
  Το HDFS (Hadoop Distributed File System) (Shvachko, et al., 2010) είναι ένα κατανεμημένο σύστημα αρχείων που παρέχει εξαιρετικά αξιόπιστη, επεκτάσιμη και ανεκτική σε σφάλματα αποθήκευση για επεξεργασία μεγάλων δεδομένων. Έχει σχεδιαστεί για να λειτουργεί σε commodity hardware και αποτελεί βασικό στοιχείο του project Apache Hadoop. Το HDFS χρησιμοποιείται για την αποθήκευση και τη διαχείριση μεγάλων συνόλων δεδομένων, που κυμαίνονται από gigabyte έως petabyte, σε ένα μεγάλο σύμπλεγμα (cluster) μηχανών.

  Το HDFS αποτελείται από δύο κύρια components: NameNode και DataNode. Το NameNode είναι ο κεντρικός κόμβος που διαχειρίζεται τα μεταδεδομένα του συστήματος αρχείων και διατηρεί μια ιεραρχία χώρου ονομάτων (namespace hierarchy). Παρακολουθεί τη θέση των μπλοκ δεδομένων στο σύμπλεγμα και χρησιμεύει ως το κύριο σημείο επαφής για εφαρμογές πελατών (client applications). Το DataNode είναι υπεύθυνο για την αποθήκευση και την ανάκτηση μπλοκ δεδομένων, τα οποία είναι τα πραγματικά δεδομένα που είναι αποθηκευμένα στο HDFS. Τα DataNodes επικοινωνούν με το NameNode για να αναφέρουν την κατάσταση της μνήμης τους και τη διαθεσιμότητα των μπλοκ δεδομένων.

  Το HDFS λειτουργεί σπάζοντας μεγάλα αρχεία σε μικρότερα κομμάτια που ονομάζονται μπλοκ δεδομένων και διανέμοντας τα σε όλο το cluster. Κάθε μπλοκ δεδομένων αναπαράγεται σε πολλούς DataNodes για να παρέχει ανοχή σφαλμάτων (fault tolerance) και υψηλή διαθεσιμότητα (high availability). Ο παράγοντας αναπαραγωγής μπορεί να διαμορφωθεί και η προεπιλεγμένη τιμή είναι τρία. Το HDFS χρησιμοποιεί ένα μοντέλο write-once-read-many (WORM), που σημαίνει ότι μόλις γραφτεί ένα αρχείο, δεν μπορεί να τροποποιηθεί. Αντίθετα, γίνονται τροποποιήσεις γράφοντας μια νέα έκδοση του αρχείου, η οποία δημιουργεί ένα νέο σύνολο μπλοκ δεδομένων.

  Ένα από τα σημαντικά πλεονεκτήματα του HDFS είναι ο ανθεκτικός σχεδιασμός του. Το HDFS αναπαράγει μπλοκ δεδομένων σε πολλούς κόμβους δεδομένων, πράγμα που σημαίνει ότι ακόμη και αν κάποιοι κόμβοι αποτύχουν, τα δεδομένα εξακολουθούν να είναι προσβάσιμα. Το HDFS χρησιμοποιεί επίσης μια αρχιτεκτονική NameNode και DataNode που παρέχει μια εξαιρετικά διαθέσιμη υπηρεσία. Εάν το NameNode αποτύχει, ένα NameNode σε κατάσταση αναμονής μπορεί να αναλάβει την ευθύνη και οι DataNodes συνεχίζουν να λειτουργούν χωρίς διακοπή.

  Ένα άλλο πλεονέκτημα του HDFS είναι η επεκτασιμότητα του. Το HDFS μπορεί να χειριστεί σύνολα δεδομένων που κυμαίνονται από gigabyte έως petabyte και μπορεί εύκολα να προσθέσει νέους κόμβους στο σύμπλεγμα για να αυξήσει τη χωρητικότητα αποθήκευσης και την απόδοση. Το HDFS έχει επίσης σχεδιαστεί για να λειτουργεί με το MapReduce, ένα κατανεμημένο πλαίσιο επεξεργασίας δεδομένων που επιτρέπει την παράλληλη επεξεργασία μεγάλων συνόλων δεδομένων.

Ωστόσο, το HDFS έχει επίσης ορισμένα μειονεκτήματα. Ένας από τους κύριους περιορισμούς του HDFS είναι η απόδοσή του για μικρά αρχεία. Το HDFS είναι βελτιστοποιημένο για την αποθήκευση και την επεξεργασία μεγάλων αρχείων και ενδέχεται να μην έχει καλή απόδοση για μικρά αρχεία λόγω των επιβαρύνσεων που απαιτούνται για την αποθήκευση και τη διαχείριση μεταδεδομένων. Ένας άλλος περιορισμός είναι η έλλειψη υποστήριξης για τυχαίες εγγραφές, πράγμα που σημαίνει ότι η προσθήκη δεδομένων σε ένα υπάρχον αρχείο μπορεί να είναι δύσκολη.

Συμπερασματικά, το HDFS είναι ένα κατανεμημένο σύστημα αρχείων που παρέχει εξαιρετικά αξιόπιστη, επεκτάσιμη και ανεκτή σε σφάλματα αποθήκευση για την επεξεργασία μεγάλων δεδομένων. Η αρχιτεκτονική του αποτελείται από στοιχεία NameNode και DataNode και λειτουργεί κατανέμοντας μπλοκ δεδομένων σε όλο το cluster. Το HDFS έχει πλεονεκτήματα όπως η ανοχή σφαλμάτων, η επεκτασιμότητα και η ενσωμάτωση με το MapReduce. Ωστόσο, έχει επίσης περιορισμούς όπως η απόδοση για μικρά αρχεία και η έλλειψη υποστήριξης για τυχαίες εγγραφές.

- Amazon S3 (Object Storage)
  Το Amazon S3 (Simple Storage Service) είναι μια υπηρεσία αποθήκευσης αντικειμένων που βασίζεται σε cloud που παρέχεται από την Amazon Web Services (AWS). Έχει σχεδιαστεί για να παρέχει επεκτάσιμη, ασφαλή και άμεσα διαθέσιμη μνήμη για δημιουργία αντιγράφων ασφαλείας, αρχειοθέτηση και δεδομένα εφαρμογών. Το Amazon S3 είναι βασικό συστατικό πολλών εφαρμογών και υπηρεσιών που βασίζονται σε cloud και χρησιμοποιείται ευρέως από επιχειρήσεις όλων των μεγεθών.

  Το Amazon S3 αποτελείται από «κουβάδες» (buckets), αντικείμενα (objects) και μεταδεδομένα (metadata). Ένα bucket είναι ένα κοντέινερ για objects και κάθε object περιέχει δεδομένα και μεταδεδομένα. Τα μεταδεδομένα περιέχουν πληροφορίες σχετικά με το αντικείμενο, όπως την ημερομηνία δημιουργίας, το μέγεθος και τον τύπο περιεχομένου. Το Amazon S3 έχει σχεδιαστεί για να παρέχει υψηλή αντοχή και διαθεσιμότητα, με εγγύηση ανθεκτικότητας 99,999999999%.

  Οι χρήστες μπορούν να δημιουργήσουν buckets, να ανεβάσουν αντικείμενα σε αυτούς και στη συνέχεια να ανακτήσουν ή να διαγράψουν αντικείμενα όπως επιθυμούν. Το Amazon S3 έχει σχεδιαστεί για να είναι εξαιρετικά επεκτάσιμο, με δυνατότητα αποθήκευσης τρισεκατομμυρίων αντικειμένων και διαχείρισης εκατομμυρίων αιτημάτων ανά δευτερόλεπτο. Μπορεί να χειριστεί μεγάλες ποσότητες δεδομένων και οι χρήστες μπορούν εύκολα να προσθέσουν ή να αφαιρέσουν χωρητικότητα αποθήκευση. Είναι επίσης εξαιρετικά διαθέσιμο (highly available), με τη δυνατότητα αναπαραγωγής αντικειμένων σε πολλές ζώνες διαθεσιμότητας (availability zones) για να διασφαλιστεί ότι τα δεδομένα είναι πάντα διαθέσιμα.

  Ένα άλλο πλεονέκτημα του Amazon S3 είναι τα χαρακτηριστικά ασφαλείας του. Παρέχει κρυπτογράφηση σε κατάσταση ηρεμίας (rest) και κατά τη μεταφορά (transit), με υποστήριξη για κρυπτογράφηση από την πλευρά του διακομιστή, κρυπτογράφηση από την πλευρά του πελάτη και Υπηρεσία Διαχείρισης Κλειδιών AWS (Key Management Service - KMS). Παρέχει επίσης δυνατότητες ελέγχου πρόσβασης, με δυνατότητα ορισμού αναλυτικών δικαιωμάτων για χρήστες και ομάδες.

Ωστόσο, έχει ορισμένα μειονεκτήματα. Ένας από τους κύριους περιορισμούς είναι το μοντέλο τιμολόγησης του, το οποίο μπορεί να είναι περίπλοκο και δύσκολο να προβλεφθεί. Οι χρήστες χρεώνονται για αποθήκευση, μεταφορά δεδομένων και αιτήματα, τα οποία μπορεί να διαφέρουν ανάλογα με την περιοχή, την κατηγορία αποθήκευσης και τα μοτίβα χρήσης. Το Amazon S3 έχει επίσης περιορισμένη υποστήριξη για έκδοση (versioning) και αναζήτηση (searching), γεγονός που μπορεί να κάνει δύσκολη τη διαχείριση μεγάλων ποσοτήτων δεδομένων.

- Βάσεις Δεδομένων

  Οι βάσεις δεδομένων χρησιμοποιούνται συχνά για την αποθήκευση μεγάλων δεδομένων σε συστήματα IoT. Οι συσκευές IoT παράγουν τεράστιες ποσότητες δεδομένων που πρέπει να συλλεχθούν, να επεξεργαστούν και να αναλυθούν για να παράσχουν πληροφορίες και να οδηγήσουν στη λήψη αποφάσεων. Οι βάσεις δεδομένων μας δίνουν έναν δομημένο τρόπο αποθήκευσης και διαχείρισης αυτών των δεδομένων, κάνοντας εφικτή την εύκολη αναζήτηση και ανάλυσή τους.

  Υπάρχουν πολλοί τύποι βάσεων δεδομένων που μπορούν να χρησιμοποιηθούν σε συστήματα IoT, συμπεριλαμβανομένων των σχεσιακών βάσεων δεδομένων, των βάσεων δεδομένων NoSQL και των βάσεων δεδομένων χρονοσειρών. Κάθε τύπος βάσης δεδομένων έχει τα δικά του δυνατά και αδύνατα σημεία και η επιλογή της βάσης δεδομένων θα εξαρτηθεί από τις συγκεκριμένες απαιτήσεις του συστήματος IoT.

  Οι σχεσιακές βάσεις δεδομένων, όπως η MySQL και η PostgreSQL, παρότι ενίοτε χρησιμοποιούνται σε συστήματα IoT για την ικανότητά τους να αποθηκεύουν δομημένα δεδομένα και να υποστηρίζουν πολύπλοκα ερωτήματα, συχνά δεν επαρκούν για σενάρια μεγάλων δεδομένων, όπου οι όγκοι δεδομένων είναι πολύ μεγάλοι ή τα δεδομένα είναι πολύ αδόμητα. Οι σχεσιακές βάσεις δεδομένων έχουν σχεδιαστεί για να χειρίζονται δομημένα δεδομένα με ένα σταθερό σχήμα, όπου τα δεδομένα οργανώνονται σε πίνακες με προκαθορισμένες στήλες και σχέσεις μεταξύ τους. Ενώ οι σχεσιακές βάσεις δεδομένων μπορούν να είναι αποτελεσματικές για τη διαχείριση συνόλων δεδομένων μικρού έως μεσαίου μεγέθους, ενδέχεται να αντιμετωπίζουν πρόβλημα σε σενάρια big data λόγω περιορισμών στην επεκτασιμότητα και την απόδοση.

  Τα μεγάλα δεδομένα συχνά περιλαμβάνουν τεράστιους όγκους μη δομημένων ή ημιδομημένων δεδομένων, όπως δεδομένα αισθητήρων, αρχεία καταγραφής ή δεδομένα μέσων κοινωνικής δικτύωσης, τα οποία μπορεί να είναι δύσκολο να αποθηκευτούν και να αναλυθούν χρησιμοποιώντας παραδοσιακές σχεσιακές βάσεις δεδομένων. Επιπλέον, η επεξεργασία μεγάλων δεδομένων απαιτεί συχνά κατανεμημένους υπολογισμούς και παράλληλη επεξεργασία, κάτι που μπορεί να είναι δύσκολο να επιτευχθεί με τα παραδοσιακά συστήματα σχεσιακών βάσεων δεδομένων.

  Για την αντιμετώπιση αυτών των προκλήσεων, έχουν προκύψει εναλλακτικές τεχνολογίες βάσεων δεδομένων, όπως βάσεις δεδομένων NoSQL, βάσεις δεδομένων χρονοσειρών και κατανεμημένες βάσεις δεδομένων, που έχουν σχεδιαστεί ειδικά για να χειρίζονται big data. Αυτές οι βάσεις δεδομένων συχνά προσφέρουν δυνατότητες όπως οριζόντια επεκτασιμότητα (horizontal scalability), υψηλή διαθεσιμότητα (high availability) και κατανεμημένο υπολογισμό (distributed computing) που τις καθιστούν καταλληλότερες από τις παραδοσιακές σχεσιακές βάσεις δεδομένων για σενάρια μεγάλων δεδομένων.

Οι βάσεις δεδομένων NoSQL είναι μια ευρεία κατηγορία μη σχεσιακών βάσεων δεδομένων που προσφέρουν ευελιξία, επεκτασιμότητα και πλεονεκτήματα απόδοσης σε σχέση με τις παραδοσιακές σχεσιακές βάσεις δεδομένων. Έχουν σχεδιαστεί για να χειρίζονται διαφορετικούς και πολύπλοκους τύπους δεδομένων, συμπεριλαμβανομένων μη δομημένων, ημι-δομημένων και δομημένων δεδομένων, και μπορούν να υποστηρίξουν τεράστια επεκτασιμότητα σε κατανεμημένα υπολογιστικά περιβάλλοντα.

Οι βάσεις δεδομένων NoSQL χρησιμοποιούνται συνήθως σε εφαρμογές μεγάλων δεδομένων, όπως εφαρμογές κλίμακας ιστού, αναλυτικά στοιχεία σε πραγματικό χρόνο και συστήματα IoT. Παραδείγματα δημοφιλών βάσεων δεδομένων NoSQL περιλαμβάνουν MongoDB, Cassandra και HBase.

Οι βάσεις δεδομένων χρονοσειρών είναι ένας εξειδικευμένος τύπος βάσης δεδομένων που έχουν βελτιστοποιηθεί για την αποθήκευση και την ανάκτηση δεδομένων με χρονική σήμανση. Έχουν σχεδιαστεί για να χειρίζονται μεγάλους όγκους δεδομένων που παράγονται με την πάροδο του χρόνου, όπως δεδομένα αισθητήρων, οικονομικά δεδομένα και δεδομένα καταγραφής, και μπορούν να υποστηρίξουν ανάλυση και επεξεργασία σε πραγματικό χρόνο.

Οι βάσεις δεδομένων χρονοσειρών προσφέρουν πολλά πλεονεκτήματα για σενάρια μεγάλων δεδομένων, συμπεριλαμβανομένης της αποτελεσματικής αποθήκευσης και ανάκτησης δεδομένων χρονοσειρών, υποστήριξης κατανεμημένων υπολογιστών και προηγμένων δυνατοτήτων ανάλυσης, όπως η πρόβλεψη και η ανίχνευση ανωμαλιών. Παραδείγματα δημοφιλών βάσεων δεδομένων χρονοσειρών περιλαμβάνουν τα InfluxDB, TimescaleDB και OpenTSDB.

- Elasticsearch
  Το Elasticsearch είναι μια κατανεμημένη, ανοιχτού κώδικα μηχανή αναζήτησης και ανάλυσης που έχει σχεδιαστεί για τη διαχείριση μεγάλων όγκων δομημένων και μη δομημένων δεδομένων. Είναι χτισμένο πάνω στη βιβλιοθήκη μηχανών αναζήτησης Apache Lucene και είναι σε θέση να κάνει indexing και να αναζητά δεδομένα σε σχεδόν πραγματικό χρόνο, καθιστώντας το μια δημοφιλή επιλογή για εφαρμογές που απαιτούν δυνατότητες γρήγορης αναζήτησης και ανάλυσης.

  Πλεονεκτήματα:

  Επεκτασιμότητα: Το Elasticsearch έχει σχεδιαστεί για οριζόντια κλίμακα, που σημαίνει ότι μπορεί να διανεμηθεί σε πολλούς κόμβους για να χειριστεί μεγάλους όγκους δεδομένων και κίνησης. Αυτό το καθιστά μια καλή επιλογή για εφαρμογές που πρέπει να χειρίζονται μεγάλα δεδομένα και μεγάλους όγκους επισκεψιμότητας.

  Αναζήτηση και ανάλυση σε πραγματικό χρόνο: Το Elasticsearch παρέχει δυνατότητες αναζήτησης και ανάλυσης σχεδόν σε πραγματικό χρόνο, καθιστώντας το μια καλή επιλογή για εφαρμογές που απαιτούν γρήγορους χρόνους απόκρισης, όπως το ηλεκτρονικό εμπόριο, τα μέσα κοινωνικής δικτύωσης και οι εφαρμογές παρακολούθησης.

  Ευελιξία: Το Elasticsearch υποστηρίζει ένα ευρύ φάσμα τύπων δεδομένων, συμπεριλαμβανομένων δομημένων, μη δομημένων και ημιδομημένων δεδομένων, και παρέχει ισχυρές δυνατότητες

αναζήτησης και ανάλυσης, συμπεριλαμβανομένης της αναζήτησης πλήρους κειμένου, της πολύπλευρης αναζήτησης και της γεωχωρικής αναζήτησης.

Ανοιχτός κώδικας: Το Elasticsearch είναι μια τεχνολογία ανοιχτού κώδικα, που σημαίνει ότι είναι δωρεάν στη χρήση και μπορεί να τροποποιηθεί και να επεκταθεί από προγραμματιστές.

Μειονεκτήματα:

Πολυπλοκότητα: Το Elasticsearch μπορεί να είναι πολύπλοκο στη ρύθμιση και τη διαμόρφωση, ιδιαίτερα σε κατανεμημένα περιβάλλοντα. Αυτό μπορεί να απαιτεί σημαντική τεχνογνωσία και πόρους για να διασφαλιστεί ότι το σύστημα είναι σωστά διαμορφωμένο και βελτιστοποιημένο για απόδοση.

Χρήση μνήμης: Το Elasticsearch μπορεί να είναι βερύ στη μνήμη, ιδιαίτερα κατά την ευρετηρίαση μεγάλων όγκων δεδομένων. Αυτό μπορεί να απαιτήσει σημαντικούς πόρους υλικού για την υποστήριξη μεγάλων αναπτύξεων.

Συνέπεια δεδομένων: Το Elasticsearch έχει σχεδιαστεί για να παρέχει δυνατότητες αναζήτησης και ανάλυσης σχεδόν σε πραγματικό χρόνο, κάτι που μπορεί να καταστήσει δύσκολη τη διατήρηση της συνέπειας των δεδομένων μεταξύ των κατανεμημένων κόμβων σε ένα σύμπλεγμα.

Πως δουλεύει:

Το Elasticsearch λειτουργεί με την αποθήκευση δεδομένων σε ένα κατανεμημένο ευρετήριο, το οποίο χωρίζεται σε θραύσματα (shards) που κατανέμονται σε πολλούς κόμβους (nodes) σε ένα σύμπλεγμα (cluster). Όταν εκδίδεται ένα ερώτημα, το Elasticsearch διανέμει το ερώτημα στα θραύσματα και επιστρέφει τα αποτελέσματα σε σχεδόν πραγματικό χρόνο.

Το Elasticsearch υποστηρίζει μια σειρά τύπων ερωτημάτων, συμπεριλαμβανομένης της αναζήτησης πλήρους κειμένου, της πολύπλευρης αναζήτησης και της γεωχωρικής αναζήτησης, και παρέχει ισχυρές δυνατότητες ανάλυσης και συγκέντρωσης που επιτρέπουν στους χρήστες να εξερευνούν και να αναλύουν τα δεδομένα τους με διάφορους τρόπους.

Το Elasticsearch αναπτύσσεται συνήθως σε συνδυασμό με άλλα στοιχεία του Elastic Stack, συμπεριλαμβανομένου του Kibana για οπτικοποίηση και παρακολούθηση, του Logstash για απορρόφηση και επεξεργασία δεδομένων και Beats για ελαφρούς αποστολείς δεδομένων.

Συνολικά, το Elasticsearch παρέχει μια ισχυρή μηχανή αναζήτησης και ανάλυσης που είναι κατάλληλη για το χειρισμό μεγάλου όγκου δεδομένων σε κατανεμημένα περιβάλλοντα. Ενώ μπορεί να είναι πολύπλοκη η εγκατάσταση και η συντήρηση του, προσφέρει σημαντικά οφέλη όσον αφορά την επεκτασιμότητα, την ευελιξία και τις δυνατότητες αναζήτησης και ανάλυσης σε πραγματικό χρόνο.

1.3.3 Τεχνολογίες streaming και επεξεργασίας δεδομένων σε πραγματικό χρόνο

Οι τεχνικές ροής και οι τεχνικές επεξεργασίας δεδομένων σε πραγματικό χρόνο χρησιμοποιούνται για τη διαχείριση δεδομένων που παράγονται συνεχώς και γρήγορα, συνήθως σε μεγάλους όγκους. Οι τεχνικές επιτρέπουν στους οργανισμούς να επεξεργάζονται και να αναλύουν δεδομένα σε πραγματικό χρόνο, όπως αυτά παράγονται, παρά μετά τη συλλογή τους.

Οι τεχνικές ροής χρησιμοποιούνται για την επεξεργασία δεδομένων που παράγονται με συνεχή και απεριόριστο τρόπο. Αυτές οι τεχνικές επιτρέπουν την επεξεργασία δεδομένων μόλις γίνουν διαθέσιμα και χρησιμοποιούνται συνήθως σε εφαρμογές που απαιτούν επεξεργασία δεδομένων σε πραγματικό χρόνο, όπως χρηματοοικονομικές συναλλαγές, ανίχνευση απάτης και αναλύσεις μέσων κοινωνικής δικτύωσης. Συνήθως περιλαμβάνουν τη χρήση μηχανών επεξεργασίας ροής, οι οποίες μπορούν να χειριστούν μεγάλους όγκους δεδομένων σε πραγματικό χρόνο και μπορούν να εκτελέσουν λειτουργίες όπως φιλτράρισμα, συγκέντρωση και μετατροπή δεδομένων καθώς ρέουν μέσω του συστήματος.

Οι τεχνικές επεξεργασίας δεδομένων σε πραγματικό χρόνο χρησιμοποιούνται για την ανάλυση δεδομένων σε πραγματικό χρόνο, όπως αυτά παράγονται. Αυτές οι τεχνικές επιτρέπουν στους οργανισμούς να εντοπίζουν γρήγορα πρότυπα, ανωμαλίες και άλλες πληροφορίες που μπορούν να χρησιμοποιηθούν για διάφορους σκοπούς. Οι τεχνικές επεξεργασίας δεδομένων σε πραγματικό χρόνο συνήθως περιλαμβάνουν τη χρήση συστημάτων επεξεργασίας σύνθετων συμβάντων (complex event processing - CEP), τα οποία μπορούν να επεξεργάζονται μεγάλους όγκους δεδομένων σε πραγματικό χρόνο και μπορούν να ανιχνεύσουν πολύπλοκα μοτίβα και συσχετίσεις σε πολλαπλές ροές δεδομένων. (Gubbi, et al., 2013)

Μερικές κοινές τεχνικές που χρησιμοποιούνται για τη ροή και την επεξεργασία δεδομένων σε πραγματικό χρόνο περιλαμβάνουν:

- Απορρόφηση δεδομένων (data ingestion): Αυτό περιλαμβάνει τη συλλογή δεδομένων από διάφορες πηγές και τη διάθεση τους για επεξεργασία.
- Κανονικοποίηση δεδομένων (data normalization): Αυτό περιλαμβάνει τυποποίηση μορφών και δομών δεδομένων για να διασφαλιστεί η συνέπεια και η ευκολία επεξεργασίας.
- Επεξεργασία ροής (stream processing): Περιλαμβάνει την επεξεργασία δεδομένων σε πραγματικό χρόνο καθώς παράγονται, συνήθως χρησιμοποιώντας μηχανές επεξεργασίας ροής.
- Επεξεργασία σύνθετων συμβάντων (complex event processing): Περιλαμβάνει τον εντοπισμό και την ανάλυση πολύπλοκων προτύπων και συσχετίσεων σε ροές δεδομένων σε πραγματικό χρόνο, συνήθως χρησιμοποιώντας συστήματα CEP.
- Οπτικοποίηση δεδομένων (data visualization): Περιλαμβάνει την παρουσίαση δεδομένων σε πραγματικό χρόνο σε οπτική μορφή που είναι εύκολο να κατανοηθεί και να ερμηνευτεί.

Οι τεχνολογίες ροής και οι τεχνολογίες επεξεργασίας δεδομένων σε πραγματικό χρόνο αναφέρονται στο λογισμικό και τα εργαλεία που χρησιμοποιούνται για την εφαρμογή τεχνικών ροής και επεξεργασίας δεδομένων σε πραγματικό χρόνο. Ακολουθούν ορισμένα παραδείγματα τεχνολογιών ροής και τεχνολογιών επεξεργασίας δεδομένων σε πραγματικό χρόνο:

- Apache Kafka
  Το Apache Kafka είναι μια κατανεμημένη πλατφόρμα ροής που επιτρέπει την επεξεργασία μεγάλων ποσοτήτων δεδομένων σε πραγματικό χρόνο. Χρησιμοποιείται ευρέως για την κατασκευή αγωγών

δεδομένων σε πραγματικό χρόνο και εφαρμογών ροής, προσφέροντας υψηλή απόδοση, επεκτασιμότητα και ανοχή σφαλμάτων. (Kafka®, 2021)

Πως δουλεύει:
Το Apache Kafka βασίζεται σε ένα μοντέλο μηνυμάτων δημοσίευσης-συνδρομής (publish-subscribe), στο οποίο τα δεδομένα παράγονται από εκδότες (publishers) και καταναλώνονται από τους συνδρομητές (subscribers). Τα δεδομένα είναι οργανωμένα σε θέματα (topics) και κάθε θέμα μπορεί να έχει πολλούς παραγωγούς (producers) και καταναλωτές (consumers). Ένα Kafka cluster αποτελείται από πολλούς brokers που αποθηκεύουν και διαχειρίζονται τα δεδομένα για κάθε topic. Οι producers στέλνουν δεδομένα σε ένα συγκεκριμένο topic και τα δεδομένα αποθηκεύονται σε partitions σε πολλούς brokers. Οι consumers εγγράφονται σε ένα ή περισσότερα topics και λαμβάνουν τα δεδομένα σε πραγματικό χρόνο καθώς παράγονται.

Το Kafka είναι εξαιρετικά επεκτάσιμο, καθώς επιτρέπει την προσθήκη περισσότερων brokers στο cluster για να χειριστεί έναν αυξανόμενο όγκο δεδομένων. Προσφέρει επίσης υψηλή απόδοση και χαμηλή καθυστέρηση, καθιστώντας το μια αποτελεσματική πλατφόρμα για επεξεργασία δεδομένων σε πραγματικό χρόνο.

Πλεονεκτήματα:
Επεκτασιμότητα: Μπορεί να κλιμακωθεί οριζόντια προσθέτοντας περισσότερους brokers στο cluster, επιτρέποντάς του να χειρίζεται αυξανόμενους όγκους δεδομένων.

Υψηλή απόδοση και χαμηλή καθυστέρηση: Το Kafka προσφέρει υψηλή απόδοση, επιτρέποντας την αποτελεσματική επεξεργασία μεγάλων όγκων δεδομένων σε πραγματικό χρόνο.

Ανοχή σφαλμάτων: Ο μηχανισμός αναπαραγωγής (replication mechanism) του Kafka διασφαλίζει ότι τα δεδομένα δεν θα χαθούν σε περίπτωση αποτυχίας του broker.

Ευελιξία: Το Kafka προσφέρει ένα ευρύ φάσματhird-party integrations, καθιστώντας εύκολη την ενσωμάτωσή του σε υπάρχουσες ροές εργασίας επεξεργασίας δεδομένων.

Μειονεκτήματα:
Πολυπλοκότητα: Το Kafka μπορεί να είναι πολύπλοκο στη δημιουργία και τη διαχείριση, απαιτώντας ένα ορισμένο επίπεδο τεχνογνωσίας για να λειτουργήσει αποτελεσματικά.

Resource-intensive: Απαιτεί πολλούς πόρους, όπως σημαντική ποσότητα μνήμης και επεξεργαστικής ισχύος για να λειτουργήσει αποτελεσματικά.

Περιορισμένες δυνατότητες αναλυτικών στοιχείων (Limited analytics capabilities): Ενώ το Kafka προσφέρει δυνατότητες επεξεργασίας δεδομένων σε πραγματικό χρόνο, δεν προσφέρει προηγμένες δυνατότητες ανάλυσης, όπως μηχανική εκμάθηση ή οπτικοποίηση δεδομένων.

Συνολικά, το Apache Kafka είναι μια ισχυρή και ευρέως χρησιμοποιούμενη πλατφόρμα διανομής ροής που επιτρέπει την επεξεργασία δεδομένων σε πραγματικό χρόνο σε κλίμακα. Η ανθεκτική σε

σφάλματα αρχιτεκτονική, οι δυνατότητες επεξεργασίας δεδομένων σε πραγματικό χρόνο και η ευελιξία του το καθιστούν ιδανική πλατφόρμα για οργανισμούς που χρειάζονται επεξεργασία και ανάλυση μεγάλου όγκου δεδομένων ροής σε πραγματικό χρόνο. Ωστόσο, η πολυπλοκότητά του και οι έντονες απαιτήσεις πόρων μπορεί να αποτελούν εμπόδιο για ορισμένους οργανισμούς και μπορεί να μην είναι κατάλληλο για οργανισμούς που απαιτούν προηγμένες δυνατότητες ανάλυσης.

- Apache Flink
  Το Apache Flink είναι ένα κατανεμημένο υπολογιστικό σύστημα ανοιχτού κώδικα που έχει σχεδιαστεί για την επεξεργασία δεδομένων ροής και δεδομένων παρτίδας (batch data). Παρέχει μια πλατφόρμα για επεξεργασία δεδομένων σε πραγματικό χρόνο, επεξεργασία ροής και επεξεργασία κατά παρτίδες.

  Πως δουλεύει:
  Το Apache Flink επεξεργάζεται δεδομένα διαιρώντας τα σε μια σειρά παραλληλιζόμενων λειτουργιών, γνωστών ως γραφήματα ροής δεδομένων (dataflow graphs). Κάθε λειτουργία εκτελείται από μια εργασία που τρέχει σε έναν κόμβο εργασίας στο cluster. Οι διαχειριστές εργασιών (task managers) στο σύμπλεγμα (cluster) είναι υπεύθυνοι για τη διαχείριση των εργασιών και τον συντονισμό των ανταλλαγών δεδομένων μεταξύ τους. Τα δεδομένα υποβάλλονται σε επεξεργασία στη μνήμη και τα αποτελέσματα ενημερώνονται συνεχώς καθώς φτάνουν νέα δεδομένα.

  Το Apache Flink προσφέρει πολλά API για την επεξεργασία δεδομένων, συμπεριλαμβανομένου του DataStream API, το οποίο χρησιμοποιείται για επεξεργασία ροής, και του DataSet API, το οποίο χρησιμοποιείται για batch processing. Υποστηρίζει επίσης SQL queries και αλγόριθμους μηχανικής μάθησης.

  Πλεονεκτήματα:
  Υψηλή απόδοση: Το Apache Flink έχει σχεδιαστεί για την επεξεργασία μεγάλων ποσοτήτων δεδομένων με υψηλή ταχύτητα, καθιστώντας το μια αποτελεσματική πλατφόρμα για επεξεργασία δεδομένων σε πραγματικό χρόνο.

  Ευελιξία: Το Apache Flink υποστηρίζει ένα ευρύ φάσμα πηγών δεδομένων, συμπεριλαμβανομένων δεδομένων παρτίδας και δεδομένων ροής, και παρέχει API για επεξεργασία και ανάλυση δεδομένων.

  Ανοχή σφαλμάτων: Το Apache Flink έχει μια αρχιτεκτονική ανεκτική σε σφάλματα που διασφαλίζει ότι τα δεδομένα δεν θα χαθούν σε περίπτωση αποτυχίας.

  Ενσωμάτωση με άλλα εργαλεία: Το Apache Flink ενσωματώνεται με άλλα εργαλεία όπως το Apache Kafka, το Apache Hadoop και το Apache Spark, καθιστώντας εύκολη την ενσωμάτωσή του σε υπάρχουσες ροές εργασίας επεξεργασίας δεδομένων.

  Μειονεκτήματα:

  Καμπύλη μάθησης: Το Apache Flink έχει μια πιο απότομη καμπύλη μάθησης σε σύγκριση με άλλα εργαλεία επεξεργασίας δεδομένων λόγω της πολύπλοκης αρχιτεκτονικής του.

Εντατικό σε πόρους: Το Apache Flink μπορεί να είναι εντάσεως πόρων, απαιτώντας σημαντική ποσότητα μνήμης και επεξεργαστικής ισχύος για να λειτουργήσει αποτελεσματικά.

Περιορισμένη υποστήριξη: Ως σχετικά νέα τεχνολογία, το Apache Flink έχει περιορισμένη υποστήριξη σε σύγκριση με πιο καθιερωμένα εργαλεία επεξεργασίας δεδομένων όπως το Apache Spark.

Συνολικά, το Apache Flink είναι ένα ισχυρό και ευέλικτο κατανεμημένο υπολογιστικό σύστημα που παρέχει μια πλατφόρμα για επεξεργασία δεδομένων σε πραγματικό χρόνο, επεξεργασία ροής και επεξεργασία κατά παρτίδες. Η υψηλή απόδοση, η αρχιτεκτονική του με ανοχή σε σφάλματα και η ενσωμάτωση με άλλα εργαλεία το καθιστούν ελκυστική επιλογή για οργανισμούς που χρειάζονται επεξεργασία και ανάλυση μεγάλου όγκου δεδομένων σε πραγματικό χρόνο. Ωστόσο, η πολυπλοκότητά του και οι απαιτήσεις σε πόρους μπορεί να αποτελούν εμπόδιο για ορισμένους οργανισμούς και μπορεί να μην είναι κατάλληλο για οργανισμούς που χρειάζονται εκτεταμένη υποστήριξη ή έχουν περιορισμένους πόρους.

- Apache Storm
  Το Apache Storm είναι ένα ανοιχτού κώδικα, κατανεμημένο υπολογιστικό σύστημα που χρησιμοποιείται για την επεξεργασία μεγάλου όγκου δεδομένων ροής σε πραγματικό χρόνο. Παρέχει μια πλατφόρμα για την επεξεργασία συνεχών ροών δεδομένων και την εκτέλεση πολύπλοκων υπολογισμών σε αυτά.

  Πως δουλεύει:
  Το Apache Storm επεξεργάζεται δεδομένα διαιρώντας τα σε μικρές μονάδες που ονομάζονται πλειάδες (tuples). Οι πλειάδες δέχονται επεξεργασία από ένα κατευθυνόμενο ακυκλικό γράφημα (Directed Acyclic Graph - DAG) λειτουργιών, το οποίο ορίζεται από τον χρήστη. Το DAG εκτελείται σε ένα σύμπλεγμα κόμβων (nodes) και κάθε κόμβος μπορεί να επεξεργαστεί μία ή περισσότερες πλειάδες. Το Storm χρησιμοποιεί μια αρχιτεκτονική master-slave, με έναν κύριο κόμβο που συντονίζει την εκτέλεση του DAG και τους κόμβους εργαζόμενους που επεξεργάζονται τα δεδομένα. Ο κύριος κόμβος εκχωρεί τις πλειάδες στους κόμβους εργασίας, οι οποίοι τις επεξεργάζονται παράλληλα.

  Το Storm παρέχει δύο βασικές αφαιρέσεις για την επεξεργασία δεδομένων: spouts and bolts. Τα spouts είναι υπεύθυνα για την απορρόφηση δεδομένων από πηγές δεδομένων, ενώ τα bolts είναι υπεύθυνα για την επεξεργασία των δεδομένων. Το Storm παρέχει επίσης μια ποικιλία από ενσωματωμένα spouts and bolts, καθώς και υποστήριξη για spouts and bolts ουλόνια τρίτων.

  Πλεονεκτήματα:
  Υψηλή απόδοση: Το Apache Storm έχει σχεδιαστεί για την επεξεργασία μεγάλου όγκου δεδομένων ροής με υψηλή ταχύτητα, καθιστώντας το μια αποτελεσματική πλατφόρμα για επεξεργασία δεδομένων σε πραγματικό χρόνο.

  Ανοχή σφαλμάτων: Το Apache Storm έχει μια αρχιτεκτονική ανεκτική σε σφάλματα που διασφαλίζει ότι τα δεδομένα δεν θα χαθούν σε περίπτωση αποτυχίας.

Ευελιξία: Το Apache Storm παρέχει μια ευέλικτη πλατφόρμα για την επεξεργασία δεδομένων ροής, με υποστήριξη για μια ποικιλία πηγών δεδομένων και ενσωματώσεις τρίτων.

Ευκολία στη χρήση: Το Apache Storm είναι εύκολο στη χρήση και έχει χαμηλή καμπύλη εκμάθησης σε σύγκριση με άλλα εργαλεία επεξεργασίας δεδομένων.

Μειονεκτήματα:

Περιορισμένες δυνατότητες αναλυτικών στοιχείων: Το Apache Storm δεν προσφέρει προηγμένες δυνατότητες ανάλυσης, όπως μηχανική εκμάθηση ή οπτικοποίηση δεδομένων.

Εντατικό σε πόρους: Το Apache Storm απαιτεί σημαντική ποσότητα μνήμης και επεξεργαστικής ισχύος για να λειτουργήσει αποτελεσματικά.

Πολύπλοκη αρχιτεκτονική: Η αρχιτεκτονική του Apache Storm μπορεί να είναι περίπλοκη, γεγονός που μπορεί να δυσκολέψει ορισμένους χρήστες να κατανοήσουν και να χρησιμοποιήσουν αποτελεσματικά.

Συνολικά, το Apache Storm είναι ένα ισχυρό και ευέλικτο κατανεμημένο υπολογιστικό σύστημα σε πραγματικό χρόνο που παρέχει μια πλατφόρμα για την επεξεργασία μεγάλου όγκου δεδομένων ροής. Η υψηλή απόδοση, η ανεκτική σε σφάλματα αρχιτεκτονική και η ευκολία χρήσης το καθιστούν ελκυστική επιλογή για οργανισμούς που χρειάζονται επεξεργασία και ανάλυση μεγάλου όγκου δεδομένων σε πραγματικό χρόνο. Ωστόσο, οι περιορισμένες δυνατότητες ανάλυσης και οι απαιτήσεις πόρων του μπορεί να αποτελούν εμπόδιο για ορισμένους οργανισμούς και η πολύπλοκη αρχιτεκτονική του μπορεί να καταστήσει δύσκολη την αποτελεσματική χρήση ορισμένων χρηστών.

1.3.4 Τεχνικές διαμοιρασμού πόρων και παράλληλης επεξεργασίας δεδομένων

Η κοινή χρήση πόρων αναφέρεται στην πρακτική της διάθεσης υπολογιστικών πόρων, όπως η αποθήκευση, η ισχύς επεξεργασίας και το εύρος ζώνης δικτύου σε πολλούς χρήστες ή εφαρμογές. Αυτό μπορεί να βοηθήσει στη βελτιστοποίηση της χρήσης των πόρων, στη μείωση του κόστους και στη βελτίωση της αποτελεσματικότητας. Μερικά παραδείγματα τεχνικών κοινής χρήσης πόρων περιλαμβάνουν:

Virtualization: Αυτή η τεχνική περιλαμβάνει τη δημιουργία εικονικών εκδόσεων υπολογιστικών πόρων, όπως διακομιστές, συσκευές αποθήκευσης και δίκτυα, οι οποίες μπορούν να κοινοποιηθούν σε πολλούς χρήστες ή εφαρμογές.

Cloud computing: Περιλαμβάνει τη χρήση απομακρυσμένων διακομιστών, αποθήκευσης και άλλων πόρων που παρέχονται από τρίτο πάροχο για την εκτέλεση υπολογιστικών εργασιών.

Containerization: Αυτή η τεχνική περιλαμβάνει τη συσκευασία εφαρμογών λογισμικού σε κοντέινερ, τα οποία μπορούν εύκολα να αναπτύξουν και να διαχειριστούν οι developers, και μπορούν να εκτελεστούν σε οποιαδήποτε υποδομή.

Η παράλληλη επεξεργασία δεδομένων, από την άλλη πλευρά, αναφέρεται στην πρακτική της διαίρεσης ενός μεγάλου συνόλου δεδομένων σε μικρότερα κομμάτια και της ταυτόχρονης επεξεργασίας τους σε πολλαπλές μονάδες επεξεργασίας, όπως CPU, GPU ή συμπλέγματα υπολογιστών. Αυτό μπορεί να βοηθήσει στην επιτάχυνση της επεξεργασίας δεδομένων και στη βελτίωση της απόδοσης. Μερικά παραδείγματα τεχνικών παράλληλης επεξεργασίας δεδομένων περιλαμβάνουν:

MapReduce: Αυτό είναι ένα μοντέλο προγραμματισμού και ένα πλαίσιο λογισμικού για την επεξεργασία μεγάλων ποσοτήτων δεδομένων σε ένα κατανεμημένο υπολογιστικό περιβάλλον.

Spark: Αυτό είναι ένα κατανεμημένο υπολογιστικό σύστημα ανοιχτού κώδικα που έχει σχεδιαστεί για την επεξεργασία μεγάλων ποσοτήτων δεδομένων.

Hadoop: Αυτό είναι ένα κατανεμημένο υπολογιστικό πλαίσιο που επιτρέπει την αποθήκευση και την επεξεργασία μεγάλων συνόλων δεδομένων σε ομάδες υπολογιστών.

Στη συνέχεια θα αναλύσουμε περισσότερο τα δύο τελευταία:

- Apache Spark
  Το Apache Spark είναι ένα κατανεμημένο υπολογιστικό σύστημα ανοιχτού κώδικα που έχει σχεδιαστεί για την παράλληλη επεξεργασία μεγάλων ποσοτήτων δεδομένων σε ομάδες υπολογιστών. Αναπτύχθηκε στο AMPLab του UC Berkeley και τώρα συντηρείται από το Apache Software Foundation. Το Spark παρέχει μια διεπαφή για προγραμματισμό σε Java, Scala, Python και R.

  Πως δουλεύει:
  Το Spark επιτρέπει στους χρήστες να γράφουν εφαρμογές χρησιμοποιώντας ένα API υψηλού επιπέδου σε Java, Scala, Python και R. Το Spark υποστηρίζει πολλές πηγές δεδομένων, όπως το Hadoop Distributed File System (HDFS), το Apache Cassandra, το Apache HBase και το Amazon S3. Μπορεί να τρέξει σε αυτόνομη λειτουργία ή σε Apache Mesos, Hadoop YARN ή Kubernetes. Περιλαμβάνει έναν αριθμό ενσωματωμένων βιβλιοθηκών, όπως το Spark SQL για ερωτήματα που βασίζονται σε SQL, το Spark Streaming για επεξεργασία δεδομένων ροής σε πραγματικό χρόνο και το MLlib για μηχανική εκμάθηση.

  Το Spark χρησιμοποιεί μια κατανεμημένη αρχιτεκτονική υπολογιστών για την παράλληλη επεξεργασία δεδομένων σε ένα cluster μηχανών. Στον πυρήνα του Spark βρίσκεται το Resilient Distributed Dataset (RDD), το οποίο είναι μια αμετάβλητη κατανεμημένη συλλογή αντικειμένων. Τα RDD μπορούν να δημιουργηθούν από δεδομένα που είναι αποθηκευμένα στη μνήμη ή στο δίσκο και μπορούν να μετασχηματιστούν και να λειτουργήσουν παράλληλα σε ένα σύμπλεγμα μηχανών. Το Spark παρέχει έναν αριθμό λειτουργιών που μπορούν να εκτελεστούν σε RDD, συμπεριλαμβανομένων μετασχηματισμών και ενεργειών (όπως μέτρηση, συλλογή και μείωση).

  Πλεονεκτήματα:

  Ταχύτητα: Το Spark είναι γνωστό για την ταχύτητά του, καθώς μπορεί να εκτελέσει εργασίες επεξεργασίας δεδομένων πολύ πιο γρήγορα από τις παραδοσιακές εργασίες Hadoop MapReduce.

Ευκολία στη χρήση: Το API του Spark έχει σχεδιαστεί για να είναι εύκολο στη χρήση και να αφαιρεί την πολυπλοκότητα των κατανεμημένων υπολογιστών.
Ευελιξία: Το Spark υποστηρίζει μια ποικιλία πηγών δεδομένων και μπορεί να εκτελεστεί σε διάφορες πλατφόρμες, καθιστώντας το ένα ευέλικτο εργαλείο για την επεξεργασία δεδομένων.
Δυνατότητες μηχανικής μάθησης: Το Spark περιλαμβάνει MLlib, μια ενσωματωμένη βιβλιοθήκη για μηχανική μάθηση που παρέχει μια σειρά αλγορίθμων για ταξινόμηση, παλινδρόμηση και ομαδοποίηση.

Μειονεκτήματα:

Απαιτήσεις μνήμης: Το Spark απαιτεί πολλή μνήμη για την εκτέλεση επεξεργασίας στη μνήμη, κάτι που μπορεί να είναι πρόκληση για ορισμένες εφαρμογές.
Καμπύλη εκμάθησης: Αν και το Spark έχει σχεδιαστεί για να είναι εύκολο στη χρήση, εξακολουθεί να έχει καμπύλη μάθησης, ιδιαίτερα για όσους δεν είναι εξοικειωμένοι με τους κατανεμημένους υπολογιστές.
Εντοπισμός σφαλμάτων: Ο εντοπισμός σφαλμάτων μπορεί να είναι απαιτητικός, ιδιαίτερα όταν αντιμετωπίζουμε πολύπλοκους μετασχηματισμούς και ενέργειες σε RDD.

- Apache Hadoop
  Το Apache Hadoop είναι ένα κατανεμημένο υπολογιστικό πλαίσιο ανοιχτού κώδικα που έχει σχεδιαστεί για την αποθήκευση και την επεξεργασία μεγάλων ποσοτήτων δεδομένων σε συμπλέγματα υπολογιστών. Αναπτύχθηκε από τους Doug Cutting και Mike Cafarella το 2005 και τώρα συντηρείται από το Apache Software Foundation.

  Πώς δουλεύει:
  Το Hadoop αποτελείται από δύο κύρια στοιχεία: το κατανεμημένο σύστημα αρχείων Hadoop (HDFS) και το MapReduce. Το HDFS είναι ένα κατανεμημένο σύστημα αρχείων που επιτρέπει την αποθήκευση μεγάλων συνόλων δεδομένων σε πολλαπλά μηχανήματα, ενώ το MapReduce είναι ένα μοντέλο προγραμματισμού που χρησιμοποιείται για την επεξεργασία και την ανάλυση των δεδομένων.

  Όταν ένα αρχείο δεδομένων αποθηκεύεται σε HDFS, χωρίζεται σε μικρότερα κομμάτια που ονομάζονται "μπλοκ" (blocks) και διανέμεται σε πολλαπλά μηχανήματα στο cluster. Κάθε μπλοκ αναπαράγεται σε πολλά μηχανήματα για ανοχή σφαλμάτων. Οι εργασίες MapReduce μπορούν στη συνέχεια να εκτελεστούν σε όλα τα δεδομένα που είναι αποθηκευμένα στο HDFS διαιρώντας τα δεδομένα σε μικρότερα "splits". Κάθε split δέχεται παράλληλη επεξεργασία σε πολλαπλά μηχανήματα στο σύμπλεγμα.

  Πλεονεκτήματα:

  Επεκτασιμότητα: Το Hadoop είναι εξαιρετικά επεκτάσιμο και μπορεί να επεξεργαστεί petabyte δεδομένων σε clusters μηχανών.
  Ανοχή σφαλμάτων: Η κατανεμημένη αρχιτεκτονική του Hadoop το καθιστά εξαιρετικά ανεκτικό σε σφάλματα, καθώς τα δεδομένα αναπαράγονται σε πολλαπλά μηχανήματα στο cluster.

Οικονομικά: Το Hadoop εκτελείται σε υλικό βασικών προϊόντων, γεγονός που το καθιστά μια οικονομικά αποδοτική λύση για την αποθήκευση και την επεξεργασία μεγάλων ποσοτήτων δεδομένων.
Ευελιξία: Το Hadoop υποστηρίζει μια ποικιλία πηγών δεδομένων και μπορεί να ενσωματωθεί με πολλά άλλα εργαλεία και τεχνολογίες μεγάλων δεδομένων.

Μειονεκτήματα:

Πολυπλοκότητα: Το Hadoop έχει μια απότομη καμπύλη μάθησης και απαιτεί εξειδικευμένες δεξιότητες και γνώσεις για την εφαρμογή και τη διαχείριση.
Απόδοση: Η αρχιτεκτονική του Hadoop μπορεί να οδηγήσει σε πιο αργή απόδοση σε μικρότερα σύνολα δεδομένων, καθώς τα γενικά έξοδα διαχείρισης κατανεμημένων υπολογιστών μπορεί να αντισταθμίσουν τα οφέλη της παράλληλης επεξεργασίας.
Περιορισμένη επεξεργασία σε πραγματικό χρόνο: Το μοντέλο ομαδικής επεξεργασίας του Hadoop δεν είναι κατάλληλο για επεξεργασία δεδομένων σε πραγματικό χρόνο. Ωστόσο, υπάρχουν πλέον πρόσθετα εργαλεία και τεχνολογίες, όπως το Apache Spark και το Apache Flink, που μπορούν να χρησιμοποιηθούν σε συνδυασμό με το Hadoop για την παροχή δυνατοτήτων επεξεργασίας σε πραγματικό χρόνο.

1.3.5 Τεχνικές ενορχήστρωσης

Η ενορχήστρωση αναφέρεται γενικά στον συντονισμό και τη διαχείριση πολλαπλών στοιχείων ή συστημάτων για την επίτευξη ενός επιθυμητού αποτελέσματος. Στο πλαίσιο των υπολογιστών, η ενορχήστρωση αναφέρεται συγκεκριμένα στην αυτοματοποίηση και διαχείριση πολύπλοκων συστημάτων ή ροών εργασίας που περιλαμβάνουν πολλαπλές υπηρεσίες, εφαρμογές και πόρους υποδομής.

Στον τομέα της ανάπτυξης λογισμικού, η ενορχήστρωση χρησιμοποιείται συχνά στο πλαίσιο του υπολογιστικού νέφους και των κατανεμημένων συστημάτων, όπου περιλαμβάνει τη διαχείριση κοντέινερ, εικονικών μηχανών και άλλων πόρων που απαιτούνται για την ανάπτυξη και εκτέλεση εφαρμογών. Εργαλεία ενορχήστρωσης όπως το Kubernetes, το Docker Swarm και το Apache Mesos χρησιμοποιούνται συνήθως για την αυτοματοποίηση της ανάπτυξης, της κλιμάκωσης και της διαχείρισης containerized εφαρμογών σε μια κατανεμημένη υποδομή.

Συνοπτικά, η ενορχήστρωση είναι μια τεχνική που χρησιμοποιείται για την αυτοματοποίηση πολύπλοκων διαδικασιών, τη διαχείριση πολλαπλών στοιχείων και τη διασφάλιση ότι όλα τα μέρη ενός συστήματος συνεργάζονται άψογα για να επιτευχθεί το επιθυμητό αποτέλεσμα.

- Docker
  Το Docker (Solomon, et al., 2014) είναι μια πλατφόρμα ανοιχτού κώδικα που επιτρέπει στους προγραμματιστές να δημιουργούν, να αποστέλλουν και να εκτελούν εφαρμογές σε κοντέινερ. Κυκλοφόρησε για πρώτη φορά το 2013 και έκτοτε έχει γίνει ένα δημοφιλές εργαλείο μεταξύ προγραμματιστών, μηχανικών DevOps και διαχειριστών συστημάτων.

  Πως δουλεύει:

Το Docker χρησιμοποιεί τεχνολογία κοντέινερ για να συσκευάσει μια εφαρμογή και τις εξαρτήσεις της σε ένα κοντέινερ. Ένα κοντέινερ είναι ένα ελαφρύ, αυτόνομο εκτελέσιμο πακέτο που περιλαμβάνει όλα όσα χρειάζονται για την εκτέλεση της εφαρμογής, συμπεριλαμβανομένου του κώδικα, του χρόνου εκτέλεσης, των βιβλιοθηκών και των εργαλείων συστήματος.

Όταν δημιουργείται ένα κοντέινερ, το Docker δημιουργεί ένα επίπεδο πάνω από το λειτουργικό σύστημα κεντρικού υπολογιστή που παρέχει ένα απομονωμένο περιβάλλον για την εκτέλεση της εφαρμογής. Κάθε κοντέινερ είναι εντελώς ανεξάρτητο και δεν παρεμβαίνει σε άλλα κοντέινερ που εκτελούνται στον ίδιο κεντρικό υπολογιστή.

Το Docker μπορεί να χρησιμοποιηθεί για τη δημιουργία, την ανάπτυξη και τη διαχείριση κοντέινερ σε διάφορα περιβάλλοντα, συμπεριλαμβανομένων κέντρων δεδομένων εσωτερικής εγκατάστασης, πλατφορμών που βασίζονται σε cloud και υβριδικών περιβαλλόντων.

Συστατικά:

Τα κύρια στοιχεία του Docker είναι:

Μηχανή Docker: Το βασικό στοιχείο του Docker, υπεύθυνο για τη δημιουργία και τη διαχείριση κοντέινερ.
Μητρώο Docker: Ένα αποθετήριο για την αποθήκευση και την κοινή χρήση εικόνων κοντέινερ.
Docker CLI: Μια διεπαφή γραμμής εντολών για αλληλεπίδραση με το Docker.
Docker Compose: Ένα εργαλείο για τον καθορισμό και την εκτέλεση εφαρμογών Docker πολλαπλών κοντέινερ.

Πλεονεκτήματα:

Φορητότητα: Το Docker επιτρέπει στους προγραμματιστές να συσκευάσουν μια εφαρμογή και τις εξαρτήσεις της σε ένα κοντέινερ, καθιστώντας εύκολη την ανάπτυξη και εκτέλεση της εφαρμογής σε οποιαδήποτε πλατφόρμα που υποστηρίζει το Docker.

Επεκτασιμότητα: Διευκολύνει την οριζόντια κλίμακα μιας εφαρμογής προσθέτοντας περισσότερα κοντέινερ, επιτρέποντας στους προγραμματιστές να χειρίζονται αυξημένη επισκεψιμότητα ή φόρτο εργασίας.

Αποδοτικότητα πόρων: Τα κοντέινερ Docker είναι ελαφριά και μοιράζονται τους πόρους του κεντρικού λειτουργικού συστήματος, καθιστώντας τα πιο αποτελεσματικά από τις εικονικές μηχανές.

Συνέπεια: Το Docker διασφαλίζει ότι η εφαρμογή εκτελείται με συνέπεια σε διαφορετικά περιβάλλοντα, εξαλείφοντας το πρόβλημα "λειτουργεί στον υπολογιστή μου".

Ασφάλεια: Παρέχει ένα ασφαλές περιβάλλον χρόνου εκτέλεσης για την εφαρμογή απομονώνοντάς την από άλλες διεργασίες που εκτελούνται στον ίδιο κεντρικό υπολογιστή.

Μειονεκτήματα:

Πολυπλοκότητα: Το Docker μπορεί να είναι πολύπλοκο στη ρύθμιση και τη διαμόρφωση, ειδικά για ομάδες που είναι καινούριες στη δημιουργία κοντέινερ.

Καμπύλη μάθησης: Οι προγραμματιστές μπορεί να χρειαστεί να μάθουν νέα εργαλεία και έννοιες όταν υιοθετούν το Docker, όπως η ενορχήστρωση κοντέινερ και η δικτύωση.

Overhead: Το Docker προσθέτει κάποια επιβάρυνση στην εφαρμογή, καθώς κάθε κοντέινερ απαιτεί το δικό του περιβάλλον χρόνου εκτέλεσης και βιβλιοθήκες συστήματος.

Συμβατότητα: Ορισμένες εφαρμογές παλαιού τύπου ενδέχεται να μην είναι συμβατές με το Docker, απαιτώντας από τους προγραμματιστές να τροποποιήσουν τον κώδικα ή τις διαμορφώσεις της εφαρμογής.

Συνολικά, το Docker είναι ένα ισχυρό εργαλείο για τη δημιουργία και τη διαχείριση σύγχρονων εφαρμογών. Η τεχνολογία μεταφοράς εμπορευματοκιβωτίων, η φορητότητα και η επεκτασιμότητα το καθιστούν απαραίτητο εργαλείο για την ανάπτυξη και ανάπτυξη σύγχρονων εφαρμογών, αλλά απαιτεί κάποια επιβάρυνση μάθησης και εγκατάστασης.

- Kubernetes
  Το Kubernetes (Burns, et al., 2013) είναι μια πλατφόρμα ενορχήστρωσης κοντέινερ ανοιχτού κώδικα που αυτοματοποιεί την ανάπτυξη, την κλιμάκωση και τη διαχείριση εφαρμογών με κοντέινερ. Αναπτύχθηκε από την Google και τώρα συντηρείται από το Cloud Native Computing Foundation (CNCF).

  Πως δουλεύει:

  Το Kubernetes διαχειρίζεται ένα σύμπλεγμα κόμβων που εκτελούν εφαρμογές με κοντέινερ. Κάθε κόμβος εκτελεί έναν χρόνο εκτέλεσης κοντέινερ, όπως το Docker ή το CRI-O, και επικοινωνεί με το επίπεδο ελέγχου Kubernetes για τη διαχείριση των κοντέινερ και των σχετικών πόρων τους.

  Το Kubernetes χρησιμοποιεί ένα δηλωτικό μοντέλο διαμόρφωσης, όπου η επιθυμητή κατάσταση της εφαρμογής ορίζεται σε ένα αρχείο YAML ή JSON. Το επίπεδο ελέγχου παρακολουθεί συνεχώς το σύμπλεγμα και συμβιβάζει τυχόν διαφορές μεταξύ της επιθυμητής κατάστασης και της τρέχουσας κατάστασης της εφαρμογής.

  Το Kubernetes παρέχει ένα ευρύ φάσμα δυνατοτήτων για τη διαχείριση κοντέινερ, όπως εξισορρόπηση φορτίου, ανακάλυψη υπηρεσίας, αυτόματη κλιμάκωση, κυλιόμενες ενημερώσεις και αυτο-ίαση.

  Συστατικά:

Τα κύρια συστατικά του Kubernetes είναι:

Επίπεδο ελέγχου: Το επίπεδο ελέγχου διαχειρίζεται το σύμπλεγμα και τα στοιχεία του, συμπεριλαμβανομένου του διακομιστή API, etcd, kube-scheduler, kube-controller-manager και cloud-controller-manager.

Κόμβοι: Οι κόμβοι είναι μηχανές εργασίας που λειτουργούν κοντέινερ και επικοινωνούν με το επίπεδο ελέγχου.

Χρόνος εκτέλεσης κοντέινερ: Ο χρόνος εκτέλεσης κοντέινερ, όπως το Docker ή το CRI-O, είναι υπεύθυνος για τη λειτουργία κοντέινερ στους κόμβους.

Pods: Τα Pods είναι η μικρότερη μονάδα ανάπτυξης στο Kubernetes και αποτελούνται από ένα ή περισσότερα κοντέινερ που μοιράζονται τον ίδιο χώρο ονομάτων δικτύου και όγκους αποθήκευσης.

Υπηρεσίες: Οι υπηρεσίες παρέχουν μια σταθερή διεύθυνση IP και όνομα DNS για ένα σύνολο ομάδων ομάδων, επιτρέποντας στους πελάτες να έχουν πρόσβαση στην εφαρμογή χωρίς να γνωρίζουν το συγκεκριμένο pod που εκτελεί το κοντέινερ.

Υπέρ και κατά:

Πλεονεκτήματα:

Επεκτασιμότητα: Το Kubernetes διευκολύνει την οριζόντια κλίμακα μιας εφαρμογής προσθέτοντας περισσότερους κόμβους ή κοντέινερ.

Ανθεκτικότητα: Το Kubernetes παρέχει δυνατότητες αυτο-ίασης, επανεκκινώντας αυτόματα τα κοντέινερ που αποτυγχάνουν και ανακατανέμοντας τον φόρτο εργασίας σε υγιείς κόμβους.

Ευελιξία: Το Kubernetes υποστηρίζει ένα ευρύ φάσμα χρόνων εκτέλεσης κοντέινερ και παρέχει ένα συνεπές API για τη διαχείριση κοντέινερ και των σχετικών πόρων τους.

Φορητότητα: Το Kubernetes λειτουργεί σε ένα ευρύ φάσμα πλατφορμών, συμπεριλαμβανομένων κέντρων δεδομένων εσωτερικής εγκατάστασης, πλατφορμών που βασίζονται σε cloud και υβριδικών περιβαλλόντων.

Οικοσύστημα: Το Kubernetes διαθέτει ένα μεγάλο και ενεργό οικοσύστημα εργαλείων, βιβλιοθηκών και υπηρεσιών που καθιστούν εύκολη την ενσωμάτωση με άλλα συστήματα και υπηρεσίες.

Μειονεκτήματα:

Πολυπλοκότητα: Το Kubernetes μπορεί να είναι πολύπλοκο στη ρύθμιση και τη διαμόρφωση, ειδικά για ομάδες που είναι νέες στην ενορχήστρωση κοντέινερ.

Καμπύλη μάθησης: Οι προγραμματιστές μπορεί να χρειαστεί να μάθουν νέα εργαλεία και έννοιες κατά την υιοθέτηση του Kubernetes, όπως αρχεία διαμόρφωσης YAML και δικτύωση ειδικά για το Kubernetes.

Overhead: Το Kubernetes προσθέτει κάποια επιβάρυνση στην εφαρμογή, καθώς κάθε κόμβος απαιτεί το δικό του περιβάλλον χρόνου εκτέλεσης και βιβλιοθήκες συστήματος.

Απαιτήσεις πόρων: Το Kubernetes απαιτεί έναν ελάχιστο αριθμό κόμβων για να λειτουργήσει αποτελεσματικά, γεγονός που μπορεί να αυξήσει το κόστος υποδομής.

Συνολικά, το Kubernetes είναι ένα ισχυρό εργαλείο για τη διαχείριση εφαρμογών σε κοντέινερ σε κλίμακα. Η επεκτασιμότητα, η ανθεκτικότητα και η ευελιξία του το καθιστούν απαραίτητο εργαλείο για την ανάπτυξη και ανάπτυξη σύγχρονων εφαρμογών, αλλά απαιτεί κάποια επιβάρυνση μάθησης και εγκατάστασης.

Ανίχνευση ανώμαλων τιμών

Η ανίχνευση ανωμαλιών είναι μια τεχνική που χρησιμοποιείται για τον εντοπισμό ασυνήθιστων μοτίβων ή παρατηρήσεων σε ένα σύνολο δεδομένων που δεν συμμορφώνονται με μια αναμενόμενη συμπεριφορά. Αυτό μπορεί να είναι χρήσιμο για τον εντοπισμό απάτης, τον εντοπισμό ελαττωματικού εξοπλισμού ή τον εντοπισμό άλλων τύπων ακραίων στοιχείων ή ασυνήθιστων συμβάντων σε ένα σύνολο δεδομένων. Οι αλγόριθμοι ανίχνευσης ανωμαλιών χωρίζονται σε τρεις μεγάλες κατηγορίες: χωρίς επίβλεψη (unsupervised), που σημαίνει ότι μαθαίνουν την κανονική συμπεριφορά ενός συνόλου δεδομένων χωρίς κανένα επισημασμένο παράδειγμα ανωμαλιών, με επίβλεψη (supervised), που σημαίνει ότι εκπαιδεύονται σε επισημασμένα παραδείγματα κανονικής και ανώμαλης συμπεριφοράς και με ημιεπίβλεψη (semi-supervised) κατά την οποία χρησιμοποιείται ένας συνδυασμός επισημασμένων και μη-επισημασμένων δεδομένων.

2.1 Ανώμαλες τιμές

Γενικά, μια ανωμαλία είναι ένα αποτέλεσμα ή μια τιμή που αποκλίνει από το αναμενόμενο. Στο πλαίσιο της ανίχνευσης ανωμαλιών, μια ανωμαλία αναφέρεται σε μοτίβα ή παρατηρήσεις σε ένα σύνολο δεδομένων που δεν συμμορφώνονται με την κανονική ή αναμενόμενη συμπεριφορά. Αυτές οι ανωμαλίες μπορεί να προκληθούν από διάφορους παράγοντες, όπως σφάλματα στη συλλογή ή επεξεργασία δεδομένων, ακραίες τιμές ή σπάνια συμβάντα. Τα ακριβή κριτήρια για το τι συνιστά μία ανωμαλία μπορεί να διαφέρουν από περίπτωση σε περίπτωση. Μπορούμε να πούμε ότι μια τιμή είναι μη φυσιολογική είτε μέσω ενός επαγωγικού συλλογισμού, επειδή διαφέρει δηλαδή από τις φυσιολογικές τιμές, είτε χρησιμοποιώντας πιθανότητες. Αν μία τιμή εμφανίζεται με μία πιθανότητα μικρότερη από το όριο, δηλαδή από τη μικρότερη πιθανότητα εμφάνισης των κανονικών τιμών, τότε μπορούμε να τη χαρακτηρίσουμε μη φυσιολογική.

Μπορούμε να διακρίνουμε 3 κατηγορίες ανωμαλιών:

- Ανωμαλία σημείου (Data point-based anomaly)
  Για να εξηγήσουμε καλύτερα τι είναι μια ανώμαλη τιμή σημείου πρέπει να καταλάβουμε τι είναι μία ακραία τιμή, δηλαδή ένα outlier. (Aggarwal, 2017) Ακραίες τιμές είναι όλες οι τιμές τις οποίες αναμένουμε να δούμε σε ένα σύνολο δεδομένων και προκαλούνται είτε από αναπόφευκτα τυχαία σφάλματα είτε από συστηματικά σφάλματα στη δειγματοληψία δεδομένων. Data point-based anomalies είναι οι ακραίες τιμές ή άλλες γενικά τιμές τις οποίες δεν αναμένουμε να υπάρχουν. Για παράδειγμα, μια μεμονωμένη συναλλαγή με πολύ μεγάλο ποσό σε σύγκριση με άλλες συναλλαγές σε ένα σύνολο δεδομένων χρηματοοικονομικών συναλλαγών.
- Συμφραζόμενη ανωμαλία (Context-based anomaly)
  Είναι η ανώμαλη τιμή, η οποία ενώ στην αρχή μας φαίνεται φυσιολογική, αν τη μελετήσουμε σε συγκεκριμένο πλαίσιο, είναι μη φυσιολογική. Για παράδειγμα, μια ένδειξη υψηλής θερμοκρασίας σε μια ζεστή καλοκαιρινή μέρα δεν θα θεωρηθεί ανωμαλία, αλλά η ίδια θερμοκρασία σε μια κρύα χειμωνιάτικη μέρα.
- Pattern-based anomalies
  Είναι οι ανώμαλες τιμές που προκύπτουν όταν παρουσιάζεται απόκλιση από ένα συστηματικό ιστορικό μοτίβο. Ένα συνηθισμένο παράδειγμα μιας ανωμαλίας που βασίζεται σε μοτίβα είναι ο εντοπισμός δόλιων συναλλαγών με πιστωτικές κάρτες. Ο αλγόριθμος εκπαιδεύεται σε ένα σύνολο δεδομένων κανονικών, μη δόλιων συναλλαγών για να μάθει τα τυπικά πρότυπα συμπεριφοράς δαπανών. Αφού εκπαιδευτεί, ο αλγόριθμος μπορεί στη συνέχεια να χρησιμοποιηθεί για τον

εντοπισμό τυχόν συναλλαγών που αποκλίνουν από αυτά τα κανονικά πρότυπα, όπως μια συναλλαγή που πραγματοποιείται σε τοποθεσία που δεν συνάδει με τις συνήθειες δαπανών του κατόχου της κάρτας .

2.2 Εντοπισμός ανώμαλων τιμών

Το ερώτημα που προκύπτει είναι πώς εντοπίζουμε ανώμαλες τιμές με συστηματικό τρόπο, δηλαδή μέσω προηγμένων αλγορίθμων που κατηγοριοποιούν κάποιες τιμές ή κάποια μοτίβα ώς ανώμαλα.

Υπάρχουν αρκετοί συστηματικοί τρόποι ανίχνευσης ανωμαλιών, όπως (Zhou, et al., 2018):

1. Στατιστικές μέθοδοι: Αυτές οι μέθοδοι χρησιμοποιούν στατιστικές τεχνικές για τον εντοπισμό ανωμαλιών στα δεδομένα. Για παράδειγμα, το Z-score και η απόσταση Mahalanobis χρησιμοποιούνται συνήθως για τον εντοπισμό σημείων δεδομένων που αποκλίνουν σημαντικά από τον μέσο όρο του συνόλου δεδομένων.
2. Ομαδοποίηση (clustering) : Αυτές οι μέθοδοι ομαδοποιούν παρόμοια σημεία δεδομένων και προσδιορίζουν σημεία δεδομένων που δεν ανήκουν σε κανένα σύμπλεγμα ως ανωμαλίες. Για παράδειγμα, το k-means και το DBSCAN είναι δημοφιλείς αλγόριθμοι ομαδοποίησης.
3. Ανάλυση χρονοσειρών: Αυτές οι μέθοδοι χρησιμοποιούν τεχνικές όπως αποσύνθεση, φιλτράρισμα και εξομάλυνση για τον εντοπισμό μη φυσιολογικών μοτίβων στα δεδομένα χρονοσειρών, όπως μια ασυνήθιστη άνοδος στις τιμές των μετοχών ή μια ξαφνική πτώση στην επισκεψιμότητα ενός ιστότοπου.
4. Μηχανική εκμάθηση (Machine Learning): Αυτές οι μέθοδοι χρησιμοποιούν εποπτευόμενους ή μη εποπτευόμενους αλγόριθμους μηχανικής μάθησης για να μάθουν τα κανονικά μοτίβα στα δεδομένα και να προσδιορίσουν τυχόν σημεία δεδομένων που αποκλίνουν από αυτά τα μοτίβα ως ανωμαλίες. Παραδείγματα τέτοιων αλγορίθμων είναι One-class SVM, Isolation Forest και άλλα.
5. Βαθιά μάθηση (Deep Learning): Αυτές οι μέθοδοι χρησιμοποιούν βαθιά νευρωνικά δίκτυα για να μάθουν τα κανονικά μοτίβα στα δεδομένα και να προσδιορίσουν τυχόν σημεία δεδομένων που αποκλίνουν από αυτά τα μοτίβα ως ανωμαλίες. Ο αυτόματος κωδικοποιητής (autoencoder) και ο αυτόματος κωδικοποιητής παραλλαγών (variational autoencoder) είναι παραδείγματα μοντέλων βαθιάς εκμάθησης που μπορούν να χρησιμοποιηθούν για τον εντοπισμό ανωμαλιών.
6. Υβριδικές μέθοδοι: Αυτές οι μέθοδοι συνδυάζουν τεχνικές από πολλαπλές μεθόδους για τη βελτίωση της ανίχνευσης ανωμαλιών. Για παράδειγμα, μια υβριδική μέθοδος μπορεί να χρησιμοποιεί έναν συνδυασμό στατιστικών μεθόδων, ομαδοποίησης και μηχανικής μάθησης για τον εντοπισμό ανωμαλιών στα δεδομένα.

Είναι σημαντικό να σημειωθεί ότι η επιλογή της μεθόδου θα εξαρτηθεί από το συγκεκριμένο πρόβλημα και το σύνολο δεδομένων που αναλύονται και διαφορετικές μέθοδοι μπορεί να έχουν διαφορετικά επίπεδα υπολογιστικής πολυπλοκότητας και απόδοσης.

Στην παρούσα εργασία από τις παραπάνω μεθόδους θα σχοληθούμε τόσο με την τέταρη και την πέμπτη, δηλαδή με το πώς εντοπίζουμε ανώμαλες τιμές μέσω της μηχανικής και της βαθιάς μάθησης, όσο και με την πρλωτη, αναλύοντας πολύ χρήσιμες στατιστικές μεθόδους. (Brownlee, 2018) Πριν εντρυφήσουμε σε αυτές όμως ας αναφέρουμε την ανίχνευση ακραίων τιμών, την αφαίρεση θορύβου

και την ανίχνευση καινοτομιών (novelty detection), οι οποίες είναι άμεσα συνδεδεμένές με την ανίχνευση ανώμαλων τιμών.

Ανίχνευση ακραίων τιμών είναι η διαδικασία αναγνώρισης τιμών ή παρατηρήσεων σε ένα σύνολο δεδομένων που αποκλίνουν σημαντικά από τις άλλες παρατηρήσεις. Αυτά τα σημεία δεδομένων αναφέρονται συνήθως ως "ακραία σημεία" επειδή είναι "εκτός γραμμής" με τα υπόλοιπα δεδομένα.

Αφαίρεση θορύβου είναι διαδικασία αφαίρεσης ή φιλτραρίσματος τυχαίων σφαλμάτων ή παραλλαγών στα δεδομένα που δεν αντιπροσωπεύουν σημαντικές πληροφορίες. Αυτό μπορεί να γίνει χρησιμοποιώντας τεχνικές όπως το διάμεσο φιλτράρισμα, το μέσο φιλτράρισμα και το φιλτράρισμα Gauss. Ο στόχος της αφαίρεσης θορύβου είναι η βελτίωση της ποιότητας των δεδομένων αφαιρώντας άσχετες ή παραπλανητικές πληροφορίες.

Ανίχνευση καινοτομίας ονομάζεται η διαδικασία εντοπισμού νέων ή προηγουμένως μη εμφανισμένων μοτίβων στα δεδομένα που διαφέρουν από τα μοτίβα στα οποία έχει εκπαιδευτεί το μοντέλο. Χρησιμοποιείται σε πεδία όπως η ανίχνευση απάτης, η ανίχνευση εισβολών και ο εντοπισμός νέων άγνωστων τύπων κυκλοφορίας δικτύου.

Είναι σημαντικό να σημειωθεί ότι υπάρχουν τρία γενικά "στυλ" ανίχνευσης ανωμαλιών. Αυτά είναι:

- Ανίχνευση ανωμαλιών με επίβλεψη

Η ανίχνευση ανωμαλιών με επίβλεψη είναι ένας τύπος ανίχνευσης ανωμαλιών που χρησιμοποιεί δεδομένα με ετικέτα (labeled) για να εκπαιδεύσει ένα μοντέλο ώστε να αναγνωρίζει ανώμαλα μοτίβα ή παρατηρήσεις σε νέα, χωρίς ετικέτα (unlabeled) δεδομένα. Το μοντέλο εκπαιδεύεται σε ένα σύνολο δεδομένων που περιλαμβάνει τόσο κανονικά όσο και ανώμαλα παραδείγματα και χρησιμοποιεί αυτά τα δεδομένα εκπαίδευσης για να μάθει τα κανονικά πρότυπα και τη συμπεριφορά των δεδομένων. Αφού εκπαιδευτεί, το μοντέλο μπορεί στη συνέχεια να χρησιμοποιηθεί για τον εντοπισμό τυχόν νέων σημείων δεδομένων που αποκλίνουν από αυτά τα κανονικά πρότυπα ως ανωμαλίες.

Στην εποπτευόμενη ανίχνευση ανωμαλιών, τα ανώμαλα παραδείγματα αναφέρονται συνήθως ως "θετικά" παραδείγματα, ενώ τα κανονικά παραδείγματα αναφέρονται ως "αρνητικά" παραδείγματα. Το μοντέλο εκπαιδεύεται ώστε να μεγιστοποιεί την ικανότητα διαχωρισμού των θετικών παραδειγμάτων από τα αρνητικά παραδείγματα με τη χρήση κάποιου αλγόριθμου δυαδικής ταξινόμησης όπως SVM, Logistic Regression ή Random Forest. Η ανίχνευση ανωμαλιών με επίβλεψη είναι χρήσιμη όταν υπάρχει επαρκής ποσότητα επισημασμένων δεδομένων διαθέσιμα και τα ανώμαλα μοτίβα είναι γνωστά εκ των προτέρων.

- Ανίχνευση ανωμαλιών με ημιεπίβλεψη

Η ανίχνευση ανωμαλιών με ημιεπίβλεψη είναι ένας τύπος ανίχνευσης ανωμαλιών που χρησιμοποιεί έναν συνδυασμό δεδομένων με ετικέτα και χωρίς ετικέτα για την εκπαίδευση ενός μοντέλου. Είναι παρόμοια με την εποπτευόμενη ανίχνευση ανωμαλιών, καθώς χρησιμοποιεί δεδομένα με ετικέτα για να εκπαιδεύσει το μοντέλο να εντοπίζει ανώμαλα μοτίβα ή παρατηρήσεις σε νέα, χωρίς ετικέτα δεδομένα. Ωστόσο, σε αντίθεση με την εποπτευόμενη ανίχνευση ανωμαλιών, η οποία απαιτεί επαρκή ποσότητα δεδομένων με ετικέτα, η ημι-εποπτευόμενη ανίχνευση ανωμαλιών μπορεί να λειτουργήσει με περιορισμένο αριθμό δεδομένων με ετικέτα.

Ο αλγόριθμος εκπαιδεύεται σε ένα μικρό σύνολο δεδομένων με ετικέτα (θετικά και αρνητικά παραδείγματα) και στη συνέχεια χρησιμοποιεί αυτή τη γνώση για να ταξινομήσει τα υπόλοιπα δεδομένα χωρίς ετικέτα. Ο αλγόριθμος μπορεί επίσης να χρησιμοποιήσει τεχνικές χωρίς επίβλεψη για να μάθει τα κανονικά μοτίβα από τα δεδομένα χωρίς ετικέτα και να χρησιμοποιήσει αυτή τη γνώση για να εντοπίσει τυχόν νέα σημεία δεδομένων που αποκλίνουν από αυτά τα μοτίβα ως ανωμαλίες.

Η ημι-εποπτευόμενη ανίχνευση ανωμαλιών μπορεί να είναι χρήσιμη όταν υπάρχει περιορισμένος αριθμός διαθέσιμων δεδομένων με ετικέτα, αλλά μεγάλος αριθμός δεδομένων χωρίς ετικέτα. Συχνά χρησιμοποιείται σε τομείς όπως η ανίχνευση απάτης, η ανίχνευση εισβολής και η ανάλυση ιατρικής απεικόνισης. Είναι επίσης χρήσιμη όταν τα ανώμαλα μοτίβα δεν είναι γνωστά εκ των προτέρων, αλλά ο αλγόριθμος μπορεί να μάθει από τα δεδομένα και να προσαρμοστεί στις αλλαγές.

- Ανίχνευση ανωμαλιών χωρίς επίβλεψη

Η ανίχνευση ανωμαλιών χωρίς επίβλεψη είναι ένας τύπος ανίχνευσης ανωμαλιών που χρησιμοποιεί δεδομένα χωρίς ετικέτα για να εκπαιδεύσει ένα μοντέλο ώστε να αναγνωρίζει ανώμαλα μοτίβα ή παρατηρήσεις σε νέα δεδομένα. Σε αντίθεση με την εποπτευόμενη ανίχνευση ανωμαλιών, η οποία απαιτεί δεδομένα με ετικέτα για την εκπαίδευση του μοντέλου, η ανίχνευση ανωμαλίας χωρίς επίβλεψη δεν βασίζεται σε δεδομένα με ετικέτα και, αντίθετα, μαθαίνει τα κανονικά πρότυπα και τη συμπεριφορά των δεδομένων από τα ίδια τα δεδομένα. Αφού εκπαιδευτεί, το μοντέλο μπορεί στη συνέχεια να χρησιμοποιηθεί για τον εντοπισμό τυχόν νέων σημείων δεδομένων που αποκλίνουν από αυτά τα κανονικά πρότυπα ως ανωμαλίες.

Η ανίχνευση ανωμαλιών χωρίς επίβλεψη μπορεί να είναι χρήσιμη όταν τα δεδομένα με ετικέτα δεν είναι διαθέσιμα ή όταν τα ανώμαλα μοτίβα δεν είναι γνωστά εκ των προτέρων. Χρησιμοποιείται συχνά σε πεδία όπως η ανίχνευση εισβολών στο δίκτυο, ο εντοπισμός απάτης και ο εντοπισμός σπάνιων γεγονότων σε δεδομένα χρονοσειρών.

2.3 Βασικές γνώσεις για την ανάλυση δεδομένων

Πριν προχωρήσουμε στην αναφορά και επεξήγηση ορισμένων μεθόδων για την ανίχνεσυη ανώμαλων τιμών (μέσω της μηχανικής και της βαθιάς μάθησης), είναι σημαντικό να αναφερθούμε σε κάποια βασικά προαπαιτούμενα.

Αληθινά θετικά (TP – true positives) είναι οι περιπτώσεις στις οποίες μια συνθήκη είναι αληθής και η πρόβλεψη είναι κι αυτή αληθής.

Τα ψευδώς θετικά (FP – false positives) είναι οι περιπτώσεις στις οποίες μια συνθήκη είναι ψευδής και η πρόβλεψη είναι αληθής.

Αληθινά αρνητικά (TN – true negatives) είναι οι περιπτώσεις στις οποίες μια συνθήκη είναι ψευδής και η πρόβλεψη είναι κι αυτή ψευδής.

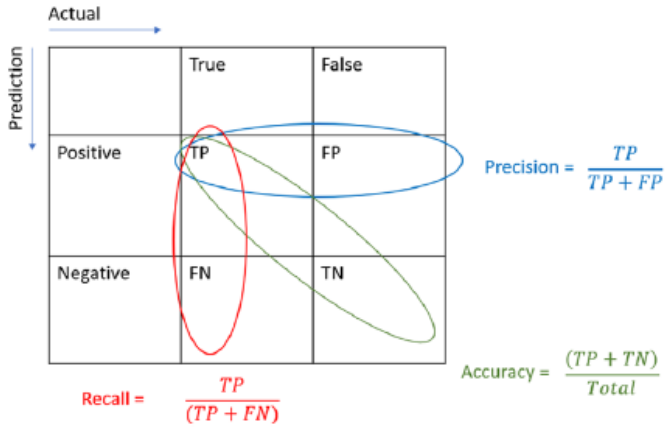
Τα ψευδώς αρνητικά (FN – false negatives) είναι οι περιπτώσεις στις οποίες μια συνθήκη είναι αληθής και η πρόβλεψη είναι ψευδής.

Ένας πίνακας σύγχυσης (confusion matrix) είναι ένας πίνακας που χρησιμοποιείται για τον καθορισμό της απόδοσης ενός αλγορίθμου ταξινόμησης. Είναι ένας πίνακας με δύο σειρές και δύο στήλες που αναφέρει τον αριθμό των αληθινών θετικών, των ψευδώς θετικών, των αληθινών αρνητικών και των ψευδώς αρνητικών.

Η ακρίβεια (precision) είναι η αναλογία των αληθινών θετικών αποτελεσμάτων μεταξύ όλων των θετικών αποτελεσμάτων. Υπολογίζεται ως αληθινά θετικά / (αληθινά θετικά + ψευδώς θετικά).

Η ακρίβεια (accuracy) είναι η αναλογία των σωστών προβλέψεων μεταξύ όλων των προβλέψεων που έγιναν. Υπολογίζεται ως (αληθινά θετικά + αληθινά αρνητικά) / συνολικές παρατηρήσεις.

Η ανάκληση (recall) επίσης γνωστή ως ευαισθησία ή πραγματικό θετικό ποσοστό, είναι η αναλογία των αληθινών θετικών μεταξύ όλων των πραγματικών θετικών. Υπολογίζεται ως αληθινά θετικά / (αληθινά θετικά + ψευδώς αρνητικά).



Εικόνα 2-1 Precision, Accuracy, Recall

Το ROC (Receiver Operating Characteristic) είναι μια γραφική γραφική παράσταση που απεικονίζει τη διαγνωστική ικανότητα ενός δυαδικού συστήματος ταξινομητή καθώς το όριο διάκρισής του ποικίλλει. Δημιουργείται σχεδιάζοντας το πραγματικό θετικό ποσοστό (TPR) έναντι του ψευδώς θετικού ποσοστού (FPR) σε διάφορες ρυθμίσεις κατωφλίου. Το αληθές θετικό ποσοστό (TPR) είναι επίσης γνωστό ως ευαισθησία, ανάκληση ή πιθανότητα ανίχνευσης (recall) και το ψευδώς θετικό ποσοστό (FPR) είναι επίσης γνωστό ως πτώση ή πιθανότητα ψευδούς συναγερμού (fall-out).

Εικόνα 2-2 Καμπύλη ROC με AUC

Η AUC (Area Under the ROC Curve - Περιοχή κάτω από την καμπύλη ROC) είναι ένα μέτρο του πόσο καλά ένας δυαδικός ταξινομητής μπορεί να διακρίνει μεταξύ θετικών και αρνητικών κλάσεων. Είναι μια τιμή μεταξύ 0 και 1, όπου το 1 αντιπροσωπεύει έναν τέλειο ταξινομητή και το 0,5 αντιπροσωπεύει έναν

ταξινομητή χωρίς αξία. Η AUC παρέχει ένα συνολικό μέτρο απόδοσης σε όλα τα πιθανά όρια ταξινόμησης (classification thresholds). Ένας τρόπος ερμηνείας της AUC είναι η πιθανότητα το μοντέλο να κατατάξει ένα τυχαίο θετικό παράδειγμα υψηλότερα από ένα τυχαίο αρνητικό παράδειγμα.

2.3 Στατιστικές Μέθοδοι Ανίχνευσης Ανωμαλιών

2.3.1 Έλεγχος Υποθέσεων (Hypothesis Testing)

Ο έλεγχος υποθέσεων είναι μια στατιστική μέθοδος που χρησιμοποιείται για να προσδιοριστεί εάν μια προτεινόμενη υπόθεση σχετικά με έναν πληθυσμό υποστηρίζεται από τα δεδομένα ή όχι. Περιλαμβάνει μια υπόθεση για τον πληθυσμό, τη συλλογή δεδομένων και την ανάλυση των δεδομένων για να προσδιοριστεί η πιθανότητα ότι η υπόθεση είναι αληθινή.

Η βασική διαδικασία του ελέγχου υποθέσεων περιλαμβάνει τα ακόλουθα βήματα:

1. Διατύπωση της υπόθεσης: Το πρώτο βήμα στον έλεγχο της υπόθεσης είναι να δηλωθεί με σαφήνεια η μηδενική υπόθεση (null hypothesis) και η εναλλακτική υπόθεση (alternative hypothesis). Η μηδενική υπόθεση είναι η υπόθεση που ελέγχεται, ενώ η εναλλακτική υπόθεση είναι το αντίθετο της μηδενικής υπόθεσης.
2. Ορσιμός επιπέδου σημαντικότητας: Το επίπεδο σημαντικότητας, που συμβολίζεται με άλφα (α), είναι η πιθανότητα απόρριψης της μηδενικής υπόθεσης όταν είναι πραγματικά αληθής. Συνήθως ορίζεται σε 0,05 ή 0,01.
3. Συλλογή δεδομένων: Το επόμενο βήμα είναι να συλλέξετε τα δεδομένα και να τα οργανώσετε με τρόπο που να είναι κατάλληλος για ανάλυση.
4. Διεξαγωγή της δοκιμής: Η στατιστική δοκιμής υπολογίζεται και συγκρίνεται με μια κρίσιμη τιμή από την κατάλληλη στατιστική κατανομή. Εάν η στατιστική δοκιμής είναι μεγαλύτερη από την κρίσιμη τιμή, η μηδενική υπόθεση απορρίπτεται υπέρ της εναλλακτικής υπόθεσης. Εάν η στατιστική δοκιμής είναι μικρότερη από την κρίσιμη τιμή, η μηδενική υπόθεση δεν απορρίπτεται.
5. Ερμηνεία αποτελεσμάτων: Τέλος, τα αποτελέσματα του τεστ υποθέσεων ερμηνεύονται στο πλαίσιο του ερευνητικού ερωτήματος. Εάν η μηδενική υπόθεση απορριφθεί, συνάγεται το συμπέρασμα ότι υπάρχουν στοιχεία που υποστηρίζουν την εναλλακτική υπόθεση. Εάν η μηδενική υπόθεση δεν απορριφθεί, συμπεραίνεται ότι δεν υπάρχουν αρκετά στοιχεία που να υποστηρίζουν την εναλλακτική υπόθεση.

Ο έλεγχος υποθέσεων είναι ένα ισχυρό εργαλείο για τη λήψη αποφάσεων που βασίζονται σε δεδομένα. Χρησιμοποιείται ευρέως στην έρευνα, τη βιομηχανία και άλλους τομείς για να απαντήσει σε ερωτήσεις σχετικά με τον πληθυσμό, όπως εάν ένα νέο φάρμακο είναι αποτελεσματικό ή εάν μια νέα πολιτική λειτουργεί. Ωστόσο, είναι σημαντικό να σημειωθεί ότι ο έλεγχος υποθέσεων έχει περιορισμούς και υποθέσεις που πρέπει να πληρούνται προκειμένου τα αποτελέσματα να είναι έγκυρα. Επομένως, είναι σημαντικό να σχεδιάσετε προσεκτικά το τεστ υποθέσεων και να ερμηνεύσετε τα αποτελέσματα στο κατάλληλο πλαίσιο.

2.3.2 GRUBB's test

Η δοκιμή Grubbs, γνωστή και ως δοκιμή ακραίων τιμών Grubbs, είναι μια στατιστική δοκιμή που χρησιμοποιείται για την ανίχνευση ακραίων τιμών σε ένα μονομεταβλητό σύνολο δεδομένων που

ακολουθεί μια κανονική κατανομή. Αυτό το τεστ πήρε το όνομά του από τον Αμερικανό μαθηματικό, Frank E. Grubbs. (Grubbs, 1969)

Η δοκιμή του Grubbs βασίζεται στη μηδενική υπόθεση ότι δεν υπάρχουν ακραίες τιμές στο σύνολο δεδομένων. Η δοκιμή λειτουργεί με τον υπολογισμό της βαθμολογίας Z της μέγιστης ή ελάχιστης τιμής στο σύνολο δεδομένων και τη σύγκριση με την κρίσιμη τιμή της δοκιμής. Εάν η υπολογιζόμενη βαθμολογία Z είναι μεγαλύτερη από την κρίσιμη τιμή, τότε η μηδενική υπόθεση απορρίπτεται και η μέγιστη ή η ελάχιστη τιμή προσδιορίζεται ως ακραία τιμή.

Απαιτεί τουλάχιστον τρεις παρατηρήσεις στο σύνολο δεδομένων και προϋποθέτει ότι τα δεδομένα ακολουθούν μια κανονική κατανομή. Το τεστ χρησιμοποιείται ευρέως σε διάφορους τομείς, συμπεριλαμβανομένης της μηχανικής, της χημείας και των οικονομικών, για τον εντοπισμό ανώμαλων σημείων δεδομένων που μπορεί να παραμορφώσουν την ανάλυση ή τη μοντελοποίηση των δεδομένων.

Ωστόσο, είναι σημαντικό να σημειωθεί ότι η δοκιμή του Grubbs είναι ευαίσθητη στο μέγεθος του δείγματος και μπορεί να μην είναι αξιόπιστη για μικρά μεγέθη δειγμάτων. Επιπλέον, ανιχνεύει μόνο ακραίες τιμές που υπάρχουν στις ακραίες ουρές της κατανομής και μπορεί να μην είναι αποτελεσματικό για τον εντοπισμό ακραίων τιμών που υπάρχουν στη μέση της κατανομής. Ως εκ τούτου, η δοκιμή Grubbs θα πρέπει να χρησιμοποιείται σε συνδυασμό με άλλες μεθόδους ανίχνευσης ακραίων τιμών για να διασφαλιστεί η ακρίβεια και η αξιοπιστία των αποτελεσμάτων.

Ο τύπος για τη δοκιμή του Grubb είναι:

$$Grubb's\ test\ statistic\ =\ (|X\ -\ mean|)/s$$

όπου X είναι η τιμή που ελέγχεται, mean είναι ο μέσος όρος του συνόλου δεδομένων και s είναι η τυπική απόκλιση του δείγματος. Αυτή η στατιστική δοκιμής συγκρίνεται με μια κρίσιμη τιμή που υπολογίζεται χρησιμοποιώντας την κατανομή t του Student με n-2 βαθμούς ελευθερίας, όπου n είναι το μέγεθος του δείγματος. Εάν η στατιστική δοκιμής είναι μεγαλύτερη από την κρίσιμη τιμή, η τιμή X θεωρείται ακραία τιμή.

2.3.3 Dixon's test

Το τεστ Dixon, γνωστό και ως τεστ Dixon's Q, είναι μια στατιστική δοκιμή που χρησιμοποιείται για την ανίχνευση ακραίων τιμών σε ένα σύνολο δεδομένων. Βασίζεται στην υπόθεση ότι τα δεδομένα ακολουθούν μια κανονική κατανομή. Το τεστ πήρε το όνομά του από τον W. J. Dixon, ο οποίος το εισήγαγε για πρώτη φορά το 1951.

Η δοκιμή του Dixon λειτουργεί συγκρίνοντας τη διαφορά μεταξύ του "ύποπτου" ακραίου σημείου και του πλησιέστερου γείτονά του με το εύρος του συνόλου δεδομένων. Εάν η διαφορά είναι μεγαλύτερη από μια ορισμένη κρίσιμη τιμή, τότε η ύποπτη ακραία τιμή θεωρείται στατιστικά σημαντική και επισημαίνεται ως ακραία τιμή.

Υπάρχουν δύο εκδοχές του τεστ του Dixon: το τεστ άνω ουράς και το τεστ κάτω ουράς. Η δοκιμή άνω ουράς χρησιμοποιείται για την ανίχνευση ακραίων σημείων στο άνω άκρο της κατανομής, ενώ η δοκιμή κάτω ουράς χρησιμοποιείται για την ανίχνευση ακραίων σημείων στο κάτω άκρο της κατανομής.

Το τεστ του Dixon έχει κάποιους περιορισμούς. Πρώτον, υποθέτει ότι τα δεδομένα ακολουθούν μια κανονική κατανομή, η οποία μπορεί να μην ισχύει σε όλες τις περιπτώσεις. Δεύτερον, μπορεί να ανιχνεύσει μόνο ένα ακραίο σημείο κάθε φορά, το οποίο μπορεί να μην είναι αρκετό για σύνολα δεδομένων με πολλαπλές ακραίες τιμές. Τρίτον, είναι ευαίσθητο στο μέγεθος του δείγματος και απαιτούνται μεγαλύτερα μεγέθη δειγμάτων για να επιτευχθούν υψηλότερα επίπεδα στατιστικής ισχύος. Συνοπτικά, η δοκιμή του Dixon είναι ένα χρήσιμο εργαλείο για την ανίχνευση ακραίων τιμών σε σύνολα δεδομένων που ακολουθούν μια κανονική κατανομή. Ωστόσο, θα πρέπει να χρησιμοποιείται με προσοχή και άλλες μέθοδοι ανίχνευσης ακραίων τιμών θα πρέπει επίσης να λαμβάνονται υπόψη ανάλογα με τη φύση των δεδομένων.

Ο τύπος για τη δοκιμή του Dixon είναι:

$$Dixon's\ Q\ test\ statistic\ =\ (|X\_i\ -\ X\_j|)/range$$

όπου X_i και X_j είναι οι τιμές που συγκρίνονται και range είναι το εύρος του συνόλου δεδομένων. Η στατιστική δοκιμής συγκρίνεται με τις κρίσιμες τιμές που καταγράφονται σε στατιστικούς πίνακες, ανάλογα με το μέγεθος του δείγματος και το επίπεδο σημαντικότητας που επιλέχθηκε. Εάν η στατιστική δοκιμής είναι μεγαλύτερη από την κρίσιμη τιμή, η τιμή X_i ή X_j θεωρείται ακραία τιμή.

2.3.4 Rosner's test

Η δοκιμή Rosner, γνωστή και ως δοκιμή ακραίων τιμών Rosner, είναι μια στατιστική δοκιμή που χρησιμοποιείται για την ανίχνευση ακραίων τιμών σε ένα σύνολο δεδομένων. Είναι μια επέκταση του τεστ του Grubbs και βασίζεται στην υπόθεση ότι τα δεδομένα ακολουθούν μια κανονική κατανομή.

Το τεστ του Rosner λειτουργεί προσδιορίζοντας την παρατήρηση με το μεγαλύτερο απόλυτο τυποποιημένο υπόλοιπο, το οποίο υπολογίζεται διαιρώντας τη διαφορά μεταξύ του ύποπτου ακραίου και του μέσου όρου με την τυπική απόκλιση των δεδομένων. Εάν το απόλυτο τυποποιημένο υπόλοιπο είναι μεγαλύτερο από μια ορισμένη κρίσιμη τιμή, τότε το ύποπτο ακραίο στοιχείο θεωρείται στατιστικά σημαντικό και επισημαίνεται ως ακραίο. (Feasel, 2022)

Ένα από τα πλεονεκτήματα της δοκιμής Rosner είναι ότι μπορεί να ανιχνεύσει πολλαπλές ακραίες τιμές σε ένα σύνολο δεδομένων, ενώ η δοκιμή του Dixon μπορεί να ανιχνεύσει μόνο μία ακραία τιμή τη φορά. Επιπλέον, το τεστ Rosner δεν είναι τόσο ευαίσθητο στο μέγεθος του δείγματος όσο το τεστ Dixon.

Ωστόσο, όπως και άλλες μέθοδοι ανίχνευσης ακραίων τιμών, η δοκιμή Rosner έχει ορισμένους περιορισμούς. Πρώτον, υποθέτει ότι τα δεδομένα ακολουθούν μια κανονική κατανομή, η οποία μπορεί να μην ισχύει σε όλες τις περιπτώσεις. Δεύτερον, μπορεί να μην είναι αποτελεσματικό για τον εντοπισμό ακραίων τιμών σε σύνολα δεδομένων με μικρό μέγεθος δείγματος. Τέλος, το τεστ μπορεί να επηρεαστεί από την παρουσία σημαντικών παρατηρήσεων, που μπορεί να αλλοιώσουν τα αποτελέσματα.

Συνοπτικά, το τεστ του Rosner είναι ένα χρήσιμο εργαλείο για την ανίχνευση ακραίων σημείων σε σύνολα δεδομένων που ακολουθούν μια κανονική κατανομή, ειδικά όταν υπάρχουν πολλά ύποπτα ακραία

σημεία. Ωστόσο, θα πρέπει να χρησιμοποιείται με προσοχή και άλλες μέθοδοι ανίχνευσης ακραίων τιμών θα πρέπει επίσης να λαμβάνονται υπόψη ανάλογα με τη φύση των δεδομένων.

Ο τύπος για τη δοκιμή του Rosner είναι:

$$Rosner's\ test\ statistic\ =\ (|X\_i\ -\ median|)/MAD\_i$$

όπου X_i είναι η τιμή που ελέγχεται, διάμεσος είναι η διάμεσος του συνόλου δεδομένων και MAD_i είναι η διάμεση απόλυτη απόκλιση του συνόλου δεδομένων μετά την αφαίρεση της τιμής X_i. Το στατιστικό τεστ συγκρίνεται με κρίσιμες τιμές που υπολογίζονται χρησιμοποιώντας τη μέθοδο Extreme Studentized Deviate (ESD). Εάν η στατιστική δοκιμής είναι μεγαλύτερη από την κρίσιμη τιμή, η τιμή X_i θεωρείται ακραία τιμή. Η δοκιμή επαναλαμβάνεται μέχρι να μην ανιχνευθούν άλλες ακραίες τιμές.

2.3.6 Standard Deviation

Η μέθοδος τυπικής απόκλισης είναι μια κοινή στατιστική τεχνική που χρησιμοποιείται στην ανίχνευση ανωμαλιών για τον εντοπισμό ακραίων τιμών σε ένα σύνολο δεδομένων. Βασίζεται στην υπόθεση ότι τα σημεία δεδομένων σε ένα σύμπλεγμα κανονικής κατανομής συγκεντρώνονται στενά μεταξύ τους, ενώ οι ανωμαλίες αποκλίνουν σημαντικά από τον μέσο όρο.

Τα βασικά βήματα της μεθόδου για τον εντοπισμό ανωμαλιών είναι:

1. Υπολογισμός της μέσης και τυπικής απόκλισης: Αρχικά, υπολογίζουμε τον μέσο όρο και την τυπική απόκλιση του συνόλου δεδομένων. Ο μέσος όρος αντιπροσωπεύει την κεντρική τάση των δεδομένων, ενώ η τυπική απόκλιση μετρά τη διασπορά ή την εξάπλωση γύρω από το μέσο όρο.

2. Ορισμός κατωφλίου (threshold): Καθορίζουμε ένα όριο για τον εντοπισμό ανωμαλιών με βάση την τυπική απόκλιση. Αυτό το όριο μπορεί να οριστεί ως πολλαπλάσιο της τυπικής απόκλισης, όπως 2 ή 3 τυπικές αποκλίσεις μακριά από τον μέσο όρο. Τα σημεία δεδομένων που υπερβαίνουν αυτό το όριο θεωρούνται πιθανές ανωμαλίες.

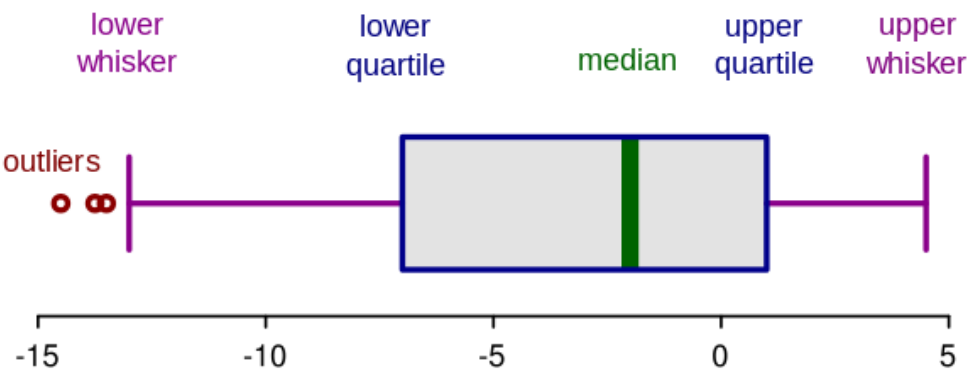
3. Προσδιορισμός ανωμαλιών: Συγκρίνουμε κάθε σημείο δεδομένων στο σύνολο δεδομένων με το όριο. Εάν η τιμή ενός σημείου δεδομένων υπερβαίνει το όριο, επισημαίνεται ως ανωμαλία. Αυτά τα σημεία διαφέρουν σημαντικά από τα περισσότερα δεδομένα και είναι δυνητικά ενδεικτικά ανώμαλης συμπεριφοράς.

4. Προσαρμογή κατωφλίου ή εξέταση περιβάλλοντος: Ανάλογα με τη συγκεκριμένη εφαρμογή και το επιθυμητό επίπεδο ευαισθησίας, μπορεί να χρειαστεί να προσαρμόσουμε το threshold ή να λάβουμε υπόψη πληροφορίες για τα συμφραζόμενα για να εντοπίσουμε τις πραγματικές ανωμαλίες και από τα σημεία τα οποία απλά ξεπερνάνε το threshold. Δεν είναι απαραίτητο όλα τα σημεία δεδομένων που υπερβαίνουν το όριο ανωμαλίες, ειδικά εάν βρίσκονται εντός ενός αποδεκτού εύρους.

2.3.7 Interquartile Range Method – IQR

Το IQR σημαίνει Interquartile Range. Είναι ένα στατιστικό μέτρο που χρησιμοποιείται για την αξιολόγηση της εξάπλωσης ή της μεταβλητότητας ενός συνόλου δεδομένων. Το IQR υπολογίζεται από τη διαφορά μεταξύ του τρίτου τεταρτημορίου (Q3) και του πρώτου τεταρτημορίου (Q1) ενός συνόλου δεδομένων.

Στην ανίχνευση ανωμαλιών, το IQR χρησιμοποιείται συχνά ως όριο για τον εντοπισμό ακραίων ή ανωμαλιών σε ένα σύνολο δεδομένων. Το IQR χρησιμοποιείται στη μέθοδο Tukey's fences, όπου τα ακραία σημεία ορίζονται ως σημεία δεδομένων που πέφτουν κάτω από το Q1 - k * IQR ή πάνω από το Q3 + k * IQR, όπου το k είναι μια σταθερά που ορίζεται από τον χρήστη που συνήθως ορίζεται σε 1,5 ή 3. Εμείς την έχουμε ορίσει ως 1,5.



Εικόνα 2-3-7 Οπτικοποίηση IQR

2.3.8 Z-score & Modified Z-score

Η μέθοδος Z-score είναι μια στατιστική τεχνική που χρησιμοποιείται στην ανίχνευση ανωμαλιών για τον εντοπισμό ακραίων τιμών σε ένα σύνολο δεδομένων. Μετρά πόσες τυπικές αποκλίσεις απέχει ένα σημείο δεδομένων από τον μέσο όρο. Το Z-score παρέχει μια κανονικοποιημένη τιμή που επιτρέπει τη σύγκριση μεταξύ διαφορετικών συνόλων δεδομένων και κατανομών.

Τα βήματα της μεθόδου είναι τα εξής:

1. Υπολογισμός της μέσης και τυπικής απόκλισης: Υπολογίζουμε τη μέση και τυπική απόκλιση του συνόλου δεδομένων. Ο μέσος όρος αντιπροσωπεύει τη μέση τιμή των δεδομένων, ενώ η τυπική απόκλιση μετρά τη διασπορά ή την εξάπλωση γύρω από τη μέση τιμή.

2. Υπολογισμός της βαθμολογίας Z: Για κάθε σημείο δεδομένων στο σύνολο δεδομένων, υπολογίζουμε τη βαθμολογία Z αφαιρώντας τη μέση τιμή από την τιμή του σημείου δεδομένων και διαιρώντας το αποτέλεσμα με την τυπική απόκλιση. Ο τύπος για τον υπολογισμό του Z-score είναι:
$Z = (X - \mu) / \sigma$,
όπου Z είναι η βαθμολογία Z, X είναι η τιμή του σημείου δεδομένων, μ είναι η μέση τιμή και σ είναι η τυπική απόκλιση.

3. Καθορισμός κατωφλίου: Καθορίζουμε ένα όριο για τον εντοπισμό ανωμαλιών με βάση τις βαθμολογίες Z. Συνήθως, χρησιμοποιείται ένα όριο Z-score 2 ή 3, το οποίο αντιστοιχεί σε σημεία δεδομένων που απέχουν δύο ή τρεις τυπικές αποκλίσεις από τη μέση τιμή. Τα σημεία δεδομένων με βαθμολογίες Z πέρα από αυτό το όριο θεωρούνται πιθανές ανωμαλίες.

4. Προσδιορισμός ανωμαλιών: Συγκρίνουμε τη βαθμολογία Z κάθε σημείου δεδομένων με το όριο. Εάν η βαθμολογία Z ενός σημείου δεδομένων υπερβαίνει το όριο, επισημαίνεται ως ανωμαλία. Αυτά τα σημεία δεδομένων έχουν σημαντικά διαφορετική τιμή σε σύγκριση με την πλειονότητα των δεδομένων και είναι πιθανές ακραίες τιμές.

5. Προσαρμογή κατωφλίου ή Εξέταση περιβάλλοντος: Παρόμοια με τη μέθοδο τυπικής απόκλισης, μπορεί να είναι απαραίτητο να προσαρμόσουμε το threshold του Z-score ή να λάβουμε υπόψη πληροφορίες σχετικά με τα συμφραζόμενα. Δεν είναι απαραίτητα όλα τα σημεία δεδομένων που υπερβαίνουν το όριο ανωμαλίες, ειδικά εάν εμπίπτουν σε ένα αποδεκτό εύρος.

Η μέθοδος modified Z-score είναι μια βελτιωμένη έκδοση της μεθόδου Z-score που αντιμετωπίζει το ζήτημα των ακραίων τιμών σε σύνολα δεδομένων με λοξές ή μη κανονικές κατανομές. Είναι μια στατιστική τεχνική που χρησιμοποιείται στην ανίχνευση ανωμαλιών για τον προσδιορισμό των ακραίων τιμών με βάση την απόκλιση τους από τη διάμεσο (median).

Τα βήματα της μεθόδου είναι τα εξής:

1. Υπολογισμός median και median absolute deviation (MAD): Αντί να χρησιμοποιησουμε τη μέση και τυπική απόκλιση όπως στο κλασικό Z-score, υπολογίζουμε τη διάμεσο και τη διάμεση απόλυτη απόκλιση (MAD) του συνόλου δεδομένων. Η διάμεσος αντιπροσωπεύει την κεντρική τιμή των δεδομένων και η MAD μετρά τη διασπορά γύρω από τη διάμεσο.

2. Υπολογισμός του modified Z-score: Για κάθε τιμή στο σύνολο δεδομένων, υπολογίζουμε το modified Z-score αφαιρώντας τη διάμεσο από την τιμή του σημείου δεδομένων και διαιρώντας το αποτέλεσμα με τη διάμεση απόλυτη απόκλιση. Ο τύπος για τον υπολογισμό της τροποποιημένης βαθμολογίας Z είναι: $modified\ Z-score\ =\ 0{,}6745\ *\ (X\ -\ median)\ /\ MAD$.

3. Καθορισμός κατωφλίου: Καθορίζουμε ένα όριο για τον εντοπισμό ανωμαλιών με βάση το modified Z-score. Συνήθως, χρησιμοποιείται ένα τροποποιημένο όριο βαθμολογίας Z 2,5 ή 3, το οποίο αντιστοιχεί σε σημεία δεδομένων που απέχουν 2,5 ή 3 φορές τη διάμεση απόλυτη απόκλιση από τη διάμεσο. Τα σημεία δεδομένων με modified Z-score πέρα από αυτό το όριο θεωρούνται πιθανές ανωμαλίες.

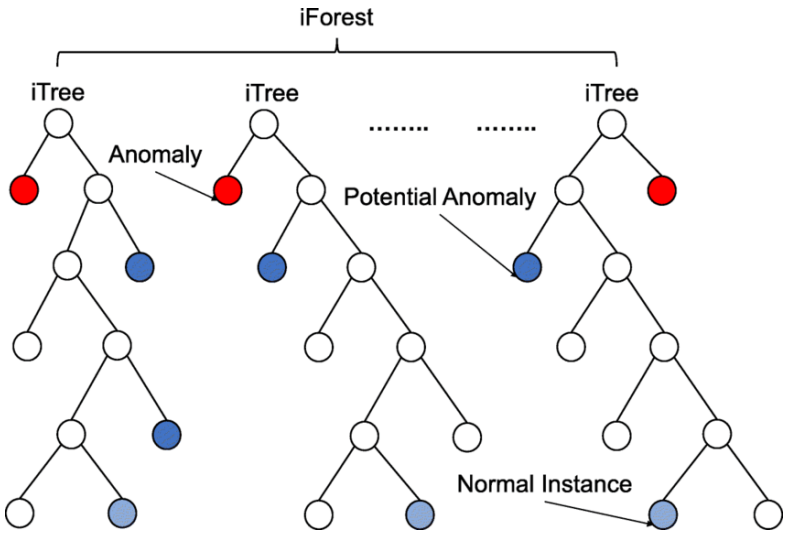
4. Προσδιορισμός ανωμαλιών: Συγκρίνουμε το modified Z-score κάθε σημείου δεδομένων με το threshold. Εάν το modified Z-score ενός σημείου δεδομένων υπερβαίνει το όριο, επισημαίνεται ως ανωμαλία. Αυτά τα σημεία δεδομένων έχουν σημαντικά διαφορετική τιμή σε σύγκριση με την πλειονότητα των δεδομένων και είναι πιθανές ακραίες τιμές.

5. Προσαρμογή κατωφλίου ή εξέταση περιβάλλοντος: Όπως είδαμε προγουμένως, μπορεί να είναι απαραίτητο να προσαρμόσουμε το threshold του modified Z-score ή να λάβουμε υπόψη τις πληροφορίες συμφραζομένων για να εντοπίσουμε τις πραγματικές ανωμαλίες. Δεν είναι απαραίτητα όλα τα σημεία δεδομένων που υπερβαίνουν το όριο ανωμαλίες, ειδικά εάν εμπίπτουν σε ένα αποδεκτό εύρος.

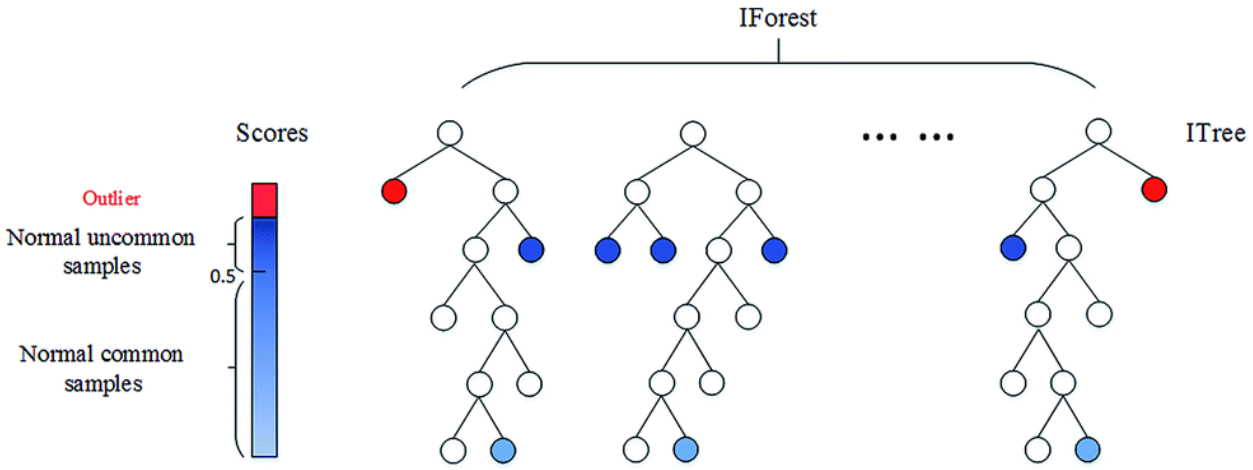
2.4 Παραδοσιακές Μέθοδοι Ανίχνευσης Ανωμαλιών

2.4.1 Isolation forest

Το Isolation Forest είναι ένας αλγόριθμος για τον εντοπισμό ανωμαλιών ή ακραίων τιμών σε ένα σύνολο δεδομένων. Βασίζεται στην έννοια της απομόνωσης, όπου ο αλγόριθμος απομονώνει μεμονωμένα σημεία δεδομένων επιλέγοντας τυχαία ένα χαρακτηριστικό και στη συνέχεια επιλέγοντας τυχαία μια διαίρεση μεταξύ της ελάχιστης και της μέγιστης τιμής αυτού του χαρακτηριστικού. Ο αριθμός των διαχωρισμών (partitions) που απαιτούνται για την απομόνωση ενός σημείου ονομάζεται βαθμολογία απομόνωσής (anomaly score) και τα σημεία με χαμηλότερη βαθμολογία απομόνωσης θεωρούνται πιο μη φυσιολογικά ή ανώμαλα. Ο αλγόριθμος λειτουργεί δημιουργώντας ένα δάσος απομονωμένων δέντρων και κάθε δέντρο εκπαιδεύεται σε ένα τυχαίο υποσύνολο δεδομένων. Τα ανώμαλα σημεία αναμένεται να έχουν μικρότερες διαδρομές από τη ρίζα και έτσι να απομονώνονται ταχύτερα από τα κανονικά σημεία.



Εικόνα 2.4 Isolation forest visualization



Εικόνα 2-5 Isolation forest visualization (ii)

Ακολουθεί μια εξήγηση βήμα προς βήμα για το πώς λειτουργεί το Isolation Forest:

1. Πρώτον, ο αλγόριθμος επιλέγει τυχαία ένα χαρακτηριστικό (feature) από το σύνολο δεδομένων.
2. Στη συνέχεια, επιλέγει μια τυχαία τιμή για αυτό το χαρακτηριστικό που εμπίπτει μεταξύ της ελάχιστης και της μέγιστης τιμής αυτού του χαρακτηριστικού στο σύνολο δεδομένων.
3. Στη συνέχεια, χωρίζει τα δεδομένα με βάση το εάν κάθε σημείο δεδομένων έχει μια τιμή για αυτό το χαρακτηριστικό που είναι μεγαλύτερη ή μικρότερη από την επιλεγμένη τιμή. Τα σημεία δεδομένων που έχουν τιμή πάνω από την επιλεγμένη τιμή τοποθετούνται σε ένα partition και τα σημεία δεδομένων που έχουν τιμή κάτω από την επιλεγμένη τιμή τοποθετούνται στο άλλο partition.
4. Ο αλγόριθμος συνεχίζει αυτή τη διαδικασία αναδρομικά, επιλέγοντας τυχαία χαρακτηριστικά και χωρίζοντας τα δεδομένα έως ότου κάθε σημείο δεδομένων βρίσκεται στο δικό του partition ή μέχρι να επιτευχθεί ένα μέγιστο βάθος.
5. Μόλις ολοκληρωθεί η κατάτμηση, ο αλγόριθμος υπολογίζει το μέσο μήκος διαδρομής (δηλαδή τον αριθμό των partitions από τα οποία διέρχεται ένα σημείο δεδομένων κατά μέσο όρο) για κάθε σημείο δεδομένων στο σύνολο δεδομένων.
6. Τέλος, ο αλγόριθμος χρησιμοποιεί το μέσο μήκος διαδρομής για τον εντοπισμό ανωμαλιών. Τα σημεία δεδομένων που έχουν μικρότερο μέσο μήκος διαδρομής από ένα συγκεκριμένο όριο θεωρούνται ανωμαλίες.

Η διαίσθηση πίσω από αυτήν την προσέγγιση είναι ότι οι ανωμαλίες εντοπίζονται συνήθως σε περιοχές του χώρου χαρακτηριστικών που είναι αραιοκατοικημένες, έτσι ώστε να μπορούν να απομονωθούν πιο εύκολα από τα κανονικά σημεία δεδομένων.

Το Isolation Forest είναι γνωστό ότι είναι ένας πολύ αποδοτικός και αποτελεσματικός αλγόριθμος για την ανίχνευση ανωμαλιών, ειδικά για σύνολα δεδομένων υψηλών διαστάσεων. Έχει γραμμική χρονική πολυπλοκότητα, καθιστώντας το αποτελεσματικό για μεγάλα σύνολα δεδομένων και μπορεί να χειριστεί κατηγορικά και αριθμητικά δεδομένα.

Επιπλέον, το Isolation Forest έχει κάποια πλεονεκτήματα σε σχέση με άλλες τεχνικές ανίχνευσης ανωμαλιών, όπως μεθόδους που βασίζονται σε PCA, μεθόδους που βασίζονται σε πυκνότητα και

μεθόδους που βασίζονται σε πλησιέστερους γείτονες, οι οποίες μπορεί να είναι ευαίσθητες στην κατανομή πυκνότητας των κανονικών περιπτώσεων και στην παρουσία θορύβου στο δεδομένα.

Είναι σημαντικό να σημειωθεί ότι το Isolation Forest δεν απαιτεί τα δεδομένα να διανέμονται κανονικά (κανονική κατανομή) και δεν απαιτεί την επισήμανση των δεδομένων (labeling). Αυτό το καθιστά χρήσιμο για εργασίες ανίχνευσης ανωμαλιών χωρίς επίβλεψη.

Μπορεί να εφαρμοστεί σε δεδομένα χρονοσειρών για ανίχνευση ανωμαλιών. Ωστόσο, απαιτεί ορισμένες τροποποιήσεις στον αλγόριθμο για να γίνει κατάλληλος για αυτό το σκοπό. Όταν ασχολούμαστε με δεδομένα χρονοσειρών, πρέπει να λάβουμε υπόψη τη χρονική εξάρτηση των σημείων δεδομένων. Η διαδοχική σειρά των σημείων δεδομένων σε μια χρονοσειρά είναι ένα σημαντικό χαρακτηριστικό που πρέπει να ληφθεί υπόψη προκειμένου να εντοπιστούν με ακρίβεια οι ανωμαλίες. Μια προσέγγιση είναι η χρήση sliding windows (συρόμενων παραθύρων) για τη δημιουργία διανυσμάτων χαρακτηριστικών που καταγράφουν χρονικές εξαρτήσεις. Κάθε παράθυρο μπορεί να θεωρηθεί ως ένα σημείο σε ένα χώρο χαρακτηριστικών υψηλών διαστάσεων. Το Isolation Forest μπορεί στη συνέχεια να εφαρμοστεί σε αυτά τα διανύσματα χαρακτηριστικών για τον εντοπισμό ανωμαλιών.

Μια άλλη προσέγγιση είναι να χρησιμοποιήσετε μια επέκταση του αρχικού αλγορίθμου Isolation Forest, που ονομάζεται Isolation Forest for Time Series (iForestTS). Το iForestTS χρησιμοποιεί μια προσέγγιση συρόμενου παραθύρου για να εξάγει χαρακτηριστικά από τα δεδομένα χρονοσειρών και, στη συνέχεια, εφαρμόζει το Δάσος απομόνωσης σε αυτές τις δυνατότητες. Ο αλγόριθμος iForestTS λαμβάνει υπόψη τόσο τις χρονικές εξαρτήσεις όσο και τα τοπικά χαρακτηριστικά των δεδομένων χρονοσειρών, καθιστώντας τον κατάλληλο για ανίχνευση ανωμαλιών σε χρονοσειρές.

Συνοπτικά, το Isolation Forest μπορεί να χρησιμοποιηθεί για ανίχνευση ανωμαλιών σε δεδομένα χρονοσειρών, αλλά απαιτεί τροποποιήσεις για να ληφθούν υπόψη οι χρονικές εξαρτήσεις των δεδομένων. Ο αλγόριθμος iForestTS είναι μια ευρέως χρησιμοποιούμενη επέκταση που έχει σχεδιαστεί ειδικά για αυτόν τον σκοπό.

2.4.2 One-class support vector machine

Το One-class Support Vector Machine (SVM) είναι ένας τύπος αλγορίθμου SVM που χρησιμοποιείται για τον εντοπισμό ανωμαλιών. Είναι ένας αλγόριθμος μάθησης με ημι-επίβλεψη, αν και μπορεί να χρησιμοποιηθεί και για ανίχνευση ανωμαλιών χωρίς επίβλεψη. Η βασική ιδέα είναι ότι το μοντέλο εκπαιδεύεται σε μία μόνο κατηγορία δεδομένων, η οποία θεωρείται ότι είναι κανονική ή μη ανώμαλη και χρησιμοποιείται για να ανιχνεύσει ανωμαλίες όταν νέα δεδομένα του παρουσιάζονται. Ο στόχος του αλγορίθμου είναι να μάθει το όριο απόφασης (decision boundary) που διαχωρίζει την κανονική κλάση από τον υπόλοιπο χώρο χαρακτηριστικών (feature space), που αντιπροσωπεύει τα ανώμαλα ή ακραία σημεία δεδομένων.

Ο αλγόριθμος SVM μιας κλάσης δημιουργεί ένα υπερεπίπεδο που μεγιστοποιεί την απόσταση (περιθώριο) μεταξύ των πλησιέστερων σημείων δεδομένων της κανονικής κλάσης και του υπερεπίπεδου. Η απόσταση ενός σημείου δοκιμής από το υπερεπίπεδο χρησιμοποιείται στη συνέχεια για να προσδιοριστεί εάν πρόκειται για ανωμαλία ή όχι. Τα σημεία που είναι πιο μακριά από το υπερεπίπεδο θεωρούνται πιο ανώμαλα από τα σημεία που είναι πιο κοντά στο υπερεπίπεδο.

Το SVM μιας κλάσης έχει πολλά πλεονεκτήματα σε σχέση με άλλες τεχνικές ανίχνευσης ανωμαλιών. Είναι ανθεκτικό σε δεδομένα υψηλών διαστάσεων και μπορεί να χειριστεί μη γραμμικά όρια απόφασης. Επιπλέον, μπορεί να χρησιμοποιηθεί τόσο για εργασίες εκτίμησης πυκνότητας όσο και για εργασίες ταξινόμησης, καθιστώντας τον έναν ευέλικτο αλγόριθμο.

Είναι σημαντικό να σημειωθεί ότι το One-class SVM υποθέτει ότι τα δεδομένα είναι γραμμικά διαχωρίσιμα και απαιτεί την επισήμανση (labeling) των δεδομένων. Επιπλέον, είναι ευαίσθητο στην επιλογή της συνάρτησης πυρήνα (kernel) και στην τιμή της παραμέτρου κανονικοποίησης (regularization). Επομένως, είναι σημαντικό να ρυθμίσει κανείς σωστά αυτές τις παραμέτρους.

Συνοπτικά, το SVM μιας κατηγορίας είναι ένας αποδοτικός και αποτελεσματικός αλγόριθμος για την ανίχνευση ανωμαλιών, ειδικά για σύνολα δεδομένων υψηλών διαστάσεων, και μπορεί να χειριστεί τόσο γραμμικά όσο και μη γραμμικά όρια απόφασης. Ωστόσο, απαιτεί δεδομένα με ετικέτα και είναι ευαίσθητο στην επιλογή της λειτουργίας του πυρήνα και της παραμέτρου κανονικοποίησης.

Συνάρτηση πυρήνα

Πιο αναλυτικά στο πλαίσιο των OC-SVM, ένας πυρήνας είναι μια συνάρτηση που λαμβάνει δεδομένα εισόδου χαμηλής διάστασης και τα μετατρέπει σε χώρο υψηλότερων διαστάσεων, όπου γίνεται ευκολότερο να βρεθεί ένα γραμμικό όριο που χωρίζει τα σημεία δεδομένων. Η ιδέα πίσω από τη χρήση ενός πυρήνα είναι ότι επιτρέπει στα SVM να εκτελούν σύνθετες εργασίες μη γραμμικής ταξινόμησης, παρόλο που ο ίδιος ο αλγόριθμος είναι ένας γραμμικός ταξινομητής. Μια συνάρτηση πυρήνα μπορεί να θεωρηθεί ως μια συνάρτηση ομοιότητας, η οποία μετρά την ομοιότητα μεταξύ δύο σημείων δεδομένων στον αρχικό χώρο εισόδου. Χρησιμοποιώντας έναν πυρήνα, ο αλγόριθμος SVM μπορεί να βρει ένα όριο απόφασης στον χώρο υψηλότερης διάστασης που αντιστοιχεί σε ένα μη γραμμικό όριο στον αρχικό χώρο εισόδου.

Υπάρχουν διάφοροι τύποι συναρτήσεων πυρήνα που μπορούν να χρησιμοποιηθούν στο SVM, όπως:

- Γραμμικός πυρήνας: Αυτός ο πυρήνας υπολογίζει απλώς το βαθμωτό γινόμενο μεταξύ δύο σημείων δεδομένων στον αρχικό χώρο εισαγωγής. Είναι ο προεπιλεγμένος πυρήνας που χρησιμοποιείται στο SVM.
- Πολυωνυμικός πυρήνας: Αυτός ο πυρήνας υπολογίζει το βαθμωτό γινόμενο μεταξύ δύο σημείων δεδομένων που υψομένων σε μια συγκεκριμένη δύναμη. Μπορεί να χρησιμοποιηθεί για τη μοντελοποίηση ορίων πολυωνυμικής απόφασης.
- Πυρήνας συνάρτησης ακτινικής βάσης (RBF): Αυτός ο πυρήνας υπολογίζει το εκθετικό του τετραγώνου της απόστασης μεταξύ δύο σημείων δεδομένων. Μπορεί να χρησιμοποιηθεί για τη μοντελοποίηση μη γραμμικών ορίων απόφασης που είναι ομαλά και κυκλικά.
- Σιγμοειδής πυρήνας: Αυτός ο πυρήνας υπολογίζει τη λογιστική συνάρτηση του βαθμωτού γινόμενου μεταξύ δύο σημείων δεδομένων. Μπορεί να χρησιμοποιηθεί για τη μοντελοποίηση ορίων απόφασης που έχουν σχήμα S.

Αξίζει επίσης να σημειωθεί ότι η επιλογή του σωστού πυρήνα και των κατάλληλων παραμέτρων πυρήνα είναι κρίσιμη για την απόδοση του SVM. Στην πράξη, ο γραμμικός πυρήνας είναι ο απλούστερος και ο πιο αποτελεσματικός πυρήνας, ωστόσο, μπορεί να μην είναι κατάλληλος για όλα τα σύνολα δεδομένων. Ο πυρήνας της συνάρτησης ακτινικής βάσης (RBF) είναι μια καλή προεπιλεγμένη επιλογή για ένα ευρύ φάσμα συνόλων δεδομένων.

Παράμετρος κανονικοποίησης

Στο πλαίσιο των OC-SVM η κανονικοποίηση είναι μια τεχνική που χρησιμοποιείται για την αποφυγή της υπερπροσαρμογής (overfitting) του μοντέλου στα δεδομένα εκπαίδευσης (training data). Η υπερπροσαρμογή συμβαίνει όταν το μοντέλο είναι πολύ περίπλοκο και γίνεται πολύ εξειδικευμένο στα δεδομένα εκπαίδευσης, γεγονός που μπορεί να οδηγήσει σε κακή απόδοση γενίκευσης σε νέα, άγνωστα δεδομένα.

Η κανονικοποίηση στα OC-SVM επιτυγχάνεται προσθέτοντας έναν όρο ποινής στην αντικειμενική συνάρτηση που προσπαθεί να βελτιστοποιήσει ο αλγόριθμος. Η αντικειμενική λειτουργία των OC-SVM όπως αναφέραμε και προηγουμένως είναι να βρουν ένα όριο, που ονομάζεται όριο απόφασης, το οποίο διαχωρίζει την κανονική κλάση από τον υπόλοιπο χώρο χαρακτηριστικών, ενώ μεγιστοποιεί το περιθώριο μεταξύ της κανονικής κλάσης και του ορίου απόφασης. Ο όρος τακτοποίησης προστίθεται στην αντικειμενική συνάρτηση για να τιμωρήσει μοντέλα που έχουν μεγάλο αριθμό διανυσμάτων υποστήριξης (support vectors) , δηλαδή σημεία δεδομένων που βρίσκονται κοντά στο όριο απόφασης, ή μεγάλο περιθώριο (margin).

Με άλλα λόγια, ο όρος κανονικοποίησης βοηθά να διατηρείται υπό έλεγχο ο αριθμός των διανυσμάτων υποστήριξης και το μέγεθος του περιθωρίου, προσθέτοντας μια ποινή για πολύπλοκα μοντέλα. Η παράμετρος κανονικοποίησης, επίσης γνωστή ως C, χρησιμοποιείται για τον έλεγχο της αντιστάθμισης μεταξύ του περιθωρίου και του όρου κανονικοποίησης. Μια μικρή τιμή του C αντιστοιχεί σε έναν μεγάλο όρο κανονικοποίησης και ένα μικρότερο περιθώριο, ενώ μια μεγάλη τιμή του C αντιστοιχεί σε έναν μικρό όρο κανονικοποίησης και ένα μεγαλύτερο περιθώριο.

Παράμετρος γάμμα

Στο πλαίσιο των OC-SVMs που χρησιμοποιούν πυρήνα συνάρτησης ακτινικής βάσης (RBF), το γάμμα είναι μια παράμετρος που ελέγχει το πλάτος της συνάρτησης Gauss που χρησιμοποιείται στον πυρήνα. Ο πυρήνας RBF υπολογίζει την εκθετική της τετραγωνικής απόστασης μεταξύ δύο σημείων δεδομένων, η οποία ορίζεται ως k(x,x') = exp(-gamma * ||x-x'||^2) όπου x, x' είναι δεδομένα σημεία και το γάμμα είναι η παράμετρος.

Η παράμετρος γάμμα καθορίζει πόσο μακριά φτάνει η επιρροή ενός μεμονωμένου προπονητικού παραδείγματος (training example), με τις χαμηλές τιμές να σημαίνουν «μακριά» και τις υψηλές «κοντά». Η παράμετρος γάμμα είναι το αντίστροφο της τυπικής απόκλισης της συνάρτησης Gauss. Μια μικρή τιμή γάμμα αντιστοιχεί σε μια ευρεία συνάρτηση Gauss και μια μεγάλη τιμή γάμμα αντιστοιχεί σε μια στενή Gaussian συνάρτηση.

Μια μικρή τιμή γάμμα αντιστοιχεί σε ένα γενικότερο όριο απόφασης, καθώς η συνάρτηση Gauss θα είναι ευρεία και θα εξετάζει περισσότερα σημεία δεδομένων κοντά στο σημείο δοκιμής. Μια μεγάλη τιμή γάμμα αντιστοιχεί σε ένα πιο περίπλοκο όριο απόφασης, καθώς η συνάρτηση Gauss θα είναι στενότερη και θα εξετάζει λιγότερα σημεία δεδομένων κοντά στο σημείο δοκιμής.

2.4.3 Local Outlier Factor - LOF

Το Local Outlier Factor (LOF) είναι ένας μη εποπτευόμενος αλγόριθμος μηχανικής μάθησης για ανίχνευση ανωμαλιών που λειτουργεί μετρώντας την τοπική πυκνότητα κάθε σημείου δεδομένων και συγκρίνοντάς

το με τις πυκνότητες των γειτόνων του. Βασίζεται στην ιδέα ότι τα ανώμαλα σημεία δεδομένων βρίσκονται συχνά σε περιοχές χαμηλής πυκνότητας του χώρου χαρακτηριστικών.

Πώς λειτουργεί ο αλγόριθμος:

Για κάθε σημείο δεδομένων, προσδιορίζουμε τους k πλησιέστερους γείτονές του στο χώρο χαρακτηριστικών. Η τιμή του k είναι μια υπερπαράμετρος που πρέπει να οριστεί πριν από την εκτέλεση του αλγόριθμου.

Υπολογίζουμε την τοπική πυκνότητα προσβασιμότητας (local reachability density - LRD) κάθε σημείου. Το LRD είναι ένα μέτρο του πόσο πυκνή είναι η τοπική γειτονιά ενός σημείου σε σύγκριση με τους γείτονές του. Υπολογίζεται λαμβάνοντας το αντίστροφο της μέσης απόστασης προσβασιμότητας των k πλησιέστερων γειτόνων ενός σημείου.

Υπολογίζουμε το LOF κάθε σημείου. Το LOF είναι ένα μέτρο του πόσο περισσότερο ή λιγότερο πυκνό είναι ένα σημείο σε σύγκριση με τους γείτονές του. Υπολογίζεται λαμβάνοντας τη μέση αναλογία του LRD των k πλησιέστερων γειτόνων ενός σημείου προς το δικό του LRD.

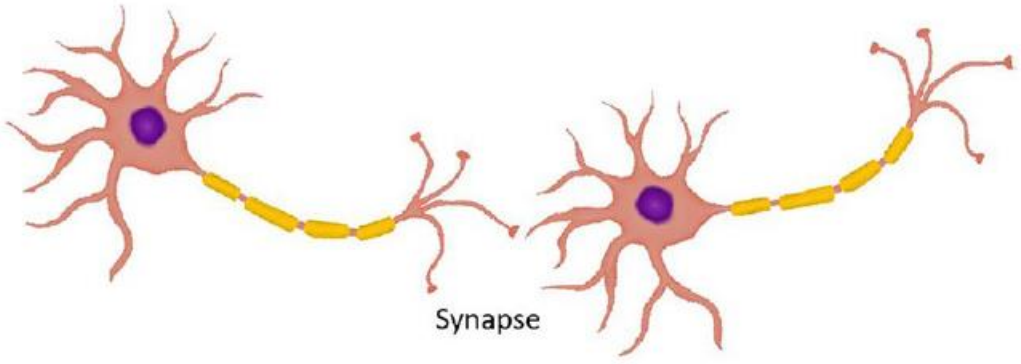
Σημεία με βαθμολογίες LOF σημαντικά μικρότερες από 1 θεωρούνται ανωμαλίες. Το όριο για το τι συνιστά σημαντική διαφορά εξαρτάται από το σύνολο δεδομένων και πρέπει να οριστεί χειροκίνητα.

2.5 Βαθιά Μάθηση - Deep Learning

Η βαθιά μάθηση είναι ένα υποπεδίο της μηχανικής μάθησης που χρησιμοποιεί τεχνητά νευρωνικά δίκτυα (artificial neural networks) με πολλά κρυφά επίπεδα για τη μοντελοποίηση και την επίλυση σύνθετων προβλημάτων. Είναι εμπνευσμένο από τη δομή και τη λειτουργία του ανθρώπινου εγκεφάλου και μπορεί να χρησιμοποιηθεί για εργασίες όπως η ταξινόμηση εικόνων, η αναγνώριση ομιλίας και η επεξεργασία φυσικής γλώσσας.

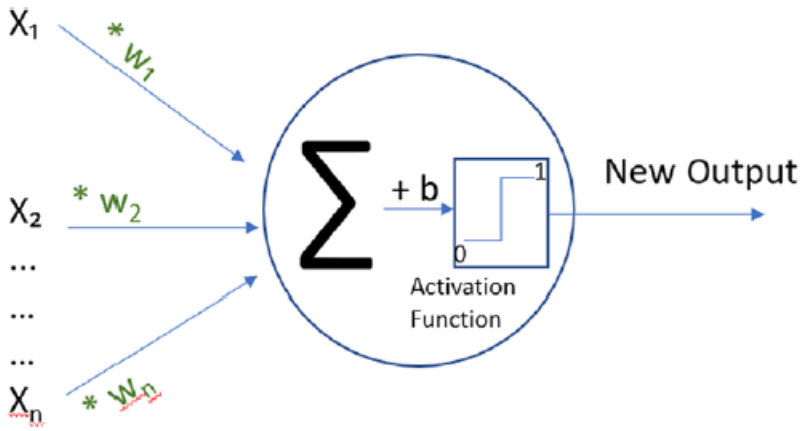
2.5.1 Τεχνητά Νευρωνικά Δίκτυα (Artificial Neural Networks)

Τα τεχνητά νευρωνικά δίκτυα (ANN) είναι αλγόριθμοι που διαμορφώνονται σύμφωνα με τη δομή και τη λειτουργία του ανθρώπινου εγκεφάλου, και συγκεκριμένα, το δίκτυο των νευρώνων και των συνάψεων. Έχουν σχεδιαστεί για να αναγνωρίζουν μοτίβα και να κάνουν προβλέψεις με βάση τα δεδομένα εισόδου.



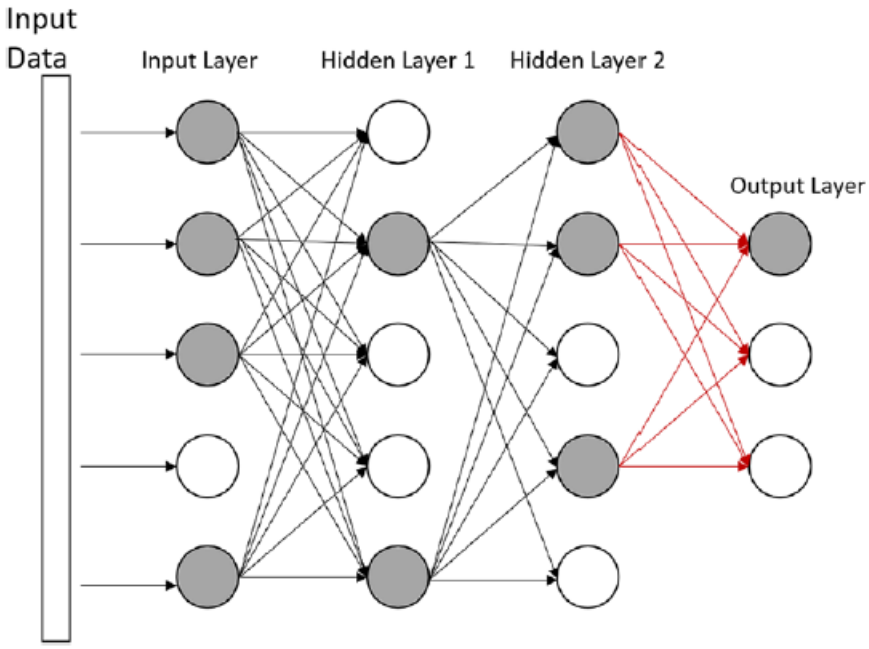
Εικόνα 2-6.Πώς δύο νευρώνες μπορούν να συνδεθούν για να σχηματίσουν μια αλυσίδα και να μεταφέρουν σήματα μέσω αυτής της σύνδεσης. Ο τερματικός άξονας του πρώτου νευρώνα συνδέεται με τους δενδρίτες του δεύτερου νευρώνα

Ένα ANN αποτελείται από πολλούς διασυνδεδεμένους «νευρώνες» που επεξεργάζονται και μεταδίδουν πληροφορίες. Κάθε νευρώνας λαμβάνει είσοδο από άλλους νευρώνες, την επεξεργάζεται χρησιμοποιώντας μια μαθηματική πράξη και παράγει μια έξοδο που μεταδίδεται σε άλλους νευρώνες.



Εικόνα 2-7.Πώς μπορεί να λειτουργήσει ένας τεχνητός νευρώνας σε ένα τεχνητό νευρωνικό δίκτυο. Αυτή η μίμηση του βιολογικού νευρώνα είναι η βάση των τεχνητών νευρωνικών δικτύων

Τα δεδομένα εισόδου συνήθως τροφοδοτούνται στο στρώμα εισόδου του δικτύου (input layer), το οποίο τα περνά μέσα από πολλά κρυφά στρώματα (hidden layers) που αποτελούνται από νευρώνες. Τα κρυφά επίπεδα χρησιμοποιούν μη γραμμικές συναρτήσεις ενεργοποίησης (non-linear activation functions) για να εισάγουν μη γραμμικότητα στο δίκτυο, επιτρέποντάς του να μοντελοποιήσει σύνθετες σχέσεις στα δεδομένα εισόδου. Η έξοδος από το τελικό επίπεδο είναι η πρόβλεψη (prediction) του δικτύου.



Εικόνα 2-8. Απεικόνιση τεχνητού νευρωνικού δικτύου

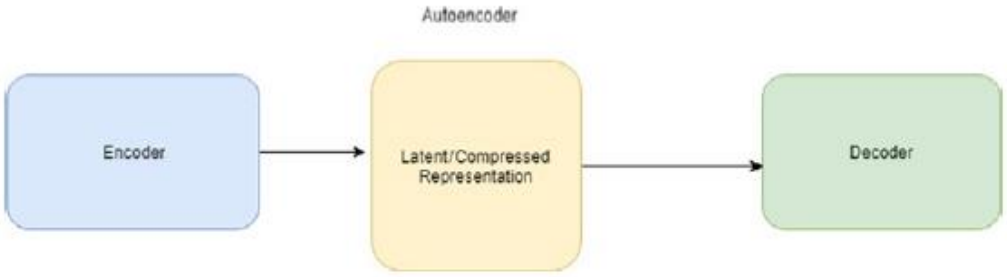
Κατά την αρχικοποίηση του μοντέλου τα βάρη των συνδέσεων μεταξύ των νευρώνων δεν έχουν τις τιμές που ευνοούν ακριβείς προβλέψεις. Για να το διορθώσουμε αυτό χρησιμοποιούμε μια τεχνική που ονομάζεται backpropagation (οπίσθια διάδοση). Ο αλγόριθμος λειτουργεί κάνοντας πρώτα μια πρόβλεψη χρησιμοποιώντας τα τρέχοντα βάρη του δικτύου. Το σφάλμα μεταξύ της πρόβλεψης και της πραγματικής παραγωγής υπολογίζεται στη συνέχεια και χρησιμοποιείται για την ενημέρωση των βαρών με τρόπο που να μειώνει το σφάλμα. Αυτό γίνεται επαναληπτικά για πολλά παραδείγματα από το σύνολο δεδομένων εκπαίδευσης, με στόχο την εύρεση του συνόλου των βαρών που παράγουν το μικρότερο συνολικό σφάλμα. Το backpropagation λειτουργεί υπολογίζοντας τη διαβάθμιση (gradient) ή την παράγωγο (derivative) του σφάλματος σε σχέση με κάθε βάρος στο δίκτυο. Η gradient παρέχει πληροφορίες σχετικά με την κατεύθυνση στην οποία πρέπει να ρυθμιστούν τα βάρη προκειμένου να μειωθεί το σφάλμα. Στη συνέχεια, τα βάρη προσαρμόζονται στην αντίθετη κατεύθυνση της gradient, έτσι ώστε να κινούνται προς μια πιο βέλτιστη λύση.

Ο αλγόριθμος backpropagation είναι ένας αποτελεσματικός τρόπος εκπαίδευσης νευρωνικών δικτύων και αποτελεί το θεμέλιο πολλών σύγχρονων τεχνικών βαθιάς μάθησης. Ωστόσο, μπορεί να είναι υπολογιστικά ακριβός, ειδικά για μεγάλα δίκτυα με πολλά κρυφά επίπεδα, και μπορεί επίσης να κολλήσει σε μη βέλτιστες λύσεις εάν ο ρυθμός εκμάθησης (learning rate) δεν ρυθμιστεί προσεκτικά και ξεπεράσουμε το ζητούμενο τοπικό μέγιστο.

Τα ANN είναι επιτυχημένα σε μια ποικιλία εφαρμογών, συμπεριλαμβανομένης της ταξινόμησης εικόνων, της αναγνώρισης ομιλίας, της επεξεργασίας φυσικής γλώσσας και άλλων. Ωστόσο, μπορεί να είναι υπολογιστικά ακριβά και απαιτούν μεγάλες ποσότητες δεδομένων με ετικέτα για εκπαίδευση. Επιπλέον, η δομή του δικτύου, συμπεριλαμβανομένου του αριθμού των κρυφών επιπέδων και του αριθμού των νευρώνων σε κάθε επίπεδο, πρέπει να καθοριστεί εκ των προτέρων και μπορεί να έχει σημαντικό αντίκτυπο στην απόδοση του δικτύου. Βέβαια οι σύγχρονες GPUs οι οποίες είναι σχεδιασμένες για να εκτελούν πολύ γρήγορα τις πράξεις μεταξύ πινάκων που απαιτεί η βαθιά μάθηση μας διευκολύνουν αρκετά.

2.6 Autoencoders

Τα νευρωνικά δίκτυα αυτόματης κωδικοποίησης (αυτόματοι κωδικοποιητές) είναι ένας τύπος αλγορίθμου μάθησης χωρίς επίβλεψη που χρησιμοποιούνται ευρέως στη βαθιά μάθηση. Αποτελούνται από δύο κύρια μέρη: τον κωδικοποιητή και τον αποκωδικοποιητή.



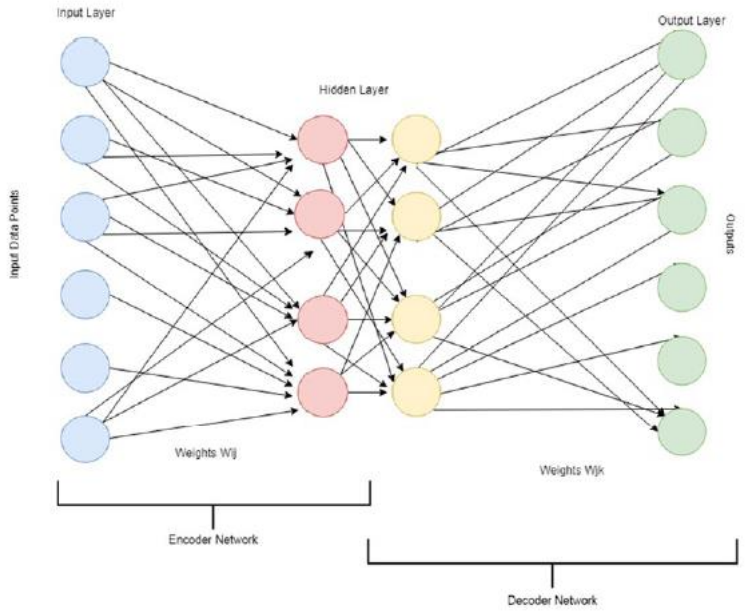
Εικόνα 2-9. Απεικόνιση αυτόματου κωδικοποιητή

Ο κωδικοποιητής λαμβάνει μια είσοδο, συνήθως μια εικόνα ή κάποια άλλα δεδομένα υψηλών διαστάσεων, και τα συμπιέζει σε μια αναπαράσταση χαμηλότερης διάστασης, γνωστή και ως λανθάνων κώδικας (latent code) ή ενσωμάτωση (embedding). Αυτό γίνεται περνώντας την είσοδο μέσω πολλαπλών

στρωμάτων νευρωνικών δικτύων με μειούμενο αριθμό νευρώνων. Ο κωδικοποιητής είναι εκπαιδευμένος να ελαχιστοποιεί την απώλεια ανακατασκευής, η οποία είναι η διαφορά μεταξύ της αρχικής εισόδου και της εξόδου του αποκωδικοποιητή.

Στη συνέχεια, ο αποκωδικοποιητής παίρνει τον λανθάνοντα κώδικα ως είσοδο και προσπαθεί να ανακατασκευάσει την αρχική είσοδο. Αυτό το κάνει περνώντας τον λανθάνοντα κώδικα μέσω πολλαπλών στρωμάτων νευρωνικών δικτύων με αυξανόμενο αριθμό νευρώνων, καταλήγοντας σε ένα στρώμα που εξάγει τον ίδιο αριθμό νευρώνων με την είσοδο. Ο αποκωδικοποιητής είναι επίσης εκπαιδευμένος ώστε να ελαχιστοποιεί την απώλεια ανακατασκευής.

Συνολικά, ο αυτόματος κωδικοποιητής είναι εκπαιδευμένος να βρίσκει μια συμπαγή αναπαράσταση των δεδομένων εισόδου που μπορεί να χρησιμοποιηθεί για την ανακατασκευή της αρχικής εισόδου με υψηλή ακρίβεια. Όσο πιο καλό είναι το νευρωνικό που χρησιμοποιείται, τόσο περισσότερες είναι οι πιθανότητες ανακατασκεύης της εισόδου από τα κωδικοποιημένα δεδομένα. Αυτή η βασική αρχή είναι πολύ σημαντική για τη χρήση των αυτόματων κωδικοποιητών στην ανίχνευση ανωμαλιών.



Εικόνα 2-10. Επεκτεταμένη απεικόνιση αυτόματου κωδικοποιητή

Βασική σημείωση είναι ότι οι αυτόματοι κωδικοποιητές δεν είναι τόσο αποτελεσματικοί αν τα δεδομένα εκπαίδευσης δεν αποτελούνται από πολλές διαστάσεις/χαρακτηριστικά. Είναι αποδοτικοί για δεδομένα πέντε διαστάσεων ή περισσότερων. Αν έχουν μία διάσταση/ ένα χαρακτηριστικό τότε πρακτικά κάνουμε μια γραμμική μετατροπή η οποία δεν είναι χρήσιμη.

Εκτός από την ανίχνευση ανωμαλιών οι αυτόματοι κωδικοποιητές χρησιμοποιούνται για να εκπαιδευτούν δίκτυα βαθιάς μάθησης, για συμπίεση, για ταξινόμηση και για παραγωγικά μοντέλα. Υπάρχουν διάφοροι τύποι νευρωνικών δικτύων αυτόματης κωδικοποίησης, συμπεριλαμβανομένων των απλών, των αραιών, των πυκνών, των συνελικτικών, των μεταβλητών και αυτών που χρησιμοποιούνται για αποθορυβοποίηση, καθένας με τα δικά του πλεονεκτήματα και αδυναμίες.

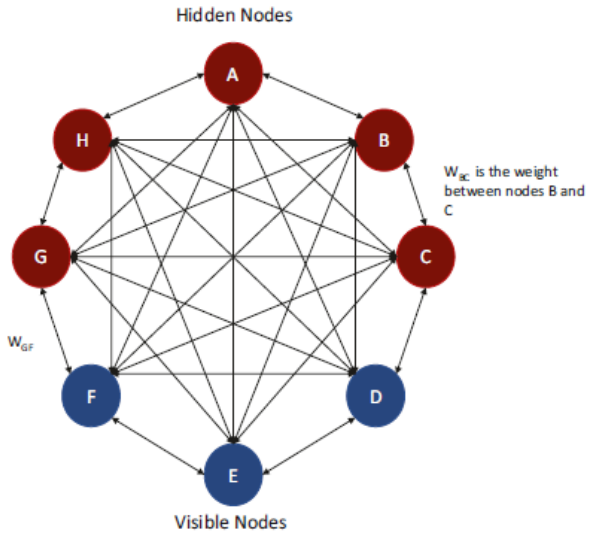
1. Απλοί αυτοκωδικοποιητές: αποτελούνται από έναν κωδικοποιητή και έναν αποκωδικοποιητή, και τα δύο είναι απλά νευρωνικά δίκτυα προώθησης. Ο κωδικοποιητής συμπιέζει τα δεδομένα εισόδου σε μια αναπαράσταση χαμηλότερης διάστασης (λανθάνων κώδικας) και ο αποκωδικοποιητής προσπαθεί να ανακατασκευάσει την αρχική είσοδο από αυτήν την αναπαράσταση.
2. Αραιοί αυτοκωδικοποιητές: ενθαρρύνουν τον λανθάνοντα κώδικα να είναι αραιός, δηλαδή να έχει μόνο έναν μικρό αριθμό ενεργών νευρώνων. Αυτό μπορεί να βοηθήσει τον αυτόματο κωδικοποιητή να μάθει πιο συμπαγείς και ερμηνεύσιμες αναπαραστάσεις των δεδομένων.
3. Πυκνοί αυτοκωδικοποιητές: Πρόκειται για αυτοκωδικοποιητές που χρησιμοποιούν πυκνά, πλήρως συνδεδεμένα στρώματα τόσο στον κωδικοποιητή όσο και στον αποκωδικοποιητή, σε αντίθεση με τα αραιά ή συνελικτικά στρώματα. Είναι κατάλληλα για μικρά σύνολα δεδομένων ή προβλήματα όπου τα δεδομένα εισόδου έχουν σχετικά απλή δομή.
4. Συνελικτικοί αυτοκωδικοποιητές: Αυτοί είναι αυτοκωδικοποιητές που έχουν σχεδιαστεί ειδικά για δεδομένα εικόνας. Ο κωδικοποιητής και ο αποκωδικοποιητής χρησιμοποιούν συνελικτικά στρώματα για να επωφεληθούν από τη χωρική δομή των δεδομένων, με αποτέλεσμα την πιο αποτελεσματική συμπίεση και ανακατασκευή.
5. Αυτόματοι κωδικοποιητές αποθορυβοποίησης: Αυτοί οι αυτόματοι κωδικοποιητές εκπαιδεύονται σε κατεστραμμένες εκδόσεις των δεδομένων εισόδου, με στόχο την ανακατασκευή των αρχικών, μη κατεστραμμένων δεδομένων. Αυτό βοηθά τον αυτόματο κωδικοποιητή να μάθει πιο ισχυρές αναπαραστάσεις των δεδομένων που είναι λιγότερο ευαίσθητα σε θόρυβο ή άλλες παραμορφώσεις.
6. Variational autoencoder (VAEs): Πρόκειται για αυτοκωδικοποιητές που έχουν επεκταθεί σε πιθανοτικά μοντέλα. Τα VAE εκπαιδεύονται όχι μόνο να ελαχιστοποιούν την απώλεια ανακατασκευής, αλλά και να μαθαίνουν μια κατανομή στους λανθάνοντες κωδικούς που είναι κοντά σε μια προηγούμενη διανομή. Αυτό επιτρέπει τη δημιουργία νέων δειγμάτων δεδομένων που είναι παρόμοια με τα δεδομένα εισόδου.

Καθένας από αυτούς τους τύπους αυτόματων κωδικοποιητών έχει τα δικά του δυνατά και αδύνατα σημεία και η επιλογή του τύπου που θα χρησιμοποιηθεί θα εξαρτηθεί από το συγκεκριμένο πρόβλημα και τα δεδομένα που υπάρχουν. Για παράδειγμα, οι συνελικτικοί αυτόματοι κωδικοποιητές είναι κατάλληλοι για δεδομένα εικόνας, ενώ οι denoising autoencoders μπορούν να βοηθήσουν στο χειρισμό θορυβωδών ή κατεστραμμένων δεδομένων. Τα VAE είναι χρήσιμα για εργασίες παραγωγής, όπου είναι επιθυμητό να δειγματιστούν νέα δεδομένα από μια ήδη γνωστή κατανομή.

2.7 Μηχανές Boltzmann

Μια μηχανή Boltzmann είναι ένας τύπος παραγωγικού στοχαστικού τεχνητού νευρωνικού δικτύου (generative stochastic artificial neural network) που μπορεί να χρησιμοποιηθεί για μια ποικιλία εργασιών, συμπεριλαμβανομένης της μείωσης διαστάσεων, της ταξινόμησης και της παραγωγικής μοντελοποίησης. Πήρε το όνομά της από τον φυσικό Ludwig Boltzmann.

Μια μηχανή Boltzmann αποτελείται από ένα σύνολο νευρώνων δυαδικής αξίας που συνδέονται μεταξύ τους μέσω συμμετρικών σταθμισμένων συνδέσεων. Οι νευρώνες μπορεί να βρίσκονται σε μία από τις δύο καταστάσεις: «ενεργοποιημένοι» ή «απενεργοποιημένοι». Η κατάσταση ενός νευρώνα καθορίζεται από τις καταστάσεις των νευρώνων με τους οποίους συνδέεται, καθώς και από τη δική του εσωτερική ενέργεια, η οποία είναι συνάρτηση των βαρών των συνδέσεων με άλλους νευρώνες.



Εικόνα 2-11. Γράφος που δείχνει πώς μπορεί να δομηθεί μια μηχανή Boltzmann

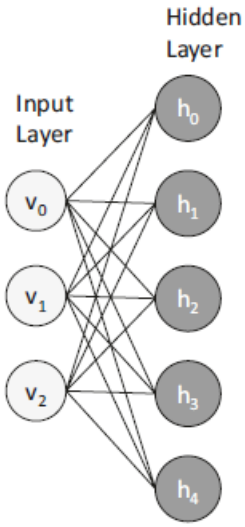
Η μηχανή Boltzmann μπορεί να εκπαιδευτεί χρησιμοποιώντας έναν αλγόριθμο αντίθεσης απόκλισης (contrastive divergence), ο οποίος ενημερώνει τα βάρη των συνδέσεων έτσι ώστε να μεγιστοποιήσει την πιθανότητα των δεδομένων εκπαίδευσης. Αυτό επιτρέπει στη μηχανή Boltzmann να μάθει μια πιθανολογική κατανομή των καταστάσεων των νευρώνων που είναι αντιπροσωπευτική των δεδομένων εκπαίδευσης.

Μόλις εκπαιδευτεί, μια μηχανή Boltzmann μπορεί να χρησιμοποιηθεί για τη δημιουργία νέων, συνθετικών δειγμάτων δεδομένων με δειγματοληψία από τη γνωστή κατανομή των καταστάσεων των νευρώνων. Αυτό καθιστά τις μηχανές Boltzmann ένα ισχυρό εργαλείο για παραγωγική μοντελοποίηση, καθώς και για εργασίες μάθησης χωρίς επίβλεψη, όπου ο στόχος είναι να βρεθεί μια συμπαγής, χαμηλών διαστάσεων αναπαράσταση των δεδομένων. Ωστόσο, οι μηχανές Boltzmann δεν είναι απαραίτητα τόσο πρακτικές και εμφανίζουν προβλήματα όταν το δίκτυο αυξάνεται σε μέγεθος. Συγκεκριμένες παραλλαγές της μηχανής Boltzmann όπως οι restricted Boltzmann machines (RBM), οι deep Boltzmann machines (DBM) και τα deep belief networks (DBN) είναι πολύ πιο κατάλληλες και πρακτικές, αν και είναι λίγο ξεπερασμένες.

2.7.1 Restricted Boltzmann Machines (RBM)

Η Περιορισμένη Μηχανή Boltzmann (RBM) είναι ένας τύπος παραγωγικού (generative) στοχαστικού τεχνητού νευρωνικού δικτύου που προέρχεται από τη μηχανή Boltzmann. Οι RBM είναι απλούστερες και πιο αποτελεσματικές από τις πλήρεις μηχανές Boltzmann, καθιστώντας τις κατάλληλες για εργασίες μηχανικής εκμάθησης μεγάλης κλίμακας.

Όπως οι μηχανές Boltzmann, οι RBM αποτελούνται από νευρώνες δυαδικών τιμών που συνδέονται μεταξύ τους μέσω σταθμισμένων συνδέσεων. Ωστόσο, οι RBM είναι «περιορισμένες» καθώς οι συνδέσεις μεταξύ των νευρώνων είναι μη κατευθυνόμενες και σχηματίζουν ένα διμερές γράφημα, που σημαίνει ότι δεν υπάρχουν συνδέσεις μεταξύ των νευρώνων σε ένα επίπεδο. Αυτό σημαίνει ότι οι νευρώνες σε ένα επίπεδο συνδέονται μόνο με τους νευρώνες στο άλλο επίπεδο και όχι μεταξύ τους.



Εικόνα 2-12. Απεικόνιση βασικής RBM

Ως αποτέλεσμα, οι διαδικασίες εκπαίδευσης και εξαγωγής συμπερασμάτων για τις RBM είναι πιο αποδοτικές υπολογιστικά από ό,τι για πλήρεις μηχανές Boltzmann. Οι RBM μπορούν να εκπαιδευτούν χρησιμοποιώντας μεθόδους βελτιστοποίησης που βασίζονται σε κλίση, όπως η απόκλιση αντίθεσης (contrastive divergence) ή η επίμονη αντιθετική απόκλιση (persistent contrastive divergence), που καθιστούν δυνατή την εκπαίδευση μεγάλης κλίμακας RBM σε μεγάλα σύνολα δεδομένων. Αυτοί οι αλγόριθμοι χρησιμοποιούν και οι δύο αλυσίδες Markov για να βοηθήσουν τον αλγόριθμο εκπαίδευσης να καθορίσει σε ποια κατεύθυνση θα εκτελέσει τους υπολογισμούς της κλίσης, αλλά διαφέρουν και έχουν τα πλεονεκτήματα και τα μειονεκτήματά τους. Το PCD μπορεί να λάβει καλύτερα δείγματα δεδομένων και να εξερευνήσει καλύτερα τον χώρο εισόδου, αλλά το CD είναι καλύτερο στην εξαγωγή χαρακτηριστικών.

Συνοπτικά, οι RBM είναι μια απλοποιημένη μορφή μηχανών Boltzmann που διατηρούν τα βασικά χαρακτηριστικά του αρχικού μοντέλου, αλλά είναι υπολογιστικά πιο αποτελεσματικές. Η κύρια διαφορά μεταξύ των RBM και των μηχανών Boltzmann είναι ότι τα RBM έχουν μια περιορισμένη δομή με μόνο μη κατευθυνόμενες συνδέσεις μεταξύ των νευρώνων, ενώ οι μηχανές Boltzmann έχουν πλήρως συνδεδεμένες, συμμετρικές συνδέσεις μεταξύ όλων των νευρώνων.

2.8 Long short-term memory models (LSTM)

Χρονοσειρές

Οι χρονοσειρές αποτελούνται από δεδομένα που συλλέγονται με την πάροδο του χρόνου, συχνά σε τακτά χρονικά διαστήματα, τα οποία μπορούν να αναλυθούν για τον εντοπισμό προτύπων, τάσεων και προβλέψεων για μελλοντικά γεγονότα. Η ανάλυση χρονοσειρών είναι μια στατιστική προσέγγιση για τη μελέτη δεδομένων χρονοσειρών και περιλαμβάνει τεχνικές όπως ανάλυση τάσεων, εποχιακή ανάλυση και μοντελοποίηση προβλέψεων. Βοηθά στην κατανόηση της υποκείμενης συμπεριφοράς των δεδομένων, στον εντοπισμό βασικών χαρακτηριστικών, την ανίχνευση ανωμαλιών και στην πραγματοποίηση προβλέψεων σχετικά με τις μελλοντικές τιμές της χρονοσειράς.

2.8.1 Επαναλαμβανόμενα Νευρωνικά Δίκτυα (Recurrent Neural Networks - RNN)

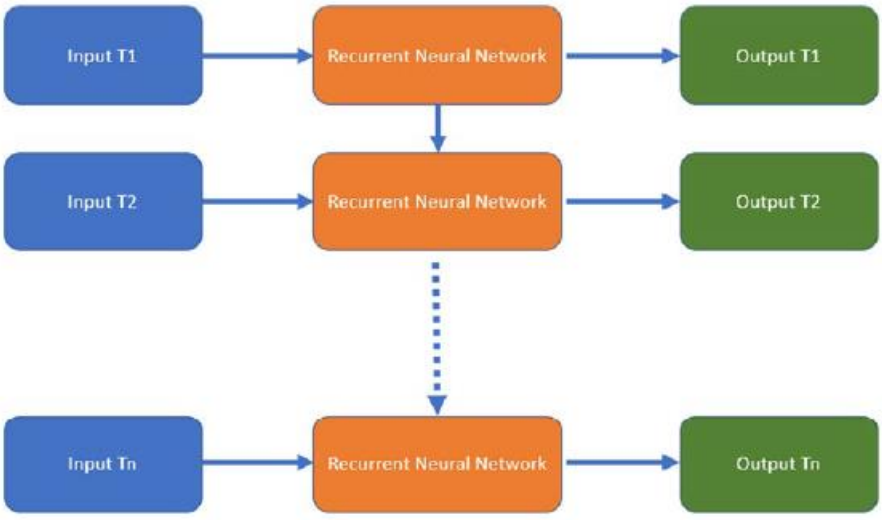
Τα επαναλαμβανόμενα νευρωνικά δίκτυα (RNN) είναι ένας τύπος νευρωνικού δικτύου που έχει σχεδιαστεί για να χειρίζεται διαδοχικά δεδομένα, όπου η έξοδος από το ένα βήμα χρησιμοποιείται ως είσοδος για το επόμενο (Sridhar & Suman, 2019). Τα RNN έχουν ένα στοιχείο μνήμης, το οποίο τους επιτρέπει να διατηρούν πληροφορίες από προηγούμενα χρονικά βήματα και να το χρησιμοποιούν για να κάνουν προβλέψεις σχετικά με τα μελλοντικά χρονικά βήματα.

Εικόνα 2-13. Αναπαράσταση υψηλού επιπέδου νευρωνικού δικτύου

Στο πλαίσιο της ανάλυσης χρονοσειρών, τα RNN είναι ιδιαίτερα χρήσιμα επειδή μπορούν να καταλάβουν μοτίβα στα δεδομένα που μπορεί να μην είναι εμφανή από ένα μόνο χρονικό βήμα. Με το να επεξεργάζοντα τα δεδομένα εισόδου ως ακολουθία, τα RNN μπορούν να αναγνωρίσουν μακροπρόθεσμες εξαρτήσεις, τάσεις και μοτίβα στα δεδομένα, τα οποία μπορούν να χρησιμοποιηθούν για να γίνουν πιο ακριβείς προβλέψεις.

Τα RNN επίσης είναι αποτελεσματικά για την διαχείριση δεδομένων χρονοσειρών με πολλαπλές εξαρτήσεις και πολλαπλές εποχικότητες. Για παράδειγμα, μπορούν να χρησιμοποιηθούν για την ανάλυση χρηματοοικονομικών δεδομένων για τον εντοπισμό προτύπων και την πραγματοποίηση προβλέψεων σχετικά με τις τιμές των μετοχών.

Συνοπτικά, τα RNN χρησιμοποιούνται ευρέως στην ανάλυση χρονοσειρών επειδή μπορούν να χειριστούν διαδοχικά δεδομένα, να καταγράψουν μοτίβα και εξαρτήσεις στα δεδομένα και να κάνουν ακριβείς προβλέψεις με βάση τα ιστορικά δεδομένα.



Εικόνα 2-14. Αναπαράσταση υψηλού επιπέδου επαναλμαβνόμενου νευρωνικού δικτύου

Λειτουργία RNN:

1. Είσοδος (Input): Η είσοδος σε ένα RNN σε κάθε χρονικό βήμα είναι ένα διάνυσμα χαρακτηριστικών που αντιπροσωπεύουν τα δεδομένα εκείνη τη στιγμή.
2. Κρυφό επίπεδο (Hidden layer): Το κρυφό επίπεδο είναι όπου το RNN αποθηκεύει πληροφορίες σχετικά με την ακολουθία. Χρησιμοποιεί ένα σύνολο συναρτήσεων ενεργοποίησης για την επεξεργασία των δεδομένων εισόδου και τη δημιουργία μιας εξόδου που τροφοδοτείται πίσω στο δίκτυο ως είσοδος για το επόμενο χρονικό βήμα.
3. Επίπεδο εξόδου (Output laye): Το επίπεδο εξόδου χρησιμοποιεί τις πληροφορίες που είναι αποθηκευμένες στο κρυφό επίπεδο για να δημιουργήσει μια πρόβλεψη ή ταξινόμηση.
4. Επαναλαμβανόμενες συνδέσεις (Recurrent connections): Το βασικό χαρακτηριστικό ενός RNN είναι οι επαναλαμβανόμενες συνδέσεις μεταξύ του κρυφού στρώματος και του εαυτού του. Αυτές οι συνδέσεις επιτρέπουν στο δίκτυο να διατηρεί πληροφορίες από προηγούμενα χρονικά βήματα και να τις χρησιμοποιεί για να κάνει προβλέψεις σχετικά με τα μελλοντικά χρονικά βήματα.
5. Unrolling: Ένα RNN μπορεί να ξετυλιχτεί με την πάροδο του χρόνου, που σημαίνει ότι μπορεί να αναπαρασταθεί ως μια σειρά από συνδεδεμένες επαναλαμβανόμενες μονάδες, κάθε μια από τις οποίες επεξεργάζεται ένα χρονικό βήμα των δεδομένων εισόδου.

Κατά τη διάρκεια της προπόνησης, το RNN χρησιμοποιεί backpropagation για να προσαρμόσει τα βάρη στο κρυφό στρώμα με βάση το σφάλμα πρόβλεψης σε κάθε χρονικό βήμα. Αυτό επιτρέπει στο δίκτυο να μάθει τις σχέσεις μεταξύ των δεδομένων εισόδου και εξόδου με την πάροδο του χρόνου και να κάνει προβλέψεις για μελλοντικά δεδομένα με βάση ιστορικά δεδομένα.

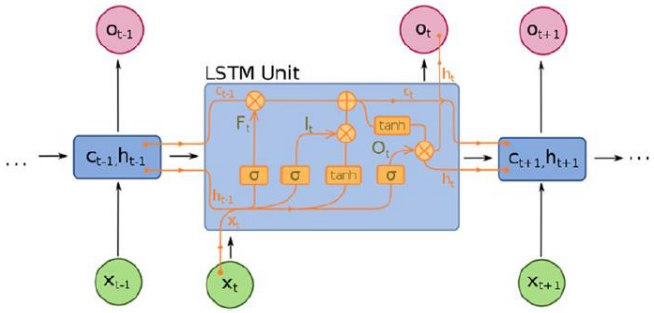
Ένα από τα προβλήματα των RNN είναι ότι καθώς προσπαθούμε να μοντελοποιήσουμε εξαρτήσεις μεταξύ τιμών ακολουθίας που διαχωρίζονται από έναν σημαντικό αριθμό άλλων τιμών, οι διαβαθμίσεις του χρονικού βήματος T εξαρτώνται από τις διαβαθμίσεις στο T-1, τις διαβαθμίσεις στο T-2, και ούτω καθεξής. Αυτό οδηγεί στη συνεισφορά της παλαιότερης κλίσης να γίνεται όλο και μικρότερη καθώς κινούμαστε κατά μήκος των χρονικών βημάτων όπου η αλυσίδα των κλίσεων γίνεται μεγαλύτερη και μεγαλύτερη. Αυτό είναι αυτό που είναι γνωστό ως πρόβλημα εξαφάνισης κλίσης (vanishing gradient problem). Ως αποτέλεσμα το RNN γίνεται προκατειλημμένο, επηρεαζόμενο μόνο από βραχυπρόθεσμα σημεία δεδομένων. Τα δίκτυα LSTM είναι ένας τρόπος επίλυσης αυτού του προβλήματος των RNN.

2.8.2 LSTM

Το Long Short-Term Memory (LSTM) είναι ένας τύπος επαναλαμβανόμενου νευρωνικού δικτύου (RNN) που έχει σχεδιαστεί ειδικά για να χειρίζεται μακροπρόθεσμες εξαρτήσεις σε διαδοχικά δεδομένα. Τα δίκτυα LSTM εισήχθησαν για την αντιμετώπιση του προβλήματος της εξαφάνισης κλίσης (vanishing gradient problem), το οποίο είναι ένα σύνηθες πρόβλημα στα παραδοσιακά RNN όταν εκπαιδεύονται σε μεγάλες ακολουθίες.

Τα LSTM λειτουργούν διατηρώντας ένα κελί μνήμης (memory cell), το οποίο είναι ένα διάνυσμα που ενημερώνεται σε κάθε χρονικό βήμα και χρησιμοποιείται για τη διατήρηση πληροφοριών από προηγούμενα χρονικά βήματα. Το κελί μνήμης ελέγχεται από τρεις πύλες: την πύλη εισόδου (input gate), την forget gate και την πύλη εξόδου (output gate). Αυτές οι πύλες χρησιμοποιούνται για τη ρύθμιση της ροής πληροφοριών προς και έξω από την κυψέλη μνήμης, επιτρέποντας στο LSTM να ελέγχει τις πληροφορίες που αποθηκεύει και να διατηρεί σημαντικές πληροφορίες για μεγαλύτερες περιόδους.

Η δομή ενός δικτύου LSTM αποτελείται από ένα επίπεδο εισόδου, ένα σύνολο μονάδων LSTM και ένα επίπεδο εξόδου. Κατά τη διάρκεια της εκπαίδευσης, το δίκτυο χρησιμοποιεί backpropagation για να προσαρμόσει τα βάρη στις μονάδες LSTM με βάση το σφάλμα πρόβλεψης σε κάθε χρονικό βήμα. Αυτό επιτρέπει στο δίκτυο να μάθει τις σχέσεις μεταξύ των δεδομένων εισόδου και εξόδου με την πάροδο του χρόνου και να κάνει προβλέψεις για μελλοντικά δεδομένα βάσει ιστορικών δεδομένων.



Εικόνα 2-15. Ένα λεπτομερές δίκτυο LSTM

Στα δίκτυα Long Short-Term Memory (LSTM), οι συναρτήσεις ενεργοποίησης χρησιμοποιούνται (activation functions) για την εισαγωγή μη γραμμικότητας στο δίκτυο και επιτρέπουν στο μοντέλο να καταγράφει σύνθετες σχέσεις στα δεδομένα. Οι συναρτήσεις ενεργοποίησης που χρησιμοποιούνται στα LSTM μπορούν να έχουν σημαντικό αντίκτυπο στην απόδοση του δικτύου και επιλέγονται προσεκτικά με βάση τη φύση των δεδομένων και την επιθυμητή έξοδο. Συχνά για δεδομένα χρονοσειρών χρησιμοποιούμε ως συνάρτηση ενεργοποίησης την tanh, η οποία μπορεί να διατηρήσει πληροφορίες για μεγαλύτερο χρνικό εύρος πριν πάει στο μηδέν.

Τα δίκτυα Long Short-Term Memory (LSTM) μπορούν να χρησιμοποιηθούν στην ανίχνευση ανωμαλιών εκπαιδεύοντας το δίκτυο σχετικά με την κανονική συμπεριφορά σε μια χρονολογική σειρά και στη συνέχεια χρησιμοποιώντας το εκπαιδευμένο μοντέλο για τον εντοπισμό ασυνήθιστων μοτίβων που αποκλίνουν από την κανονική συμπεριφορά.

Πιο συγκεκριμένα τα βήματα που ακολουθούνται είναι:

1. Προετοιμασία δεδομένων: Το πρώτο βήμα είναι η προετοιμασία των δεδομένων χρονοσειρών για την εκπαίδευση του δικτύου LSTM. Αυτό συνήθως περιλαμβάνει την κανονικοποίηση (normalizing) των δεδομένων, τη μετατροπή τους σε ακολουθίες και τον διαχωρισμό τους σε σετ εκπαίδευσης (training set) και δοκιμών (test set).
2. Εκπαίδευση μοντέλου: Το δίκτυο LSTM εκπαιδεύεται στην κανονική συμπεριφορά στα δεδομένα χρονοσειρών ελαχιστοποιώντας το σφάλμα πρόβλεψης μεταξύ των πραγματικών και των προβλεπόμενων τιμών. Κατά τη διάρκεια της εκπαίδευσης, το LSTM μαθαίνει να κωδικοποιεί τα μοτίβα και τις σχέσεις στα κανονικά δεδομένα.
3. Ανίχνευση ανωμαλιών: Μετά την εκπαίδευση, το δίκτυο LSTM χρησιμοποιείται για την ανίχνευση ανωμαλιών σε δεδομένα νέων χρονοσειρών. Το δίκτυο δημιουργεί μια πρόβλεψη για κάθε χρονικό βήμα με βάση τα δεδομένα εισόδου και την κρυφή κατάσταση των μονάδων LSTM. Η απόκλιση μεταξύ των πραγματικών και των προβλεπόμενων τιμών μπορεί να χρησιμοποιηθεί για τον εντοπισμό ασυνήθιστων μοτίβων που αποκλίνουν από την κανονική συμπεριφορά.

4. Αξιολόγηση: Η απόδοση του συστήματος ανίχνευσης ανωμαλιών που βασίζεται σε LSTM μπορεί να αξιολογηθεί χρησιμοποιώντας διάφορες μετρήσεις, όπως accuracy, precision, ανάκληση (recall) και βαθμολογία F1 (F1 score), για να προσδιοριστεί η ικανότητά του να ανιχνεύει ανωμαλίες.

Συνοπτικά, τα LSTM μπορούν να χρησιμοποιηθούν στην ανίχνευση ανωμαλιών εκπαιδεύοντας το δίκτυο σε κανονική συμπεριφορά σε μια χρονολογική σειρά, χρησιμοποιώντας το εκπαιδευμένο μοντέλο για τον εντοπισμό ασυνήθιστων μοτίβων που αποκλίνουν από την κανονική συμπεριφορά και αξιολογώντας την απόδοση του συστήματος ανίχνευσης ανωμαλιών. Τα LSTMs έχουν αποδειχθεί αποτελεσματικά στην ανίχνευση ανωμαλιών σε διάφορους τύπους δεδομένων χρονοσειρών, όπως η κυκλοφορία δικτύου και τα οικονομικά δεδομένα.

2.9 Temporal Convolutional Networks

Τα προσωρινά συνελικτικά δίκτυα (TCN) είναι ένας τύπος αρχιτεκτονικής που χρησιμοποιεί μονοδιάστατα συνελικτικά επίπεδα για την επεξεργασία διαδοχικών δεδομένων. (Jukan & Ganchev, 2017) Σε αντίθεση με τα παραδοσιακά συνελικτικά δίκτυα, τα TCN χρησιμοποιούν αιτιώδεις συνελίξεις, που σημαίνει ότι πληροφορίες από μελλοντικά χρονικά βήματα δεν χρησιμοποιούνται για την επεξεργασία πληροφοριών από προηγούμενα χρονικά βήματα. Αυτό σημαίνει ότι το μοντέλο επεξεργάζεται μόνο δεδομένα που προχωρούν στο χρόνο, καθιστώντας το λιγότερο επιρρεπές σε σφάλματα στην επεξεργασία ακολουθίας.

Ένα από τα προβλήματα με τα επαναλαμβανόμενα νευρωνικά δίκτυα στο πλαίσιο της γλωσσικής μετάφρασης είναι ότι διαβάζει προτάσεις από αριστερά προς τα δεξιά, κάνοντας λάθος μετάφραση σε ορισμένες περιπτώσεις όπου η σειρά των λέξεων έχει αλλάξει για να δοθεί έμφαση. Για να αντιμετωπιστεί αυτό, χρησιμοποιούνται αμφίδρομοι κωδικοποιητές, αλλά αυτό απαιτεί την εξέταση μελλοντικών πληροφοριών στο παρόν. Ωστόσο, τα TCN δεν αντιμετωπίζουν αυτό το πρόβλημα, καθώς δεν βασίζονται σε πληροφορίες από προηγούμενα χρονικά βήματα και μπορούν να διατηρήσουν την αιτιότητά τους.

Επιπλέον, τα TCN έχουν τη δυνατότητα να αντιστοιχίσουν μια ακολουθία εισόδου οποιουδήποτε μήκους σε μια ακολουθία εξόδου με το ίδιο μήκος, ακριβώς όπως τα RNN. Αυτό τα καθιστά μια βιώσιμη εναλλακτική λύση στα RNN για την επεξεργασία διαδοχικών δεδομένων και την επίλυση εργασιών όπως η ταξινόμηση και η πρόβλεψη ακολουθιών.

Τα χρονικά συνελικτικά δίκτυα (TCN) φαίνεται να είναι μια εξαιρετική εναλλακτική στα RNN.

Γενικά τα πλεονεκτήματα των TCN, ειδικά λαμβάνοντας υπόψη τα RNN είναι:

1. Παράλληλοι υπολογισμοί: Τα συνελικτικά δίκτυα συνδυάζονται καλά με την εκπαίδευση μέσω GPU, ιδιαίτερα επειδή οι υπολογισμοί των συνελικτικών επιπέδων ταιριάζουν καλά στη δομή των GPU, οι οποίες είναι ρυθμισμένες να πραγματοποιούν υπολογισμούς πινάκων που αποτελούν μέρος της επεξεργασίας γραφικών. Εξαιτίας αυτού, τα TCN μπορούν να εκπαιδεύονται πολύ πιο γρήγορα από τα RNN.
2. Ευελιξία: Τα TCN μπορούν να αλλάξουν το μέγεθος εισόδου, το μέγεθος φίλτρου, να αυξήσουν τους παράγοντες διαστολής (dilation factors), να χρησιμοποιήσουν περισσότερα επίπεδα κ.λπ. προκειμένου να εφαρμοστούν εύκολα σε διάφορους τομείς.

3. Consistent gradients (Συνεπείς κλίσεις): Επειδή τα TCN αποτελούνται από συνελικτικά στρώματα, πραγματοποιούν διαφορετικό (backpropagation) από τα RNN, και έτσι αποθηκεύονται όλες οι κλίσεις. Τα RNN παρουσιάζουν ένα πρόβλημα που ονομάζεται εξαφάνιση κλίσης (vanishing gradient), δηλαδή μερικές φορές η υπολογιζόμενη κλίση είναι είτε εξαιρετικά μεγάλη είτε εξαιρετικά μικρή, με αποτέλεσμα το αναπροσαρμοσμένο βάρος να είναι πολύ ακραίο σε σχέση με μια αλλαγή ή να είναι μια σχετικά ανύπαρκτη αλλαγή. Για να καταπολεμηθεί αυτό, αναπτύχθηκαν τύποι RNN όπως τα LSTM.
4. Ελαφρύτερα στη μνήμη: Τα LSTM αποθηκεύουν πληροφορίες στις πύλες κυψέλης τους, έτσι εάν η ακολουθία εισόδου είναι μεγάλη, χρησιμοποιείται πολλή μνήμη. Συγκριτικά, τα TCN είναι σχετικά απλά επειδή αποτελούνται από πολλά επίπεδα που μοιράζονται όλα τα δικά τους αντίστοιχα φίλτρα. Σε σύγκριση με τα LSTM, τα TCN είναι πολύ πιο ελαφριά όσον αφορά τη χρήση της μνήμης τους.

Μειονεκτήματα των TCN σε σύγκριση με τα RNN:

1. Χρήση μνήμης κατά τη λειτουργία αξιολόγησης: Τα RNN χρειάζεται μόνο να γνωρίζουν κάποια είσοδο xt για να δημιουργήσουν μια πρόβλεψη, καθώς διατηρούν μια περίληψη όλων όσων έμαθαν μέσω των διανυσμάτων κρυφής κατάστασης. Συγκριτικά, τα TCN χρειάζονται ξανά ολόκληρη τη χρονοσειρά μέχρι το τρέχον σημείο για να κάνουν μια αξιολόγηση, οδηγώντας σε δυνητικά υψηλότερη χρήση μνήμης από ένα RNN.
2. Προβλήματα με τη μεταφορά μάθησης (transfer learning): Αρχικά, ας ορίσουμε τι είναι το transfer learning. Είναι όταν ένα μοντέλο έχει εκπαιδευτεί για μια συγκεκριμένη εργασία (ταξινόμηση οχημάτων, για παράδειγμα), και αφαιρείται το τελευταίο στρώμα και επανεκπαιδεύεται πλήρως, έτσι ώστε το μοντέλο να μπορεί να χρησιμοποιηθεί για μια νέα εργασία ταξινόμησης (ταξινόμηση ζώων, για παράδειγμα).

   Στην όραση υπολογιστών, υπάρχουν μερικά πραγματικά ισχυρά μοντέλα, όπως το μοντέλο inception-v3, που έχουν εκπαιδευτεί σε ισχυρές GPU για αρκετό καιρό, προκειμένου να επιτύχουν τις τωρινές επιδόσεις τους. Αντί να εκπαιδεύουμε το δικό μας CNN από την αρχή (και οι περισσότεροι από εμάς δεν έχουμε το υλικό GPU ή το χρόνο για να αφιερώσουμε σε μακροχρόνια εκπαίδευση σε ένα εξαιρετικά βαθύ μοντέλο όπως το inception-v3), μπορούμε απλά να πάρουμε το inception-v3, για παράδειγμα, το οποίο είναι πολύ καλό στην εξαγωγή χαρακτηριστικών από εικόνες και να το εκπαιδεύσουμε στο να συσχετίζει τις δυνατότητες που εξάγει με ένα εντελώς νέο σύνολο κλάσεων. Αυτή η διαδικασία απαιτεί πολύ λιγότερο χρόνο, καθώς τα βάρη σε ολόκληρο το δίκτυο είναι ήδη καλά βελτιστοποιημένα, επομένως ασχολούμαστε μόνο με την εύρεση των βέλτιστων βαρών για τα επίπεδα που επανεκπαιδεύουμε.

   Γι' αυτό η μεταφορά μάθησης είναι μια τόσο πολύτιμη διαδικασία. Μας επιτρέπει να πάρουμε ένα προεκπαιδευμένο μοντέλο υψηλής απόδοσης και απλώς να επανεκπαιδεύσουμε τα τελευταία στρώματα με το υλικό μας και να διδάξουμε στο μοντέλο μια νέα εργασία ταξινόμησης (για CNN).

   Επιστρέφοντας στα TCN, το μοντέλο μπορεί να χρειαστεί να θυμάται διάφορα επίπεδα ιστορικού ακολουθίας προκειμένου να κάνει προβλέψεις. Εάν το μοντέλο δεν χρειαζόταν να λάβει τόσο πολύ ιστορικό στην αρχική χρήση του για να κάνει προβλέψεις, αλλά για τη νέα χρήση του έπρεπε να λάβει περισσότερο/λιγότερο ιστορικό, αυτό προκαλεί προβλήματα και μπορεί να οδηγήσει το μοντέλο σε κακή απόδοση .

Συμπερασματικά, τα TCN και τα RNN έχουν τα πλεονεκτήματα και τα μειονεκτήματά τους. Τα TCN είναι υπολογιστικά αποδοτικά, ανθεκτικά στο vanishing gradient problem και απαιτούν εισόδους σταθερού μήκους, ενώ τα RNN είναι πιο ευέλικτα με εισόδους μεταβλητού μήκους και πιο ερμηνεύσιμα. Η επιλογή

μεταξύ TCN και RNN εξαρτάται από τη φύση των δεδομένων και το καλύτερο μοντέλο μπορεί να διαφέρει ανάλογα με την εφαρμογή.

2.10 Ανίχνευση ανωμαλιών με forecasting

Η ανίχνευση ανωμαλιών περιλαμβάνει τον εντοπισμό προτύπων ή συμβάντων που αποκλίνουν σημαντικά από την αναμενόμενη ή κανονική συμπεριφορά σε ένα σύνολο δεδομένων. Η πρόβλεψη, από την άλλη πλευρά, είναι η διαδικασία να γίνονται προβλέψεις για μελλοντικές αξίες ή γεγονότα με βάση ιστορικά δεδομένα. Η ανίχνευση ανωμαλιών με χρήση της πρόβλεψης συνδυάζει αυτές τις δύο έννοιες για τον εντοπισμό ανωμαλιών συγκρίνοντας τις παρατηρούμενες τιμές με τις προβλεπόμενες τιμές.

Ακολουθεί μια γενική προσέγγιση για τον εντοπισμό ανωμαλιών χρησιμοποιώντας πρόβλεψη:

Προεπεξεργασία δεδομένων: Ξεκινάμε με τη συλλογή και την προεπεξεργασία των δεδομένων που σχετίζονται με το πρόβλημα που αντιμετωπίζουμε.

Ανάλυση χρονοσειρών: Εάν τα δεδομένα μας έχουν ένα χρονικό στοιχείο, όπως μια χρονική σήμανση που σχετίζεται με κάθε παρατήρηση, εφαρμόζουμε τεχνικές ανάλυσης χρονοσειρών. Αυτό περιλαμβάνει την εξέταση των μοτίβων, των τάσεων και της εποχικότητας στα δεδομένα για να αποκτήσουμε πληροφορίες για τη συμπεριφορά τους με την πάροδο του χρόνου. Βοηθά στην κατανόηση των φυσιολογικών προτύπων και στον εντοπισμό αποκλίσεων από αυτά.

Μοντέλο πρόβλεψης: Στη συνέχεια, πρέπει να δημιουργήσουμε ένα μοντέλο πρόβλεψης για να προβλέψουμε μελλοντικές τιμές βάσει ιστορικών δεδομένων. Υπάρχουν διάφορες διαθέσιμες τεχνικές πρόβλεψης, όπως ARIMA (AutoRegressive Integrated Moving Average), μέθοδοι εκθετικής εξομάλυνσης ή αλγόριθμοι μηχανικής μάθησης όπως γραμμική παλινδρόμηση, δέντρα αποφάσεων ή νευρωνικά δίκτυα. Η επιλογή του μοντέλου εξαρτάται από τη φύση των δεδομένων και την εργασία πρόβλεψης.

Εκπαίδευση και επικύρωση (train and test): Διαχωρίζουμε τα διαθέσιμα δεδομένα σε ένα σύνολο εκπαίδευσης και ένα σύνολο επικύρωσης. Χρησιμοποιήστε το σετ εκπαίδευσης για να εκπαιδεύσετε το μοντέλο πρόβλεψης, προσαρμόζοντας τις παραμέτρους του για να ελαχιστοποιήσετε τη διαφορά μεταξύ των προβλεπόμενων και των πραγματικών τιμών. Ελέγχουμε την απόδοση του μοντέλου χρησιμοποιώντας το σύνολο επικύρωσης συγκρίνοντας τις προβλεπόμενες τιμές με τις πραγματικές τιμές.

Ανίχνευση ανωμαλιών: Μόλις εκπαιδευτεί και επικυρωθεί το μοντέλο πρόβλεψης, το χρησιμοποιούμε για να προβλέψουμε μελλοντικές τιμές. Συγκρίνουμε αυτές τις προβλεπόμενες τιμές με τις παρατηρούμενες τιμές σε πραγματικό χρόνο. Εάν οι παρατηρούμενες τιμές αποκλίνουν σημαντικά από τις προβλεπόμενες τιμές, υποδηλώνουν την παρουσία ανωμαλίας.

Κατώφλι ή Στατιστικές Μέθοδοι: Για να προσδιορίσουμε το όριο για τον εντοπισμό ανωμαλιών, χρησιμοποιούμε διάφορες προσεγγίσεις. Μια κοινή μέθοδος είναι ο καθορισμός ενός ορίου με βάση τις στατιστικές ιδιότητες των δεδομένων, όπως ο ορισμός των ανωμαλιών ως τιμών που βρίσκονται έξω από έναν ορισμένο αριθμό τυπικών αποκλίσεων από τον μέσο όρο. Εναλλακτικά, χρησιμοποιούμε ιστορικά δεδομένα για να υπολογίσουμε τα διαστήματα πρόβλεψης και να προσδιορίσουμε ανωμαλίες εάν οι παρατηρούμενες τιμές βρίσκονται εκτός αυτών των διαστημάτων.

Warning/Action: Όταν εντοπίζουμε μια ανωμαλία, μπορούμε να δημιουργήσουμε μια ειδοποίηση ή να πραγματοποιήσουμε μια συγκεκριμένη ενέργεια ανάλογα με το περιβάλλον. Για παράδειγμα, σε ένα

σύστημα κυβερνοασφάλειας, μια ανωμαλία που υποδεικνύει μια πιθανή εισβολή μπορεί να προκαλέσει μια ειδοποίηση στον διαχειριστή του συστήματος ή μια αυτοματοποιημένη απάντηση για την αντιμετώπιση της απειλής.

Αξίζει να σημειωθεί ότι η ανίχνευση ανωμαλιών με χρήση προβλέψεων δεν είναι αλάνθαστη μέθοδος και απαιτεί προσεκτική εξέταση των δεδομένων, επιλογή μοντέλου πρόβλεψης και προσδιορισμό των κατάλληλων ορίων. Είναι σημαντικό να παρακολουθείτε και να αξιολογείτε συνεχώς την απόδοση του συστήματος για να διασφαλίζεται η αποτελεσματικότητά του στον ακριβή εντοπισμό ανωμαλιών. Από τις μεθόδους, τις οποίες αναλύσαμε προηγουμένως για ανίχνευση ανωμαλιών με forecasting μπορούν να χρησιμοπιηθούν το LSTM και το TCN. Ωστόσο σημαντικό είναι να επισημάνουμε οτί εμείς στη συνέχεια θα πραγματοποιήσουμε αποκλειστικά ανίχνευση ανωμαλιών με reconstruction error.

2.11 Εφαρμογές ανίχνευσης ανώμαλων τιμών

Τομείς στους οποίους μπορεί να εφαρμοστεί ή εφαρμόζεται η ανίχνευση ανώμαλων τιμών.

- Χρηματοπιστωτικά συστήματα

Η ανίχνευση απάτης στα χρηματοπιστωτικά συστήματα είναι μια από τις πιο σημαντικές εφαρμογές ανίχνευσης ανωμαλιών με χρήση βαθιάς μάθησης. Ο στόχος αυτής της εφαρμογής είναι να εντοπίσει ασυνήθιστα μοτίβα στις οικονομικές συναλλαγές που μπορεί να υποδηλώνουν απάτη.

Ένας τρόπος για να γίνει αυτό είναι χρησιμοποιώντας μοντέλα βαθιάς μάθησης για την ανάλυση μεγάλων ποσοτήτων δεδομένων χρηματοοικονομικών συναλλαγών. Αυτά τα μοντέλα μπορούν να μάθουν να εντοπίζουν μοτίβα στα δεδομένα που είναι ενδεικτικά δόλιας δραστηριότητας. Για παράδειγμα, ένα μοντέλο βαθιάς μάθησης μπορεί να εκπαιδευτεί για να ανιχνεύει μοτίβα συναλλαγών που συμβαίνουν σε ασυνήθιστους χρόνους ή από ασυνήθιστες τοποθεσίες, που θα μπορούσαν να υποδηλώνουν δόλια δραστηριότητα.

Ένας άλλος τρόπος χρήσης της βαθιάς μάθησης για τον εντοπισμό απάτης είναι η χρήση μοντέλων δημιουργίας για τη δημιουργία συνθετικών συναλλαγών, οι οποίες μπορούν να χρησιμοποιηθούν για την εκπαίδευση ενός μοντέλου για τον εντοπισμό μη φυσιολογικών συναλλαγών. Εκπαιδεύοντας το μοντέλο σε συνθετικές συναλλαγές, μπορεί να μάθει να εντοπίζει μη κανονικές συναλλαγές που είναι παρόμοιες, αλλά όχι πανομοιότυπες με τις συνθετικές συναλλαγές στις οποίες εκπαιδεύτηκε.

Μια άλλη προσέγγιση είναι η χρήση του αυτόματου κωδικοποιητή για τον εντοπισμό ανωμαλιών. Ο αυτόματος κωδικοποιητής είναι ένας τύπος νευρωνικού δικτύου που έχει εκπαιδευτεί να αναδομεί τις εισόδους του. Μπορεί να εκπαιδευτεί σε κανονικές οικονομικές συναλλαγές και στη συνέχεια να χρησιμοποιηθεί για τον εντοπισμό συναλλαγών που δεν ταιριάζουν με τα τυπικά πρότυπα στα δεδομένα.

Συνολικά, η βαθιά μάθηση μπορεί να είναι ένα ισχυρό εργαλείο για τον εντοπισμό απάτης στα χρηματοπιστωτικά συστήματα, καθώς επιτρέπει την ανάλυση μεγάλων ποσοτήτων δεδομένων και τον εντοπισμό λεπτών προτύπων που μπορεί να υποδηλώνουν δόλια δραστηριότητα.

- Περιβάλλον

Ο εντοπισμός ασυνήθιστων μοτίβων στις δορυφορικές εικόνες για παρατήρηση της γης είναι μια άλλη εφαρμογή ανίχνευσης ανωμαλιών χρησιμοποιώντας βαθιά μάθηση. Ο στόχος αυτής της εφαρμογής είναι να ανιχνεύσει αλλαγές ή μοτίβα στην επιφάνεια της γης που μπορεί να υποδηλώνουν κάτι ασυνήθιστο ή σημαντικό.

Ένας τρόπος για να γίνει αυτό είναι με τη χρήση μοντέλων βαθιάς μάθησης για την ανάλυση μεγάλων ποσοτήτων δεδομένων δορυφορικών εικόνων. Αυτά τα μοντέλα μπορούν να μάθουν να εντοπίζουν πρότυπα στα δεδομένα που είναι ενδεικτικά συγκεκριμένων φαινομένων, όπως αλλαγές στη χρήση γης, ανάπτυξη καλλιεργειών ή αστικοποίηση. Για παράδειγμα, ένα μοντέλο βαθιάς μάθησης μπορεί να εκπαιδευτεί για να ανιχνεύει μοτίβα αποψίλωσης, αστικής επέκτασης ή υποχώρησης παγετώνων.

Ένας άλλος τρόπος χρήσης της βαθιάς εκμάθησης για παρατήρηση της γης είναι η χρήση μοντέλων για σύγκριση πολλαπλών εικόνων της ίδιας τοποθεσίας με την πάροδο του χρόνου. Αυτά τα μοντέλα μπορούν να μάθουν να αναγνωρίζουν αλλαγές στην επιφάνεια της γης, όπως η εμφάνιση ή η εξαφάνιση κτιρίων ή δρόμων.

Επιπλέον, τα μοντέλα βαθιάς μάθησης μπορούν να χρησιμοποιηθούν για την ταξινόμηση των διαφορετικών τύπων κάλυψης γης, όπως η βλάστηση, το νερό και οι αστικές περιοχές, με υψηλή ακρίβεια και αποτελεσματικότητα.

Συνολικά, η βαθιά μάθηση μπορεί να είναι ένα ισχυρό εργαλείο για τον εντοπισμό ασυνήθιστων μοτίβων σε δορυφορικές εικόνες για παρατήρηση της γης. Επιτρέπει την ανάλυση μεγάλων ποσοτήτων δεδομένων και τον εντοπισμό λεπτών μοτίβων που μπορεί να υποδηλώνουν σημαντικές αλλαγές ή φαινόμενα στην επιφάνεια της γης.

- Βιομηχανικά Συστήματα

Η παρακολούθηση βιομηχανικών συστημάτων για αστοχίες εξοπλισμού είναι μια άλλη εφαρμογή ανίχνευσης ανωμαλιών χρησιμοποιώντας βαθιά εκμάθηση. Ο στόχος αυτής της εφαρμογής είναι να ανιχνεύσει μοτίβα στα δεδομένα αισθητήρων που μπορεί να υποδεικνύουν μια επικείμενη βλάβη του εξοπλισμού, επιτρέποντας τον προγραμματισμό της συντήρησης πριν από την εμφάνιση βλάβης.

Ένας τρόπος για να χρησιμοποιηθεί η βαθιά εκμάθηση για αυτήν την εφαρμογή είναι να εκπαιδευτεί ένα μοντέλο σε ιστορικά δεδομένα αισθητήρων από βιομηχανικά συστήματα. Το μοντέλο μπορεί να μάθει να αναγνωρίζει μοτίβα στα δεδομένα που είναι ενδεικτικά της κανονικής λειτουργίας και, στη συνέχεια, να χρησιμοποιήσει αυτή τη γνώση για να ανιχνεύσει μοτίβα που αποκλίνουν από τον κανόνα, γεγονός που μπορεί να υποδηλώνει μια επικείμενη αποτυχία.

Μια άλλη προσέγγιση είναι η χρήση του αυτόματου κωδικοποιητή για την ανίχνευση ανωμαλιών, όπου ο αυτόματος κωδικοποιητής εκπαιδεύεται να ανακατασκευάζει τα δεδομένα του αισθητήρα. Οποιαδήποτε απόκλιση από τα κανονικά δεδομένα, μπορεί να υποδηλώνει μια επικείμενη αποτυχία.

Ένας άλλος τρόπος χρήσης της βαθιάς εκμάθησης για αυτήν την εφαρμογή είναι η χρήση τεχνικών όπως τα επαναλαμβανόμενα νευρωνικά δίκτυα (RNN) ή τα δίκτυα μακράς βραχυπρόθεσμης μνήμης (LSTM), τα οποία μπορούν να επεξεργάζονται διαδοχικά δεδομένα και να ανιχνεύουν μοτίβα με την πάροδο του χρόνου.

Άρα, η βαθιά εκμάθηση μπορεί να είναι ένα ισχυρό εργαλείο για την παρακολούθηση βιομηχανικών συστημάτων για αστοχίες εξοπλισμού, καθώς επιτρέπει την ανάλυση μεγάλων ποσοτήτων δεδομένων αισθητήρων και τον εντοπισμό λεπτών μοτίβων που μπορεί να υποδηλώνουν μια επικείμενη αστοχία. Με τη χρήση αυτών των τεχνικών, μπορεί να είναι δυνατή η πρόβλεψη της βλάβης του εξοπλισμού προτού συμβεί, γεγονός που μπορεί να εξοικονομήσει σημαντικές ποσότητες χρόνου, χρημάτων και πόρων στις εταιρείες.

- Υγειονομική περίθαλψη

Ο εντοπισμός ασυνήθιστων προτύπων στα ιατρικά δεδομένα για την έγκαιρη διάγνωση ασθενειών είναι μια άλλη εφαρμογή της ανίχνευσης ανωμαλιών χρησιμοποιώντας τη βαθιά μάθηση. Ο στόχος αυτής της εφαρμογής είναι να ανιχνεύσει μοτίβα σε ιατρικά δεδομένα, όπως αποτελέσματα εργαστηρίου, ζωτικές ενδείξεις ασθενών και δεδομένα απεικόνισης, που μπορεί να υποδεικνύουν ένα πρώιμο στάδιο μιας ασθένειας ή μια ασυνήθιστη κατάσταση.

Ένας τρόπος για να χρησιμοποιηθεί η βαθιά εκμάθηση για αυτήν την εφαρμογή είναι να εκπαιδευτεί ένα μοντέλο σε ιστορικά ιατρικά δεδομένα από ασθενείς. Το μοντέλο μπορεί να μάθει να εντοπίζει μοτίβα στα δεδομένα που είναι ενδεικτικά της φυσιολογικής υγείας και στη συνέχεια να χρησιμοποιήσει αυτή τη γνώση για να ανιχνεύσει μοτίβα που αποκλίνουν από τον κανόνα, τα οποία μπορεί να υποδεικνύουν ένα πρώιμο στάδιο μιας ασθένειας.

Μια άλλη προσέγγιση είναι η χρήση του αυτόματου κωδικοποιητή για την ανίχνευση ανωμαλιών. Ο αυτόματος κωδικοποιητής εκπαιδεύεται να αναδομεί τα ιατρικά δεδομένα. Οποιαδήποτε απόκλιση από τα φυσιολογικά δεδομένα, μπορεί να υποδηλώνει πρώιμο στάδιο μιας ασθένειας ή μια ασυνήθιστη κατάσταση.

Ένας άλλος τρόπος χρήσης της βαθιάς εκμάθησης για αυτήν την εφαρμογή είναι η χρήση τεχνικών όπως τα επαναλαμβανόμενα νευρωνικά δίκτυα (RNN) ή τα δίκτυα μακράς βραχυπρόθεσμης μνήμης (LSTM), τα οποία μπορούν να επεξεργάζονται διαδοχικά δεδομένα και να ανιχνεύουν μοτίβα με την πάροδο του χρόνου, όπως ζωτικά σημεία ή αποτελέσματα εργαστηρίου.

Η βαθιά μάθηση μπορεί επίσης να χρησιμοποιηθεί για την ανάλυση δεδομένων απεικόνισης, όπως μια ακτινογραφία, αξονική τομογραφία, μαγνητική τομογραφία και σαρώσεις PET, για τον εντοπισμό μοτίβων που είναι ενδεικτικά συγκεκριμένων ασθενειών ή καταστάσεων. Για παράδειγμα, ένα μοντέλο βαθιάς μάθησης μπορεί να εκπαιδευτεί για να ανιχνεύει μοτίβα στις αξονικές τομογραφίες που είναι ενδεικτικά του καρκίνου του πνεύμονα.

Συνολικά, η βαθιά μάθηση μπορεί να είναι ένα ισχυρό εργαλείο για τον εντοπισμό ασυνήθιστων προτύπων στα ιατρικά δεδομένα για την έγκαιρη διάγνωση ασθενειών, καθώς επιτρέπει την ανάλυση μεγάλου όγκου δεδομένων και τον εντοπισμό προτύπων που μπορεί να υποδηλώνουν πρώιμο στάδιο μιας ασθένειας ή ασυνήθιστη κατάσταση. Με τη χρήση αυτών των τεχνικών, μπορεί να είναι δυνατή η διάγνωση ασθενειών σε πρώιμο στάδιο, γεγονός που μπορεί να βελτιώσει την υγεία των ασθενών και να μειώσει το κόστος υγειονομικής περίθαλψης.

- Κυβερνοασφάλεια

Η ανίχνευση εισβολής σε δίκτυο είναι μια άλλη εφαρμογή ανίχνευσης ανωμαλιών με χρήση βαθιάς μάθησης. Ο στόχος αυτής της εφαρμογής είναι να ανιχνεύσει μοτίβα στα δεδομένα κίνησης δικτύου που μπορεί να υποδηλώνουν απόπειρα εισβολής ή κακόβουλη επίθεση.

Ένας τρόπος για να χρησιμοποιηθεί η βαθιά εκμάθηση για αυτήν την εφαρμογή είναι να εκπαιδευτεί ένα μοντέλο σε ιστορικά δεδομένα κίνησης δικτύου από ένα συγκεκριμένο σύστημα ή δίκτυο. Το μοντέλο μπορεί να μάθει να αναγνωρίζει μοτίβα στα δεδομένα που είναι ενδεικτικά της κανονικής δραστηριότητας δικτύου και στη συνέχεια να χρησιμοποιεί αυτή τη γνώση για να ανιχνεύει μοτίβα που αποκλίνουν από τον κανόνα, τα οποία μπορεί να υποδηλώνουν απόπειρα εισβολής ή κακόβουλη επίθεση.

Μια άλλη προσέγγιση είναι η χρήση του αυτόματου κωδικοποιητή για την ανίχνευση ανωμαλιών, όπου ο αυτόματος κωδικοποιητής εκπαιδεύεται να ανακατασκευάζει τα δεδομένα κίνησης δικτύου. Οποιαδήποτε απόκλιση από τα κανονικά δεδομένα, μπορεί να υποδηλώνει απόπειρα εισβολής ή κακόβουλη επίθεση.

Ένας άλλος τρόπος χρήσης της βαθιάς εκμάθησης για αυτήν την εφαρμογή είναι η χρήση τεχνικών όπως τα επαναλαμβανόμενα νευρωνικά δίκτυα (RNN) ή τα δίκτυα μακράς βραχυπρόθεσμης μνήμης (LSTM), τα οποία μπορούν να επεξεργάζονται διαδοχικά δεδομένα και να ανιχνεύουν μοτίβα με την πάροδο του χρόνου (Yan, et al., 2014).

Η βαθιά εκμάθηση μπορεί επίσης να χρησιμοποιηθεί για την ανάλυση δεδομένων κίνησης δικτύου σε διαφορετικά επίπεδα, όπως επίπεδο πακέτων, επίπεδο ροής και επίπεδο εφαρμογής, για τον εντοπισμό μοτίβων που είναι ενδεικτικά συγκεκριμένων τύπων επιθέσεων. Για παράδειγμα, ένα μοντέλο βαθιάς εκμάθησης μπορεί να εκπαιδευτεί για να ανιχνεύει μοτίβα στην κυκλοφορία δικτύου που είναι ενδεικτικά μιας επίθεσης κατανεμημένης άρνησης υπηρεσίας (DDoS).

Συνοψίζοντας, η βαθιά εκμάθηση μπορεί να είναι ένα ισχυρό εργαλείο για την ανίχνευση εισβολής στο δίκτυο. Με τη χρήση αυτών των τεχνικών, μπορεί να είναι δυνατός ο έγκαιρος εντοπισμός και η απόκριση σε κυβερνοεπιθέσεις, γεγονός που μπορεί να βοηθήσει στην προστασία συστημάτων και δικτύων από ζημιές και διακοπές.

- Κατασκευαστικός Τομέας

Στον κατασκευαστικό τομέα η ανίχνευση ανωμαλιών μέσω της βαθιάς μάθησης βοηθά στον εντοπισμό ελαττωματικών αντικειμένων σε μια διαδικασία κατασκευής. Ένας τρόπος για να γίνει αυτό είναι να εκπαιδευτεί ένα μοντέλο σε ιστορικά δεδομένα αισθητήρων ή εικόνες κατασκευασμένων αντικειμένων. Το μοντέλο μπορεί να μάθει να εντοπίζει μοτίβα στα δεδομένα που είναι ενδεικτικά κανονικών ή καλών στοιχείων και στη συνέχεια να χρησιμοποιεί αυτή τη γνώση για να ανιχνεύει μοτίβα που αποκλίνουν από τον κανόνα, τα οποία μπορεί να υποδηλώνουν ελάττωμα ή μη συμμόρφωση.

Όπως και σε ορισμένα από τα προγούμενα παραδείγματα μια άλλη αποδοτική προσέγγιση είναι η χρήση του αυτόματου κωδικοποιητή για την ανίχνευση ανωμαλιών. Ο αυτόματος κωδικοποιητής εκπαιδεύεται να ανακατασκευάζει τα δεδομένα του αισθητήρα ή τις εικόνες των κατασκευασμένων αντικειμένων. Οποιαδήποτε απόκλιση από τα κανονικά δεδομένα, μπορεί να υποδηλώνει ελάττωμα ή μη συμμόρφωση.

- Επιτήρηση μέσω βίντεο

Πλέον είναι σύνηθες να βλέπουμε κάμερες ασφαλείας και συστήματα παρακολούθησης σε πολλούς δημόσιους και ιδιωτικούς χώρους. Δεδομένων των νέων τεχνολογικών εξελίξεων στις έξυπνες εφαρμογές και τα smartphones, αυτό σίγουρα δεν πρόκειται να αλλάξει σύντομα. Για παράδειγμα στο άμεσο μέλλον, αναμένουμε πολύ περισσότερα έξυπνα αυτοκίνητα και αυτοκινούμενα οχήματα. Αυτά εξαρτώνται από τη συνεχή επεξεργασία βίντεο σε πραγματικό χρόνο για να εντοπίσουν και να αναλύσουν διάφορα αντικείμενα κρίνοντας τα ως φυσιολογικά ή όχι. Όταν βρίσκεται κανείς σε ένα αυτοκινούμενο όχημα σε εθνική οδό, το βίντεο του αυτοκινήτου υποδεικνύει τι είναι φυσιολογικό σύμφωνα με το πώς πρέπει να φαίνεται ο δρόμος, πού πρέπει να βρίσκονται οι πινακίδες, πού πρέπει να βρίσκονται τα δέντρα και πού πρέπει να βρίσκεται το επόμενο αυτοκίνητο. Χρησιμοποιώντας την ανίχνευση ανωμαλιών, τα οχήματα μπορούν να εντοπίσουν προβλήματα στη διαδρομή και στη συνέχεια να αναλάβουν διορθωτική δράση πριν συμβεί οτιδήποτε.

Βέβαια οι κάμερες και τα συστήματα ασφαλείας ανιχνεύουν ανωμαλες εικόνες και σε απλούστερες περιστάσεις. Όταν μια οικειακή κάμερα ανιχνεύει μια ύποπτη κίνηση στο σπίτι (άγριο ζώο, εισβολέας κοκ), το σύστημα ασφαλείας του σπιτιού είναι σε θέση να δει ότι αυτό δεν είναι φυσιολογικό και να ειδοποιήσει τον ιδιοκτήτη. Για να συμβεί αυτό αποτελεσματικά, οι κατασκευαστές εκπαιδεύουν εξελιγμένα μοντέλα μηχανικής μάθησης για να να αξιολογούν τα σήματα βίντεο σε πραγματικό χρόνο.

- Μεταφορές

Στον τομέα των μεταφορών, η ανίχνευση ανωμαλιών μπορεί να χρησιμοποιηθεί για τη διασφάλιση της ορθής λειτουργίας των οδών και των οχημάτων. Εάν μπορούμε να συλλέξουμε διαφορετικούς τύπους συμβάντων από όλους τους αισθητήρες που λειτουργούν στις οδικές αρτηρίες, όπως οι σταθμοί διοδίων, τα φανάρια, οι κάμερες ασφαλείας και τα σήματα GPS, μπορούμε να δημιουργήσουμε μια μηχανή ανίχνευσης ανωμαλιών για να εντοπίσουμε μη φυσιολογικά πρότυπα κυκλοφορίας.

Η ανίχνευση ανωμαλιών μπορεί επίσης να χρησιμοποιηθεί για την εξέταση των χρόνων στα δρομολόγια των δημόσιων συγκοινωνιών και των σχετικών συνθηκών κυκλοφορίας στην περιοχή. Μπορούμε επίσης να αναζητήσουμε ανώμαλη δραστηριότητα όσον αφορά την κατανάλωση καυσίμων, τον αριθμό των επιβατών που υποστηρίζει η δημόσια συγκοινωνία, εποχιακές τάσεις κ.λπ.

- Τραπεζικές υπηρεσίες

Στον τραπεζικό τομέα, μερικές από εφαρμογές ανίχνευσης ανωμαλιών είναι η επισήμανση ασυνήθιστα υψηλών συναλλαγών, ύποπτων δραστηριοτήτων, επιθέσεις phishing κ.λπ. Οι πιστωτικές κάρτες χρησιμοποιούνται από πολλά άτομα παγκοσμίως, και συνήθως κάθε άτομο έχει έναν συγκεκριμένο τρόπο χρήσης της πιστωτικής του κάρτας, ο οποίος διαφέρει από όλους τους άλλους. Έτσι, δημιουργείται ένα προφίλ χρήσης της κάρτας με στοιχεία όπως το πότε τη χρησιμοποιεί, το γιατί τη χρησιμοποιεί και για ποιές αγορές. Αν η τράπεζα έχει τέτοιες πληροφορίες σχετικά με τη χρήση της πιστωτικής κάρτας από πολύ μεγάλο αριθμό καταναλωτών, είναι δυνατόν να χρησιμοποιήσει ανίχνευση ανωμαλιών για να εντοπίσει πότε μια συγκεκριμένη συναλλαγή πιστωτικής κάρτας είναι ύποπτη.

Οι αυτόματοι κωδικοποιητές (autoencoders) είναι πολύ χρήσιμοι σε αυτή την περίπτωση. Μπορούμε να πάρουμε όλες τις συναλλαγές πιστωτικών καρτών από μεμονωμένους καταναλωτές και να μετατρέψουμε τα χαρακτηριστικά τους σε αριθμητικά χαρακτηριστικά, έτσι ώστε να μπορούμε να αποδώσουμε ορισμένες βαθμολογίες σε κάθε πιστωτική κάρτα με βάση διάφορους παράγοντες, μαζί με έναν δείκτη για το αν η συναλλαγή είναι φυσιολογική ή μη φυσιολογική. Στη συνέχεια, χρησιμοποιώντας αυτόματους κωδικοποιητές, μπορούμε να δημιουργήσουμε ένα μοντέλο ανίχνευσης ανωμαλιών που μπορεί να προσδιορίσει γρήγορα αν μια συγκεκριμένη συναλλαγή είναι φυσιολογική ή μη φυσιολογική, λαμβάνοντας υπόψη όλα όσα γνωρίζουμε για όλες τις άλλες συναλλαγές ενός πελάτη. Ο αυτόματος κωδικοποιητής δεν χρειάζεται καν να είναι εξαιρετικά περίπλοκος, μπορεί να κατασκευαστεί με λίγα μόνο κρυφά στρώματα για τον κωδικοποιητή και λίγα κρυφά στρώματα για τον αποκωδικοποιητή και να λειτουργεί αποδοτικά.

- Κοινωνικά δίκτυα

Σε πλατφόρμες κοινωνικής δικτύωσης όπως το Twitter, το Facebook και το Instagram, η ανίχνευση ανωμαλιών μπορεί να χρησιμοποιηθεί για τον εντοπισμό χακαρισμένων λογαριασμών που κάνουν spamming, ψευδών διαφημίσεων, ψεύτικων κριτικών κ.λπ. Οι πλατφόρμες μέσων κοινωνικής δικτύωσης χρησιμοποιούνται εκτενώς από δισεκατομμύρια ανθρώπους, επομένως ο όγκος της δραστηριότητας στις πλατφόρμες είναι εξαιρετικά υψηλός και αυξάνεται συνεχώς. Προκειμένου να διασφαλιστεί η ιδιωτικότητα των ατόμων που τις χρησιμοποιούν, καθώς και να εξασφαλιστεί η σωστή εμπειρία για κάθε άτομο, υπάρχουν πολλές τεχνικές που μπορούν να εφαρμοστούν. Χρησιμοποιώντας την ανίχνευση ανωμαλιών, κάθε ατομική δραστηριότητα μπορεί να εξεταστεί για φυσιολογική και μη φυσιολογική συμπεριφορά.

Ομοίως, κάθε διαφήμιση των πλατφορμών, κάθε εξατομικευμένη σύσταση φίλων, κάθε ειδησεογραφικό άρθρο που μπορεί να ενδιαφέρει το άτομο, όπως οι εκλογές, μπορούν να υποβληθούν σε επεξεργασία για ανώμαλη ή ανώμαλη δραστηριότητα. Θα ήταν πολύ χρήσιμο, αν η ανίχνευση ανωμαλιών μπορούσε να εντοπίσει τα τρολ στα tweets, bots, ψευδείς ειδήσεις και ούτω καθεξής.

Εφαρμογή Ανίχνευσης Ανωμαλιών σε IoT Σύστημα για Ανίχνευση Βλαβών και Σχεδιασμός Αρχιτεκτονικής Επεξεργασίας Δεδομένων Μεγάλης Κλίμακας

Η αποτελεσματική λειτουργία των βιομηχανικών μηχανών είναι ζωτικής σημασίας για τη μεγιστοποίηση της παραγωγικότητας και τη μείωση του λειτουργικού κόστους. Η κατανάλωση ενέργειας διαδραματίζει κρίσιμο ρόλο από αυτή την άποψη, καθώς επηρεάζει άμεσα τη συνολική απόδοση και βιωσιμότητα των βιομηχανικών διεργασιών. Με την έλευση του Industrial Internet of Things (IIoT) και την αυξανόμενη διαθεσιμότητα δεδομένων αισθητήρων, η παρακολούθηση και η ανάλυση των προτύπων κατανάλωσης ενέργειας έχει γίνει πιο εφικτή και πολύτιμη για τη βελτιστοποίηση της απόδοσης των μηχανών.

Οι τεχνικές ανίχνευσης ανωμαλιών έχουν κερδίσει σημαντική προσοχή τα τελευταία χρόνια ως ένα ισχυρό εργαλείο για τον εντοπισμό ανώμαλης συμπεριφοράς ή αποκλίσεων από τα αναμενόμενα μοτίβα στα δεδομένα χρονοσειρών. Η εφαρμογή μεθόδων ανίχνευσης ανωμαλιών σε δεδομένα κατανάλωσης ενέργειας από βιομηχανικά μηχανήματα είναι μια πολλά υποσχόμενη προσέγγιση για τη βελτίωση της λειτουργικής απόδοσης, τη μείωση του χρόνου διακοπής λειτουργίας, την πρόληψη δαπανηρών βλαβών και την προώθηση της βιώσιμης χρήσης ενέργειας. Εντοπίζοντας ανώμαλα πρότυπα κατανάλωσης ενέργειας, οι χειριστές μπορούν να αντιμετωπίσουν προληπτικά τα υποκείμενα ζητήματα, να βελτιστοποιήσουν τα χρονοδιαγράμματα συντήρησης και να βελτιώσουν τη συνολική αξιοπιστία του συστήματος.

Διαθέτουμε δεδομένα χρονοσειρών για την κατανάλωση ενέργειας μιας μηχανής/μηχανών σε εργοστάσιο παρασκευής πλαστικών ειδών συσκευασίας και στόχος μας είναι η διερεύνηση και αξιολόγηση διαφόρων μεθόδων ανίχνευσης ανωμαλιών για την ανάλυση της κατανάλωσης ενέργειας της συγκεκριμένης μηχανής. Συγκεκριμένα, στοχεύουμε:

α) Στη διερεύνηση των χαρακτηριστικών και των προκλήσεων που σχετίζονται με τα δεδομένα κατανάλωσης ενέργειας σε ένα βιομηχανικό πλαίσιο.

β) Στην επανεξέταση σύγχρονων αλγορίθμων και τεχνικών ανίχνευσης ανωμαλιών κατάλληλων για ανάλυση δεδομένων χρονοσειρών.

γ) Στην ανάπτυξη συστηματικού πλαισίου προεπεξεργασίας και εξαγωγής χαρακτηριστικών από χρονοσειρές κατανάλωσης ενέργειας.

δ) Στην εφαρμογή και σύγκριση διαφορετικών μεθόδων ανίχνευσης ανωμαλιών, συμπεριλαμβανομένων στατιστικών προσεγγίσεων, αλγορίθμων μηχανικής μάθησης και μοντέλων βαθιάς μάθησης, για τον εντοπισμό μη φυσιολογικών προτύπων κατανάλωσης ενέργειας.

ε) Στην αξιολόγηση της απόδοσης και της αποτελεσματικότητας των μεθόδων που εφαρμόζονται χρησιμοποιώντας κατάλληλες μετρήσεις αξιολόγησης.

στ) Στο σχεδιασμό κατάλληλου συστήματος διαχείρισης βιομηχανικών δεδομένων χρονοσειρών για ανίχνευση ανωμαλιών σε αυτά.

3.1 Γενικά χαρακτηριστικά & προκλήσεις

1. Υψηλή ένταση και ταχύτητα:

Ο όγκος των δεδομένων κατανάλωσης ενέργειας που παράγονται από βιομηχανικές μηχανές είναι συχνά μεγάλος και ο ρυθμός παραγωγής τους γρήγορος. Οι βιομηχανικές διαδικασίες μπορεί να περιλαμβάνουν πολλές μηχανές που λειτουργούν ταυτόχρονα, με αποτέλεσμα τεράστια ποσότητα δεδομένων να παράγονται σε πραγματικό χρόνο. Η αποτελεσματική διαχείριση και επεξεργασία αυτού του μεγάλου όγκου δεδομένων αποτελεί σημαντική πρόκληση.

2. Εποχικότητα και μοτίβα:

Η κατανάλωση ενέργειας σε βιομηχανικά περιβάλλοντα μπορεί να παρουσιάζει ξεχωριστά μοτίβα και εποχικότητα. Για παράδειγμα, μπορεί να υπάρχουν διακυμάνσεις στη χρήση ενέργειας με βάση διαφορετικές βάρδιες, χρονοδιαγράμματα παραγωγής ή συγκεκριμένες λειτουργικές απαιτήσεις. Ο εντοπισμός και η κατανόηση αυτών των προτύπων είναι ζωτικής σημασίας για την ακριβή ανίχνευση ανωμαλιών, καθώς οι αποκλίσεις από τα αναμενόμενα πρότυπα μπορεί να υποδηλώνουν μη φυσιολογική συμπεριφορά.

3. Ζητήματα θορυβωδών δεδομένων και ποιότητας δεδομένων:

Τα δεδομένα κατανάλωσης ενέργειας που συλλέγονται από βιομηχανικά μηχανήματα ενδέχεται να έχουν θόρυβο, ακραίες τιμές, τιμές που λείπουν και άλλα ζητήματα ποιότητας δεδομένων. Παράγοντες όπως δυσλειτουργίες αισθητήρα, σφάλματα μέτρησης ή διακοπές δικτύου μπορεί να εισάγουν θόρυβο και ανακρίβειες στα δεδομένα. Η αντιμετώπιση τέτοιων θορυβωδών δεδομένων και η διασφάλιση της ποιότητας των δεδομένων είναι απαραίτητη για την αξιόπιστη ανίχνευση ανωμαλιών.

4. Πολυμεταβλητή φύση:

Τα δεδομένα κατανάλωσης ενέργειας σε ένα βιομηχανικό πλαίσιο συχνά περιλαμβάνουν πολλαπλές μεταβλητές ή χαρακτηριστικά, όπως κατανάλωση ενέργειας, επίπεδα τάσης, θερμοκρασία ή λειτουργικές παραμέτρους. Η ανάλυση των αλληλεπιδράσεων και των εξαρτήσεων μεταξύ αυτών των μεταβλητών καθίσταται κρίσιμη για την ακριβή ανίχνευση ανωμαλιών. Η ενσωμάτωση τεχνικών πολυμεταβλητής ανάλυσης και η εξέταση των σχέσεων μεταξύ διαφορετικών χαρακτηριστικών παρουσιάζουν πρόσθετες προκλήσεις.

5. Γνώση συμφραζομένων:

Η ερμηνεία των δεδομένων κατανάλωσης ενέργειας σε ένα βιομηχανικό πλαίσιο απαιτεί γνώση και κατανόηση των υποκείμενων διαδικασιών σε συγκεκριμένο τομέα. Οι ανωμαλίες μπορεί να μην είναι πάντα εμφανείς μόνο από τα δεδομένα. πληροφορίες σχετικά με τα συμφραζόμενα σχετικά με το μηχάνημα, τη λειτουργία του και το περιβάλλον περιβάλλον μπορούν να παρέχουν πολύτιμες πληροφορίες για τη διάκριση των κανονικών παραλλαγών από τα μη φυσιολογικά συμβάντα. Η ενσωμάτωση της τεχνογνωσίας του τομέα και της γνώσης των συμφραζομένων στη διαδικασία ανίχνευσης ανωμαλιών είναι απαραίτητη για ακριβή και ουσιαστικά αποτελέσματα.

6. Μη ισορροπημένα δεδομένα:

Στα δεδομένα κατανάλωσης ενέργειας βιομηχανικών μηχανών, οι ανωμαλίες είναι συνήθως σπάνια συμβάντα σε σύγκριση με τις κανονικές συνθήκες λειτουργίας. Αυτή η ανισορροπία κλάσης αποτελεί πρόκληση για τους παραδοσιακούς αλγόριθμους ανίχνευσης ανωμαλιών που έχουν σχεδιαστεί για ισορροπημένα σύνολα δεδομένων. Μέθοδοι για την αντιμετώπιση της ανισορροπίας τάξης, όπως η υπερδειγματοληψία σπάνιων γεγονότων ή η χρήση τεχνικών βαθμολόγησης ανωμαλιών, πρέπει να ληφθούν υπόψη για να διασφαλιστεί η αποτελεσματική ανίχνευση ανωμαλιών.

7. Προσαρμοστικότητα και επεκτασιμότητα:

Τα βιομηχανικά περιβάλλοντα είναι δυναμικά, με εξελισσόμενες λειτουργίες, αλλαγές στις διαμορφώσεις των μηχανών και ποικίλα πρότυπα κατανάλωσης ενέργειας. Οι μέθοδοι ανίχνευσης ανωμαλιών που εφαρμόζονται στα δεδομένα κατανάλωσης ενέργειας βιομηχανικών μηχανών θα πρέπει να είναι προσαρμόσιμες και επεκτάσιμες για να προσαρμόζονται στις μεταβαλλόμενες συνθήκες. Η ικανότητα ενημέρωσης μοντέλων και ανίχνευσης ανωμαλιών σε πραγματικό χρόνο είναι ζωτικής σημασίας για την προληπτική συντήρηση και την έγκαιρη λήψη αποφάσεων.

Η κατανόηση αυτών των χαρακτηριστικών και προκλήσεων που σχετίζονται με τα δεδομένα κατανάλωσης ενέργειας σε ένα βιομηχανικό πλαίσιο είναι θεμελιώδης για την επιλογή κατάλληλων τεχνικών ανίχνευσης ανωμαλιών και την ανάπτυξη ισχυρών λύσεων που μπορούν να παρακολουθούν και να βελτιστοποιούν αποτελεσματικά την απόδοση του μηχανήματος, την ενεργειακή απόδοση και τη λειτουργική αξιοπιστία.

3.2 Οπτικοποίηση και προεπεξεργασία δεδομένων

Για να απλοποιήσουμε τη διαδικασία της ανίχνευσης ανωμαλιών καθώς οι διαθέσιμοι πόροι για την εργασία δεν είναι αρκετοί ώστε να είναι εφικτή η διαχείριση τεράστιου όγκου δεδομένων, έχουμε επιλέξει να απομονώσουμε τα δεδομένα χρονοσειράς της μηχανής 19 (M19) και να εργαστούμε πάνω σε αυτά.

Σε μορφή dataframe τα δεδομένα μας έχουν τη συγκεκριμένη μορφή:

	cnodeuid	timestamp	value	type	cnt
0	76470	2018-02-09 10:30:00	27.000	\N	1
1	76470	2018-02-09 10:45:00	26.250	\N	1
2	76470	2018-02-09 11:00:00	27.125	\N	1
3	76470	2018-02-09 11:15:00	26.625	\N	1
4	76470	2018-02-09 11:30:00	26.750	\N	1

Το μέγεθος του dataframe είναι: `(171839, 5)`

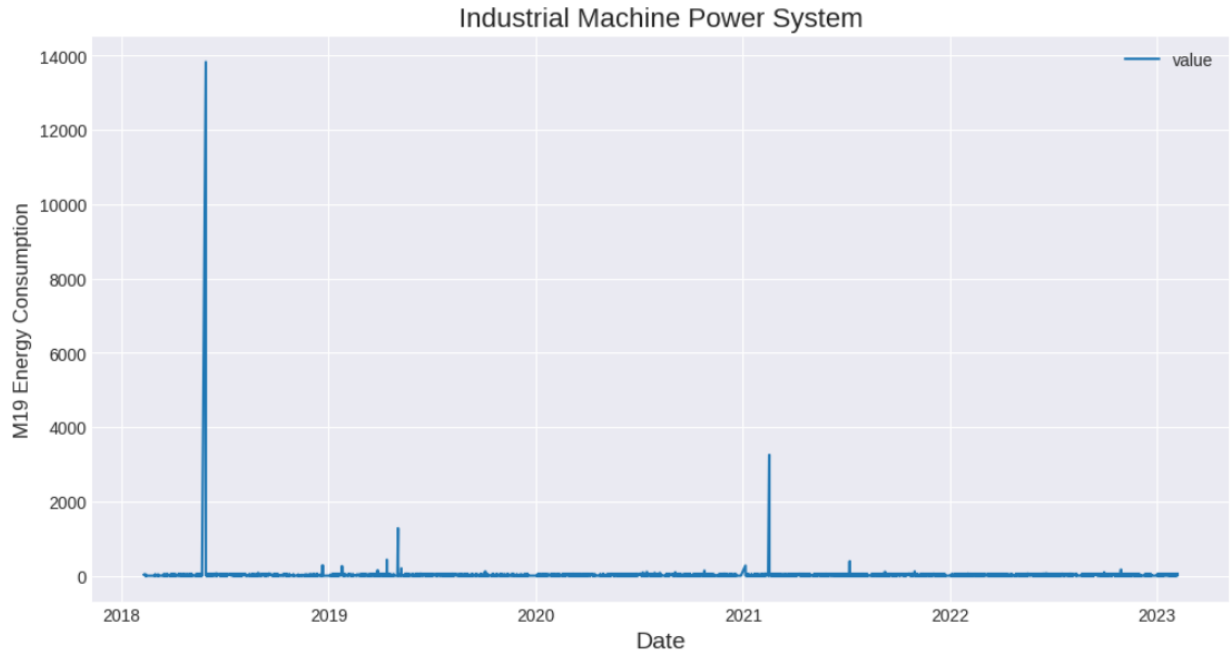
Εμάς μας απασχολούν οι στήλες timestamp και value. Η στήλη value είναι ουσιαστικά η τιμή της ενέργειας σε kWh που καταναλώνει η M19 τη χρονική στιγμή που μας δείχνει η τιμή της στήλης timestamp.

```
[ ]  df_1.describe()
```

	cnodeuid	value	cnt
count	171839.0	171839.000000	171839.0
mean	76470.0	18.699019	1.0
std	0.0	36.576330	0.0
min	76470.0	0.000000	1.0
25%	76470.0	2.000000	1.0
50%	76470.0	25.250000	1.0
75%	76470.0	28.375000	1.0
max	76470.0	13828.375000	1.0

Παρατηρούμε ότι η μέγιστη τιμή της στήλης value που εμφανίζεται στον πίνακα απέχει πολύ από το μέσο όρο των τιμών. Ας δούμε και την οπτικοποίηση των δεδομένων μας χωρίς κάποια αρχική προεργασία.

```
          count       mean       std  min  25%    50%     75%       max
value  171839.0  18.699019  36.57633  0.0  2.0  25.25  28.375  13828.375
```



Εικόνα 3-1 Χρονοσειρά M19 χωρίς προεπεξεργασία

Κάποιες τιμές παρουσιάζουν μεγάλη απόκλιση από τα υπόλοιπα δεδομένα και εμποδίζουν τη σωστή ανίχνευση ανωμαλιών. Οι ακραίες τιμές μπορούν να αλλάξουν δραστικά τα αποτελέσματα της ανάλυσης δεδομένων και της στατιστικής μοντελοποίησης.

Για να μπορούμε να μελετήσουμε πιο αποτελεσματικά τη χρονοσειρά μας και να εφαρμόσουμε στη συνέχεια τις μεθόδους που θέλουμε, επιλέγουμε να αφαιρέσουμε τα οφθαλμοφανή outliers.

```
maxx = df_1['value'].mean()+2*df_1['value'].std()
print('maxx df_1 = ', maxx)
minn = df_1['value'].mean()-2*df_1['value'].std()
print('minn df_1 = ', minn)
```

```
maxx df_1 =  91.8516783196474
minn df_1 =  -54.45364003962949
```

```
#Data Frame after some preprocessing
df_1 = df_1[(df_1['value'] < maxx)]
```

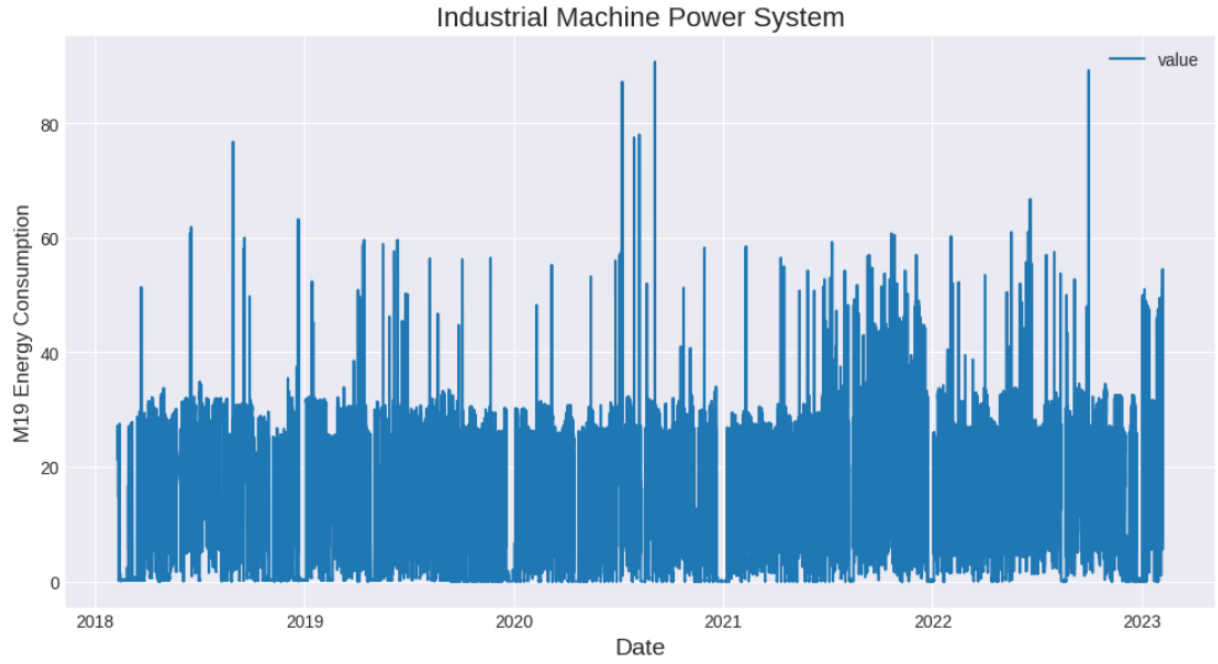
```
df_1.shape
```

```
(171821, 5)
```

Βλέπουμε ότι από `171839` οι γραμμές στο df μας έγιναν `171821`.

Η οπτικοποίηση των δεδομένων μας μετά την προεργασία είναι:

	count	mean	std	min	25%	50%	75%	max
value	171821.0	18.575618	12.405307	0.0	2.0	25.25	28.375	90.75



Εικόνα 3-2 Χρονοσειρά M19 μετά την προεπεξεργασία

Τώρα η μέγιστη τιμή της χρονοσειράς παρουσιάζει αισθητά μικρότερη απόκλιση από τον μέσο όρο. Αν και συνεχίζουν να υπαρχουν outliers ορατά με το μάτι, φαίνεται ότι η χρονοσειρά τώρα μπορεί να αναλυθεί πιο σωστά.

Για την παρούσα διπλωματική έχουμε γράψει ορισμένα google colaboratory notebooks στα οποία ύστερα από το κατάλληλο preprocessing εφαρμόσαμε διάφορες τεχνικές για ανίχνευση των ανώμαλων τιμών στη χρονοσειρά της μηχανής M19. Αρχικά, χρησιμοποιήσαμε μεθόδους από τη στατιστική ανάλυση, όπως Interquartile Range Method (IQR), Standard Deviation, Z-score, modified Z-score και Grubb's test. Ύστερα προχωρήσαμε εφαρμόζοντας αναλυτικά τη μέθοδο Isolation forest κι ένα απλό LSTM. Στο τέλος χρησιμοποιήσαμε αυτόματο κωδικοποιητή RNN.

Στη συνέχεια θα παρουσιάσουμε αναλυτικά τη δουλειά μας.

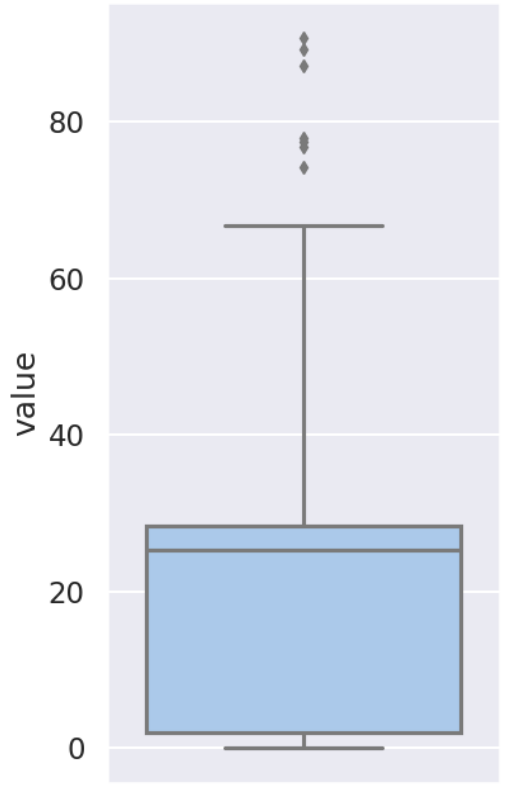
3.3 Ανίχνευση ανωμαλιών και ακραίων τιμών

3.3.1 Interquartile Range Method (IQR)

Μετά την αρχική προεπεξεργασία δεδομένων, για να έχουμε καλύτερη εποπτεία των δεδομένων μας τα απεικονίσαμε γραφικά.

```
plt.figure(figsize = (4,8))
sns.boxplot(y = df_1.value)
```

```
<Axes: ylabel='value'>
```

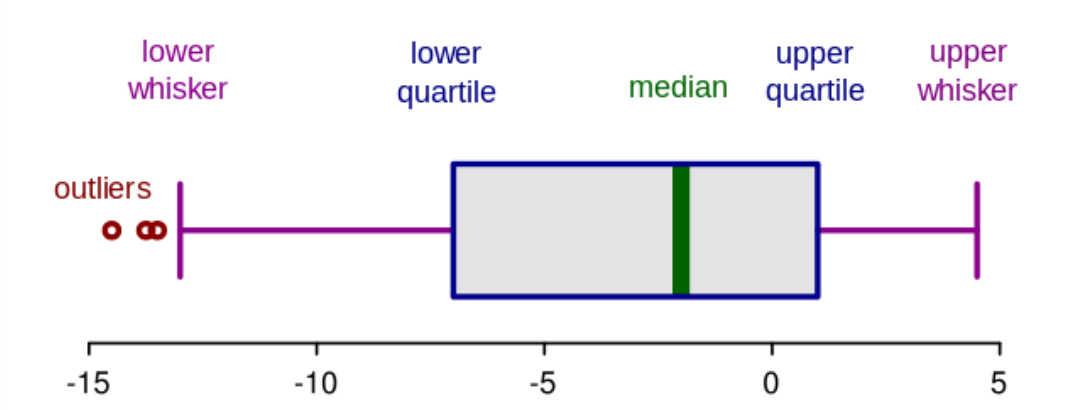


Εικόνα 3-3 Boxplot δεδομένων της χρονοσειράς M19

Από το παραπάνω boxplot φαίνεται ξεκάθαρα ότι υπάρχουν ακραίες τιμές στα δεδομένα μας.

Το IQR σημαίνει Interquartile Range. Είναι ένα στατιστικό μέτρο που χρησιμοποιείται για την αξιολόγηση της εξάπλωσης ή της μεταβλητότητας ενός συνόλου δεδομένων. Το IQR υπολογίζεται από τη διαφορά μεταξύ του τρίτου τεταρτημορίου (Q3) και του πρώτου τεταρτημορίου (Q1) ενός συνόλου δεδομένων.

Στην ανίχνευση ανωμαλιών, το IQR χρησιμοποιείται συχνά ως όριο για τον εντοπισμό ακραίων ή ανωμαλιών σε ένα σύνολο δεδομένων. Το IQR χρησιμοποιείται στη μέθοδο Tukey's fences, όπου τα ακραία σημεία ορίζονται ως σημεία δεδομένων που πέφτουν κάτω από το Q1 - k * IQR ή πάνω από το Q3 + k * IQR, όπου το k είναι μια σταθερά που ορίζεται από τον χρήστη που συνήθως ορίζεται σε 1,5 ή 3. Εμείς την έχουμε ορίσει ως 1,5.



Εικόνα 3-4 Οπτικοποίηση IQR

Ορίσαμε την παρακάτω συνάρτηση για τον υπολογισμό του IQR.

```
def out_iqr(df , column):
    global lower,upper
    q25, q75 = np.quantile(df[column], 0.25), np.quantile(df[column], 0.75)
    # calculate the IQR
    iqr = q75 - q25
    # calculate the outlier cutoff
    cut_off = iqr * 1.5
    # calculate the lower and upper bound value
    lower, upper = q25 - cut_off, q75 + cut_off
    print('The IQR is',iqr)
    print('The lower bound value is', lower)
    print('The upper bound value is', upper)
    # Calculate the number of records below and above lower and above bound value respectively
    df1 = df[df[column] > upper]
    df2 = df[df[column] < lower]
    return print('Total number of outliers are', df1.shape[0]+ df2.shape[0])
```

Η λογική πίσω από τη χρήση του IQR για την ανίχνευση ανωμαλιών είναι ότι παρέχει ένα ισχυρό μέτρο της εξάπλωσης του συνόλου δεδομένων, λιγότερο επηρεασμένο από ακραίες τιμές ή ακραίες τιμές. Ορίζοντας ένα όριο με βάση το IQR, τα σημεία δεδομένων που αποκλίνουν σημαντικά από το τυπικό εύρος τιμών επισημαίνονται ως ανωμαλίες.

Τρέχοντας τη συνάρτηση με τα δεδομένα μας παίρνουμε:

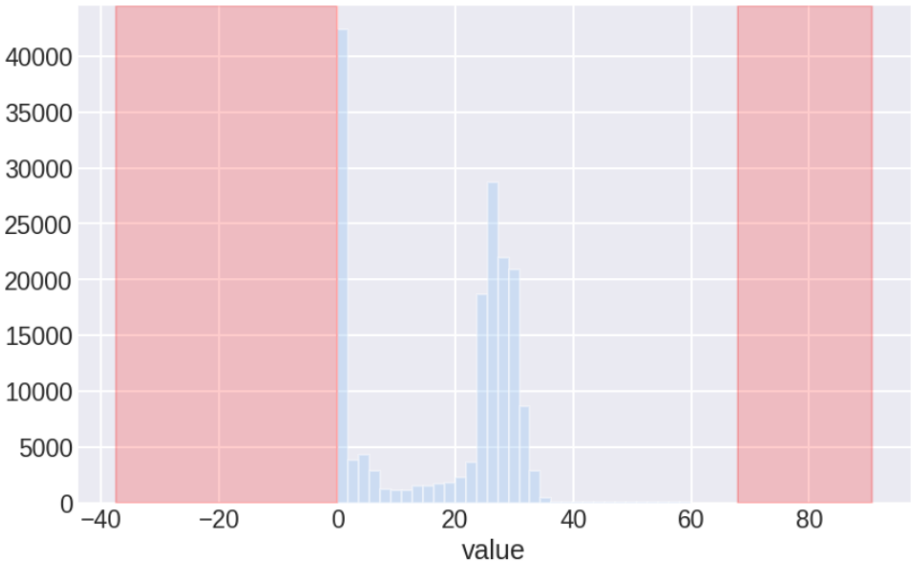
```
out_iqr(df_1,'value')
```

```
The IQR is 26.375
The lower bound value is -37.5625
The upper bound value is 67.9375
Total number of outliers are 7
```

Τα αποτελέσματα αυτά οπτικοποιημένα είναι:

```
plt.figure(figsize = (10,6))
sns.distplot(df_1.value, kde=False)
plt.axvspan(xmin = lower,xmax= df_1.value.min(),alpha=0.2, color='red')
plt.axvspan(xmin = upper,xmax= df_1.value.max(),alpha=0.2, color='red')
```

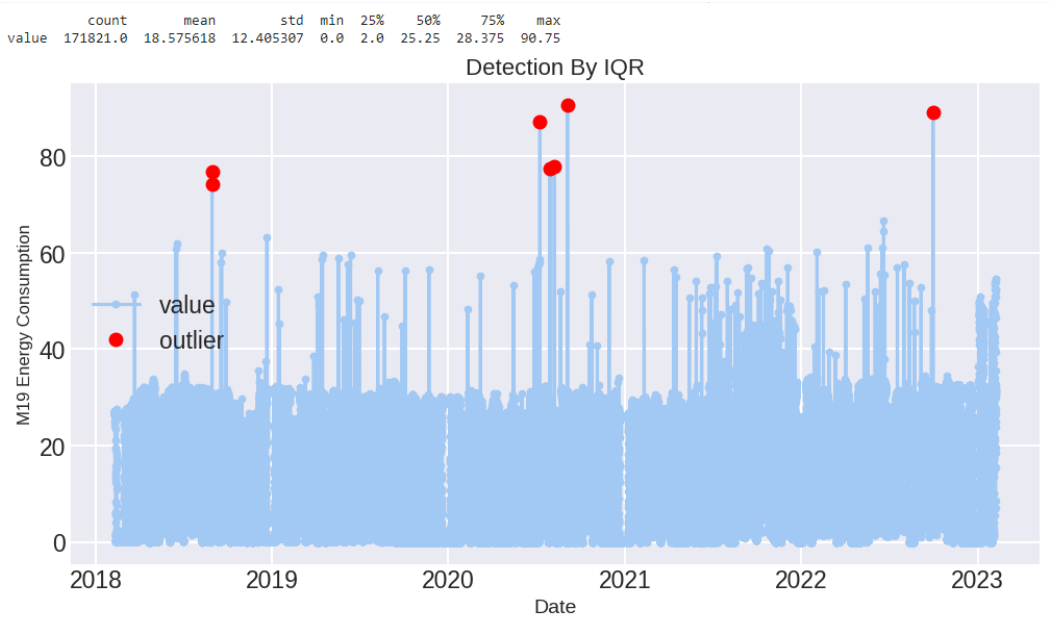
```
<matplotlib.patches.Polygon at 0x7f071e43ed40>
```



Εικόνα 3-5 Ραβδόγραμμα αποτελεσμάτων με IQR

Εδώ η κόκκινη ζώνη αντιπροσωπεύει την ζώνη ακραίων τιμών. Τα σημεία που υπάρχουν σε αυτή τη ζώνη θεωρούνται ακραίες τιμές.

```
plt.style.use('seaborn-darkgrid')
plt.figure(figsize = (12, 6))
plt.xlabel('Date', fontsize = 14)
plt.ylabel('M19 Energy Consumption', fontsize = 12)

print(df_1[['value']].describe().T)
plt.plot(df_1['timestamp'], df_1['value'], marker = '.', label = 'value')
plt.plot(outliers['timestamp'], outliers['value'], 'o', color = 'red', label = 'outlier')

plt.title('Detection By IQR', fontsize = 16)
plt.legend()
plt.show()
```



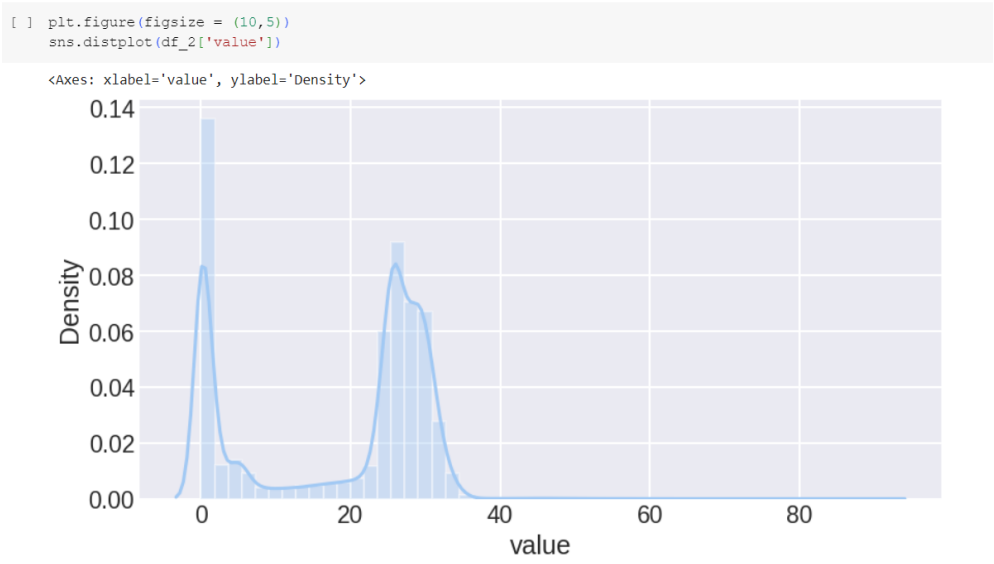
Εικόνα 3-6 Χρονοσειρά M19 με επισημασμένες τις ανωμαλίες με χρήση IQR

Η μέθοδος IQR είναι απλή και υπολογιστικά αποδοτική, άρα κατάλληλη για τον εντοπισμό ακραίων τιμών σε διάφορες εφαρμογές, συμπεριλαμβανομένης της ανάλυσης δεδομένων χρονοσειρών.

3.3.2 Standard Deviation Method

Η μέθοδος τυπικής απόκλισης είναι μια κοινή στατιστική τεχνική που χρησιμοποιείται στην ανίχνευση ανωμαλιών για τον εντοπισμό ακραίων τιμών σε ένα σύνολο δεδομένων. Βασίζεται στην υπόθεση ότι τα σημεία δεδομένων σε ένα σύμπλεγμα κανονικής κατανομής συγκεντρώνονται στενά μεταξύ τους, ενώ οι ανωμαλίες αποκλίνουν σημαντικά από τον μέσο όρο.

Χρησιμοποιώντας ένα απλό διάγραμμα πυκνότητας για τα δεδομένα μας, βλέπουμε:



Εικόνα 3-7 Διάγραμμα τυπικής απόκλισης M19

Στο notebook ορίσαμε την παρακάτω συνάρτηση για τον υπολογισμό του Standard deviation.

```
def out_std(df, column):
    global lower,upper
    # calculate the mean and standard deviation of the data frame
    data_mean, data_std = df[column].mean(), df[column].std()
    # calculate the cutoff value
    cut_off = data_std * 3
    # calculate the lower and upper bound value
    lower, upper = data_mean - cut_off, data_mean + cut_off
    print('The lower bound value is', lower)
    print('The upper bound value is', upper)
    # Calculate the number of records below and above lower and above bound value respectively
    df1 = df[df[column] > upper]
    df2 = df[df[column] < lower]
    return print('Total number of outliers are', df1.shape[0]+ df2.shape[0])
```

Τρέχοντας τη συνάρτηση με τα δεδομένα μας παίρνουμε:

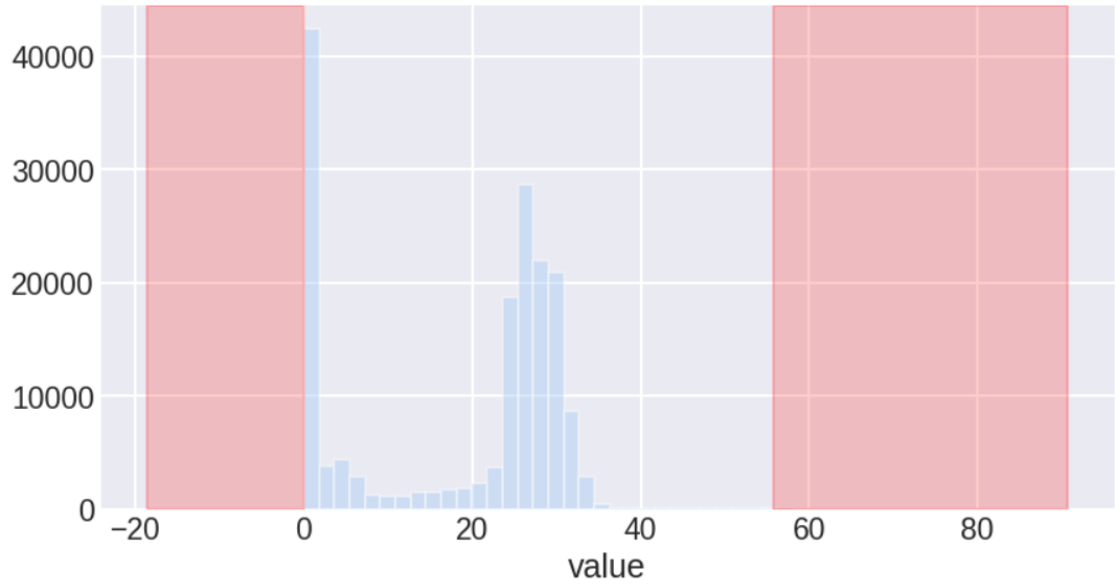
```
out_std(df_2,'value')
```

```
The lower bound value is -18.64030400751105
The upper bound value is 55.79153988671092
Total number of outliers are 43
```

Τα αποτελέσματα αυτά οπτικοποιημένα είναι:

```
plt.figure(figsize = (10,5))
sns.distplot(df_2['value'], kde=False)
plt.axvspan(xmin = lower,xmax= df_2['value'].min(),alpha=0.2, color='red')
plt.axvspan(xmin = upper,xmax= df_2['value'].max(),alpha=0.2, color='red')
```

```
<matplotlib.patches.Polygon at 0x7f0752abe800>
```

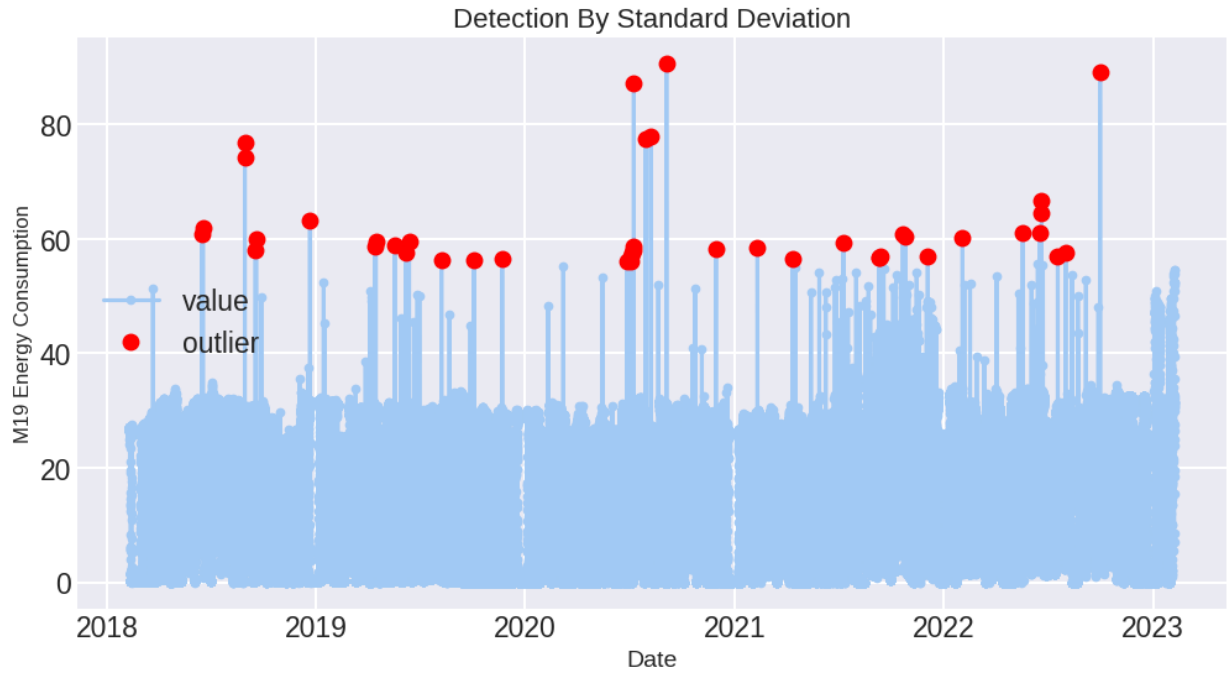


Εικόνα 3-8 Ραβδόγραμμα Αποτελεσμάτων με Standard Deviation

Για να καταλάβουμε ακριβώς με τη συγκεκριμένη μέθοδο που βρίσκονται τα ανώμαλα σημεία μας:

```
plt.style.use('seaborn-darkgrid')
plt.figure(figsize = (12, 6))
plt.xlabel('Date', fontsize = 14)
plt.ylabel('M19 Energy Consumption', fontsize = 12)

print(df_1[['value']].describe().T)
plt.plot(df_1['timestamp'], df_1['value'], marker = '.', label = 'value')
plt.plot(outliers_std['timestamp'], outliers_std['value'], 'o', color = 'red', label = 'outlier')

plt.title('Detection By Standard Deviation', fontsize = 16)
plt.legend()
plt.show()
```

```
         count       mean        std  min  25%    50%     75%    max
value  171821.0  18.575618  12.405307  0.0  2.0  25.25  28.375  90.75
```



Εικόνα 3-9 Χρονοσειρά M19 με επισημασμένες τις ανωμαλίες με Standard Deviation

Η Μέθοδος Τυπικής Απόκλισης είναι σχετικά απλή στην εφαρμογή και παρέχει έναν γρήγορο τρόπο εντοπισμού ανωμαλιών με βάση την υπόθεση των κανονικά κατανεμημένων δεδομένων. Ωστόσο, είναι σημαντικό να σημειωθεί ότι αυτή η μέθοδος προϋποθέτει ότι τα δεδομένα ακολουθούν μια κανονική κατανομή. Σε περιπτώσεις όπου τα δεδομένα αποκλίνουν σημαντικά από την κανονικότητα, θα πρέπει να ληφθούν υπόψη εναλλακτικές μέθοδοι ή τεχνικές ανίχνευσης αγνωστικών ανωμαλιών κατανομής για πιο ακριβή αποτελέσματα.

Στη δική μας περίπτωση όπως φανερώνει και το αρχικό διάγραμμα πυκνότητας, τα δεδομένα δεν ακολουθούν κανονική κατανομή και δεν μας αρκεί η μέθοδος τυπικής απόκλισης, η οποία μάλιστα σε σχέση με τη μέθοδο IQR μας επιστρέφει πολλές ακραίες τιμές.

3.3.3 Z-score Method

Η μέθοδος Z-score είναι μια στατιστική τεχνική που χρησιμοποιείται στην ανίχνευση ανωμαλιών για τον εντοπισμό ακραίων τιμών σε ένα σύνολο δεδομένων. Μετρά πόσες τυπικές αποκλίσεις απέχει ένα σημείο δεδομένων από τον μέσο όρο. Το Z-score παρέχει μια κανονικοποιημένη τιμή που επιτρέπει τη σύγκριση μεταξύ διαφορετικών συνόλων δεδομένων και κατανομών.

Στο notebook ορίσαμε την παρακάτω συνάρτηση για τον υπολογισμό του Z-score.

```
def out_zscore(data):
    global outliers,zscore
    outliers = []
    zscore = []
    threshold = 3
    mean = np.mean(data)
    std = np.std(data)
    for i in data:
        z_score= (i - mean)/std
        zscore.append(z_score)
        if np.abs(z_score) > threshold:
            outliers.append(i)
    return print("Total number of outliers are",len(outliers))
```

Τρέχοντας τη συνάρτηση με τα δεδομένα μας παίρνουμε:

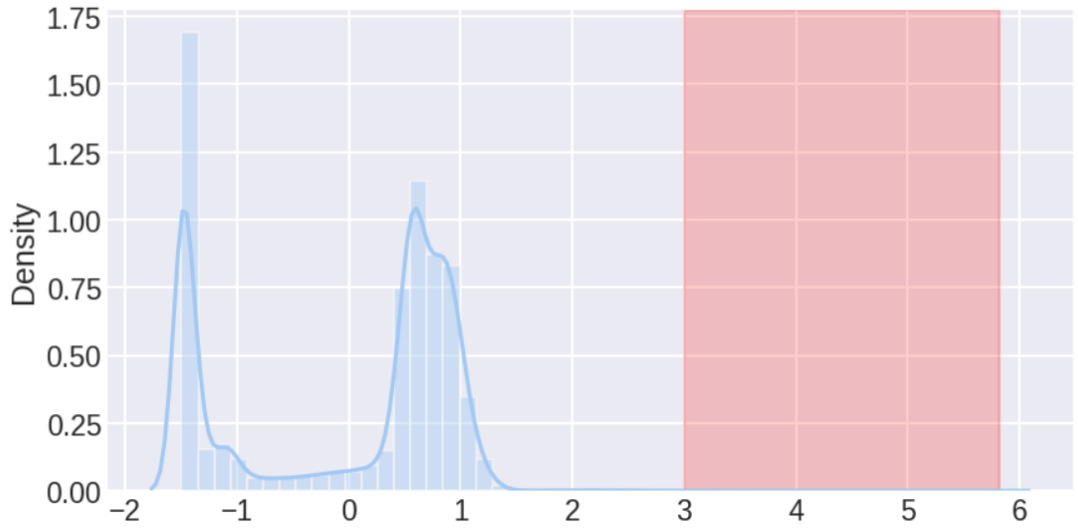
```
out_zscore(df_3.value)
```

```
Total number of outliers are 43
```

Τα αποτελέσματα αυτά οπτικοποιημένα είναι:

```
plt.figure(figsize = (10,5))
sns.distplot(zscore)
plt.axvspan(xmin = 3 ,xmax= max(zscore),alpha=0.2, color='red')
```

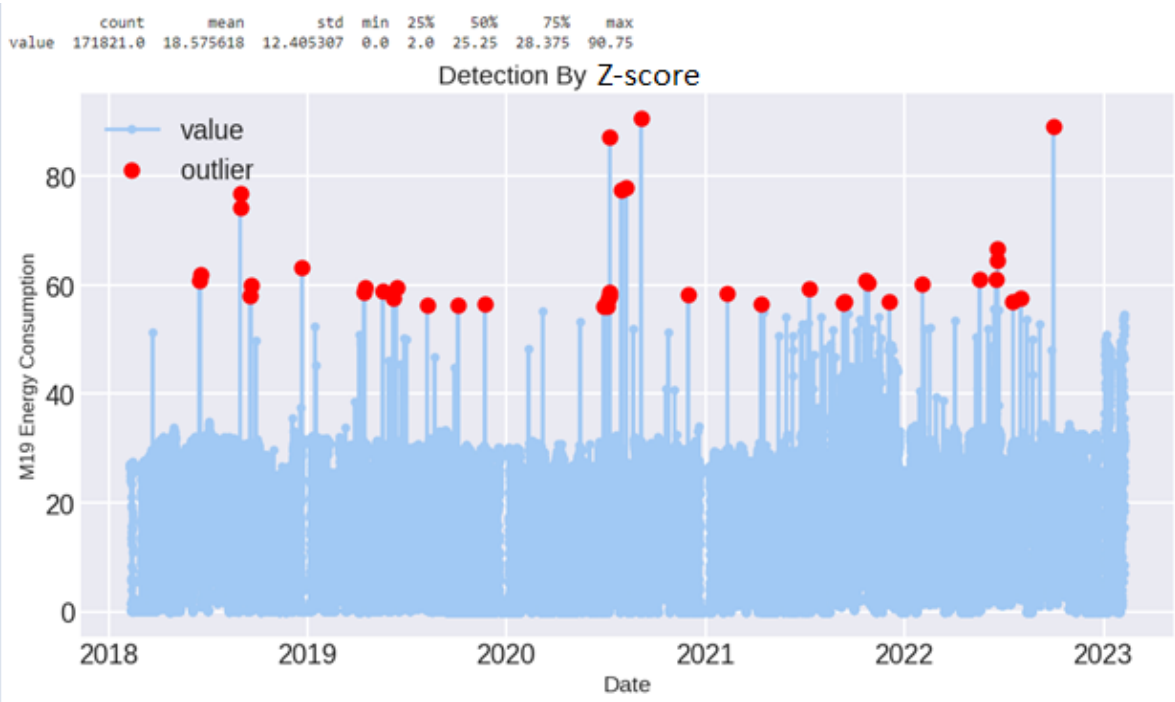
```
<matplotlib.patches.Polygon at 0x7f071bde1990>
```



Εικόνα 3-10 Ραβδόγραμμα Αποτελεσμάτων με Z-score

Και ακόμη πιο ευδιάκριτη οπτικοποίηση:

	count	mean	std	min	25%	50%	75%	max
value	171821.0	18.575618	12.405307	0.0	2.0	25.25	28.375	90.75



Εικόνα 3-11 Χρονοσειρά M19 με επισημασμένες τις ανωμαλίες με Z-score

Η μέθοδος Z-score χρησιμοποιείται ευρέως στην ανίχνευση ανωμαλιών επειδή παρέχει ένα τυποποιημένο μέτρο που μπορεί να εφαρμοστεί σε σύνολα δεδομένων με διαφορετικές κλίμακες και

κατανομές. Ωστόσο, είναι σημαντικό να σημειωθεί ότι η μέθοδος Z-score υποθέτει ότι τα δεδομένα ακολουθούν μια κανονική κατανομή. Εάν τα δεδομένα δεν συμμορφώνονται με μια κανονική κατανομή, θα πρέπει να ληφθούν υπόψη εναλλακτικές μέθοδοι ή τεχνικές ανίχνευσης ανωμαλιών για ακριβή αποτελέσματα.

Όπως επισημάναμε πριν, τα δικά μας δεδομένα δεν ακολουθούν κανονική κατανομή. Εν συνεχεία, λοιπόν, συνεχίζοντας να δουλεύουμε με στατιστικές μεθόδους για ανίχνευση ανωμαλιών εξετάσαμε πώς λειτουργεί το modified Z-score σε σχέση με το απλό καθώς και το Grubb's test.

3.3.4 Modified Z-Score Method

Η μέθοδος modified Z-score όπως έχουμε αναφερει σε προηγούμενο κεφάλαιο είναι μια βελτιωμένη έκδοση της μεθόδου Z-score που αντιμετωπίζει το ζήτημα των ακραίων τιμών σε σύνολα δεδομένων με λοξές ή μη κανονικές κατανομές.

Στο notebook ορίσαμε την παρακάτω συνάρτηση για τον υπολογισμό του Z-score.

```
def modified_zscore(data, consistency_correction=1.4826):

    median = np.median(data)

    deviation_from_med = np.array(data) - median

    mad = np.median(np.abs(deviation_from_med))
    mod_zscore = deviation_from_med/(consistency_correction*mad)
    return mod_zscore, mad
```

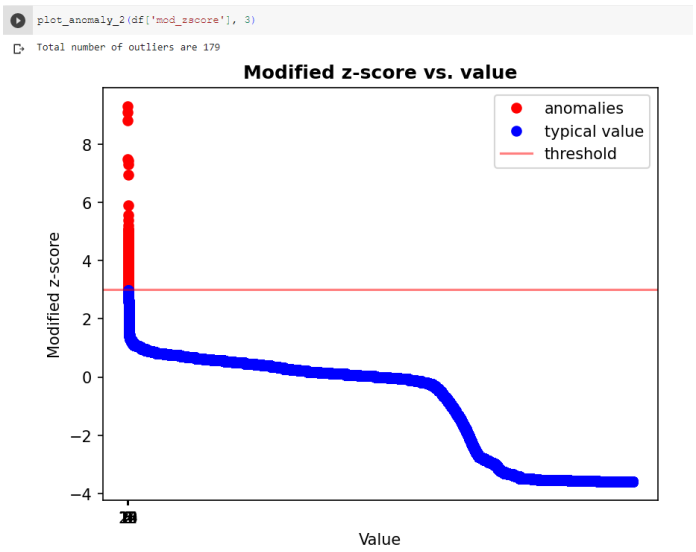
Τρέχοντας τη συνάρτηση με τα δεδομένα μας παίρνουμε:

```
mod_zscore_value, mad_value = modified_zscore(df['value'])
df = df.assign(mod_zscore=mod_zscore_value)
```

```
df.head(5)
```

timestamp	cnodeuid	value	type	cnt	zscore	mod_zscore
2018-02-09 10:30:00	76470	27.000	\N	1	0.679097	0.248497
2018-02-09 10:45:00	76470	26.250	\N	1	0.618639	0.141998
2018-02-09 11:00:00	76470	27.125	\N	1	0.689173	0.266246
2018-02-09 11:15:00	76470	26.625	\N	1	0.648868	0.195247
2018-02-09 11:30:00	76470	26.750	\N	1	0.658944	0.212997

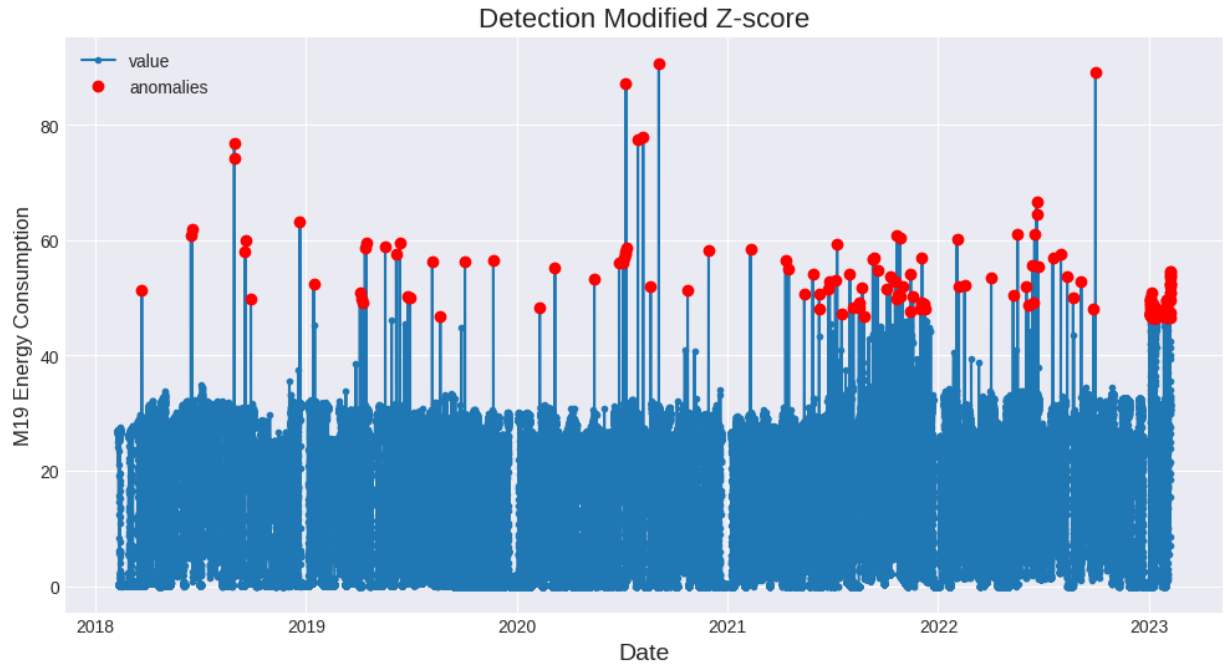
Τα αποτελέσματα αυτά οπτικοποιημένα είναι:



Εικόνα 3-12 Αποτέλεσμα modified Z-score

Και με διαφορετικό διάγραμμα:

	count	mean	std	min	25%	50%	75%	max
value	171821.0	18.575618	12.405307	0.0	2.0	25.25	28.375	90.75



Εικόνα 3-13 Χρονοσειρά M19 με επισημασμένες τις ανωμαλίες με Modified Z-score

Η μέθοδος τροποποιημένης βαθμολογίας Z είναι ιδιαίτερα χρήσιμη για σύνολα δεδομένων που παρουσιάζουν λοξότητα ή έχουν μη κανονικές κατανομές, καθώς βασίζεται στη διάμεσο και στο MAD αντί για τη μέση και τυπική απόκλιση. Αυτό το καθιστά πιο ανθεκτικό έναντι των ακραίων τιμών και παρέχει ένα αξιόπιστο μέτρο απόκλισης από την κεντρική τάση των δεδομένων. Ωστόσο, είναι σημαντικό να σημειωθεί ότι η μέθοδος modified z-score μπορεί να εξακολουθεί να έχει περιορισμούς σε κάποιες

κατανομές ή σύνολα δεδομένων με πολυτροπικά μοτίβα, οπότε θα πρέπει να ληφθούν υπόψη εναλλακτικές μέθοδοι ή τεχνικές ανίχνευσης αγνωστικών ανωμαλιών κατανομής.

3.3.5 Grubb's Test

Η δοκιμή Grubbs, γνωστή και ως δοκιμή ακραίων τιμών Grubbs, είναι μια στατιστική δοκιμή που χρησιμοποιείται για την ανίχνευση ακραίων τιμών σε ένα μονομεταβλητό σύνολο δεδομένων που ακολουθεί μια κανονική κατανομή, όπως έχει αναφερθεί εκτενέστερα σε προηγούμενο κεφάλαιο. Παρά το γεγονός ότι τα δεδομένα μας δεν ακολουθούν κανονική κατανομή, όπως έχουμε διαπιστώσει ως τώρα, αξίζει να δούμε τα αποτελέσματα της δοκιμής Grubb's σε ένα σύνολο δεδομένων όπως το δικό μας.

Ακολουθεί ο κώδικας μας για τη δοκιμή Grubb's:

```
from outliers import smirnov_grubbs as grubbs
```

```
# Detect anomalies using the Grubbs' test method
data = df['value'].to_numpy()
print(data.size)

grubbs_test = grubbs.test(data, alpha=0.05)
print(grubbs_test)
print(grubbs_test.size)

grubIndexes = grubbs.two_sided_test_indices(data, alpha=.05)
print(grubIndexes)
#print(grubbs.two_sided_test_outliers(data, alpha=.05))

anomalies_grubbstest_two_sided = df.iloc[grubIndexes]
anomalies_grubbstest_two_sided
```

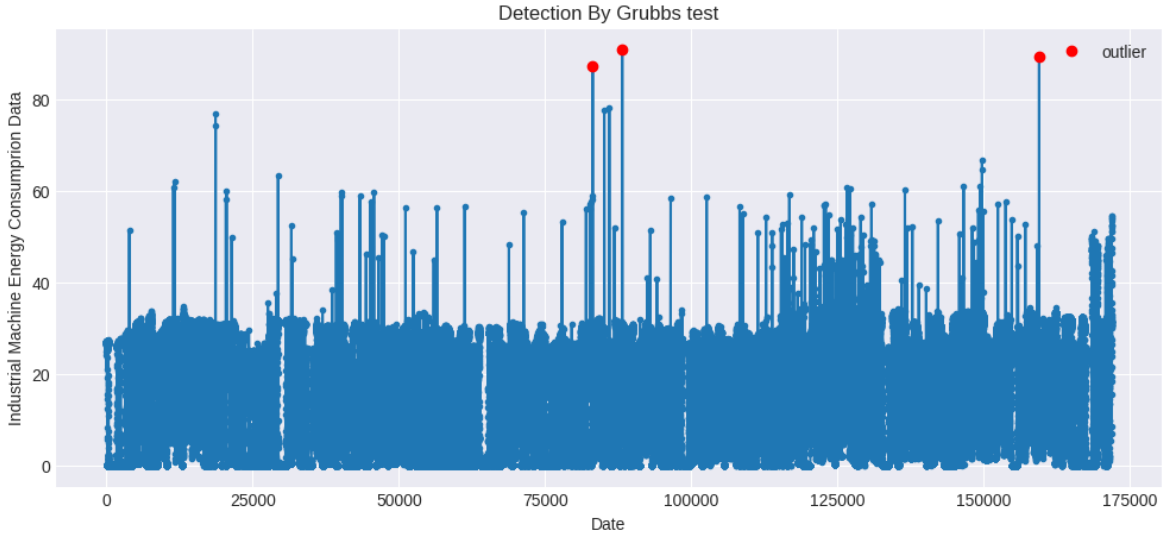
```
171821
[27.    26.25  27.125 ... 31.5   30.5   31.   ]
171818
[88178, 159449, 83127]
```

	cnodeuid	timestamp	value	type	cnt	zscore	mod_zscore
88189	76470	2020-09-03 16:15:00	90.75	\N	1	5.818041	9.300873
159466	76470	2022-09-30 12:15:00	89.25	\N	1	5.697125	9.087875
83137	76470	2020-07-08 16:15:00	87.25	\N	1	5.535903	8.803879

Ενδιαφέρον έχει να δούμε και τα αποτελέσματα της δοκιμής οπτικοποιημένα.

```
plt.figure(figsize = (12, 5))

plt.plot(df['value'], marker = '.')
plt.plot(anomalies_grubbstest_two_sided['value'], 'o', color = 'red', label = 'outlier')
plt.title('Detection By Grubbs test')

#plt.grid()
plt.xlabel('Date')
plt.ylabel('Industrial Machine Energy Consumprion Data')
plt.legend()
```

```
<matplotlib.legend.Legend at 0x7f4a53a1d1e0>
```



Εικόνα 3-14 Χρονοσειρά M19 με επισημασμένες τις ανωμαλίες με Grubb's Test

3.3.6 Isolation Forest

Καταπιαστήκαμε και με μια κλασική μέθοδο ανίχνευσης ανωμαλιών, το Isolation Forest, την οποία έχουμε εξηγήσει εκτενώς σε προηγούμενο κεφάλαιο. Συνοπτικά επαναλαμβάνουμε ότι ο αλγόριθμος λειτουργεί δημιουργώντας ένα δάσος απομονωμένων δέντρων και κάθε δέντρο εκπαιδεύεται σε ένα τυχαίο υποσύνολο δεδομένων. Τα ανώμαλα σημεία αναμένεται να έχουν μικρότερες διαδρομές από τη ρίζα και έτσι να απομονώνονται ταχύτερα από τα κανονικά σημεία.

Στο συγκεκριμένο notebook επιλέξαμε να εφαρμόσουμε τη συγκεκριμένη τεχνική στη χρονοσειρά M19 με και χωρίς προεπεγεργασία. Αρχικά, θα δούμε την εφαρμογή του Isolation Forest με προεπεξεργασία, δηλαδή έχοντας αφαιρέσει ορισμένες ακραίες τιμές από τη M19.

Μερικές από τις παραμέτρους που πρέπει να λάβουμε υπόψη μας στον κώδικα μας είναι:

- max_features → χαρακτηριστικά για την εκπαίδευση κάθε δέντρου (προεπιλογή = 1.0)
- n_estimators → δέντρα που θα χτιστούν στο δάσος (προεπιλογή = 100)
- max_samples = 'auto' → δείγματα που πρέπει να ληφθούν για την εκπαίδευση κάθε δέντρου
- contamination → αναμενόμενο ποσοστό ακραίων τιμών στο σύνολο δεδομένων (οριακό ποσοστό/κλάσμα στη βαθμολογία του δείγματος)

Αρχικοποιούμε το μοντέλο μας με contamination level 5%.
Να σημειωθεί ότι το max_features εφόσον δεν το ορίζουμε διαφορετικά έχει την προεπιλεγμένη τιμή 1.0 και το χαρακτηριστικό με βάση το οποίο θα γίνουν τα partitions είναι αποκλειστικά η στήλη value.

```
#initiate the model with 5% contamination

model = IsolationForest(random_state = 0, contamination = float(0.05))
model.fit(df_1[['value']])
```

```
IsolationForest(contamination=0.05, random_state=0)
```

Στη συνέχεια προσθέτουμε στο dataframe μας τις στήλες anomaly value και score. Το score ουσιαστικά είναι ο αριθμός των διαχωρισμών (partitions) που απαιτούνται για την απομόνωση ενός σημείου (βαθμολογία απομόνωσης). Τα σημεία με χαμηλότερη βαθμολογία απομόνωσης (κάτω από το μέσο score) θεωρούνται πιο μη φυσιολογικά ή ανώμαλα και άρα έχουν anomaly value -1. Τα κανονικά σημεία έχουν anomaly value 1.

```
df_1['score'] = model.decision_function(df_1[['value']])
df_1['anomaly_value'] = model.predict(df_1[['value']])
df_1.head()

#the lower the score (closer to 0), the lesser outlying the datapoint
```

timestamp	cnodeuid	value	type	cnt	Date	score	anomaly_value
2018-02-09 10:30:00	76470	27.000	\N	1	2018-02-09 10:30:00	0.167727	1
2018-02-09 10:45:00	76470	26.250	\N	1	2018-02-09 10:45:00	0.189680	1
2018-02-09 11:00:00	76470	27.125	\N	1	2018-02-09 11:00:00	0.156666	1
2018-02-09 11:15:00	76470	26.625	\N	1	2018-02-09 11:15:00	0.179050	1
2018-02-09 11:30:00	76470	26.750	\N	1	2018-02-09 11:30:00	0.172420	1

Συνολικα 163259 σημεία της χρονοσειράς μας είναι φυσιολογικα και 8562 θεωρούνται ανωμαλίες. Ήδη χωρίς να δούμε και οπτικοποιημένα τα αποτελέσματα, τα ανώμαλα σημεία φαίνονται πολλά, κάτι που μας υποδεικνύει ότι οι παράμετροι μας χρειάζονται ένα fine tuning.

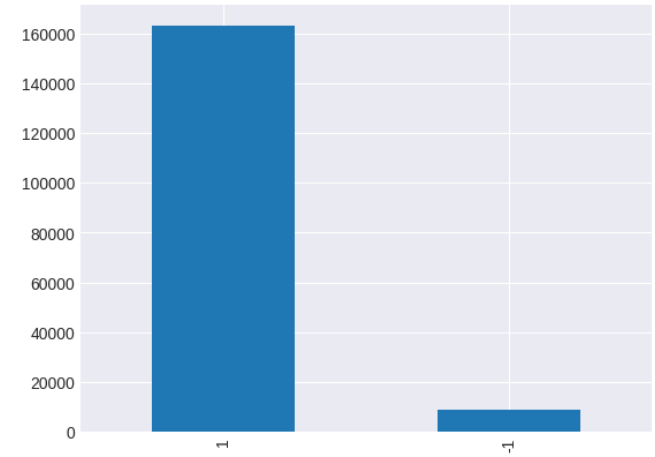
```
outliers = df_1.loc[df_1['anomaly_value'] == -1]
outlier_index = list(outliers.index)

#datapoints classified -1 are anomalous
df_1['anomaly_value'].value_counts()
```

```
 1    163259
-1      8562
Name: anomaly_value, dtype: int64
```

Απλό ραβδόγραμμα:

```
df_1['anomaly_value'].value_counts().plot(kind = 'bar')
```

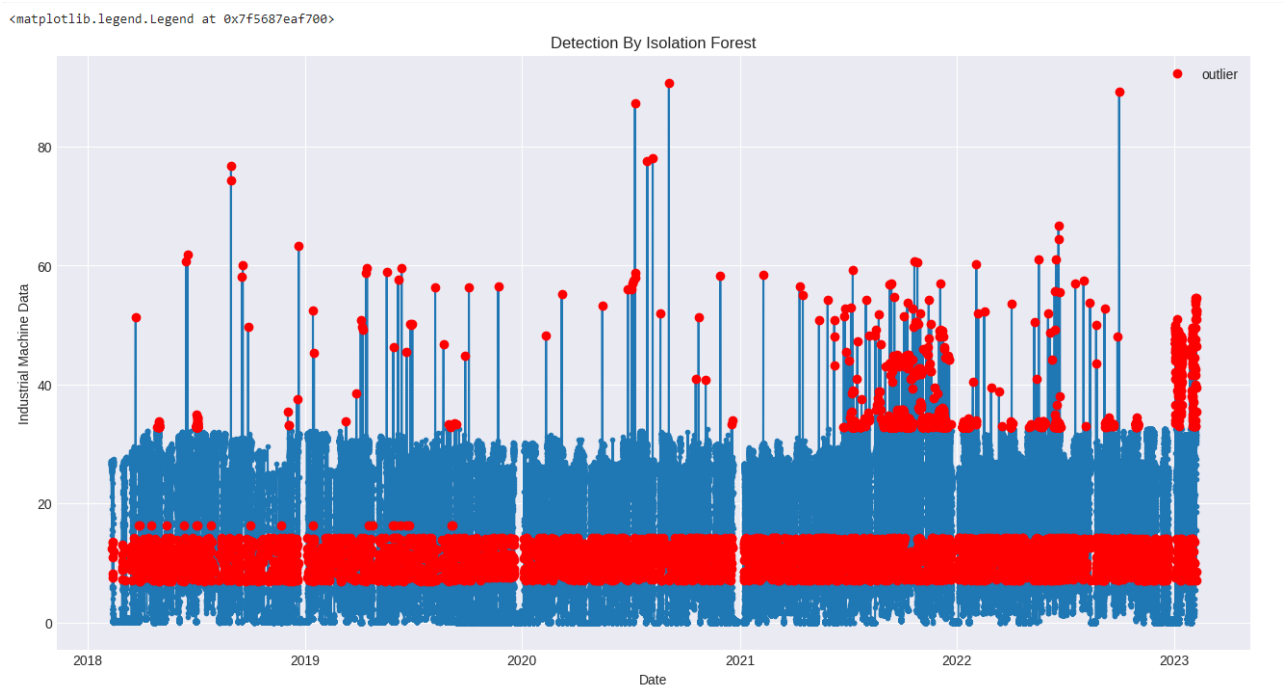
```
<Axes: >
```



Εικόνα 3-15 Απλό ραβδόγραμμα ανώμαλων σημείων (Isolation Forest) [i]

Πιο αναλυτική οπτικοποίηση μέσω απεικόνισης των ανωμαλιών πάνω στη χρονοσειρά:

```
plt.figure(figsize = (16, 8))

plt.plot(df_1['value'], marker = '.')
plt.plot(outliers['value'], 'o', color = 'red', label = 'outlier')
plt.title('Detection By Isolation Forest')

#plt.grid()
plt.xlabel('Date')
plt.ylabel('Industrial Machine Data')
plt.legend()
```



Εικόνα 3-16 Χρονοσειρά M19 με επισημασμένες τις ανωμαλίες με Isolation Forest (i)

Είναι προφανές ότι η προηγούμενη παρατήρηση μας για το πλήθος των ανώμαλων σημείων είναι ορθή και γι' αυτό θα προχωρήσουμε σε fine tuning των παραμέτρων μας για το Isolation Forest, ξεκινώντας από το contamination level, το οποίο αυτή τη φορά θα είναι 1%.

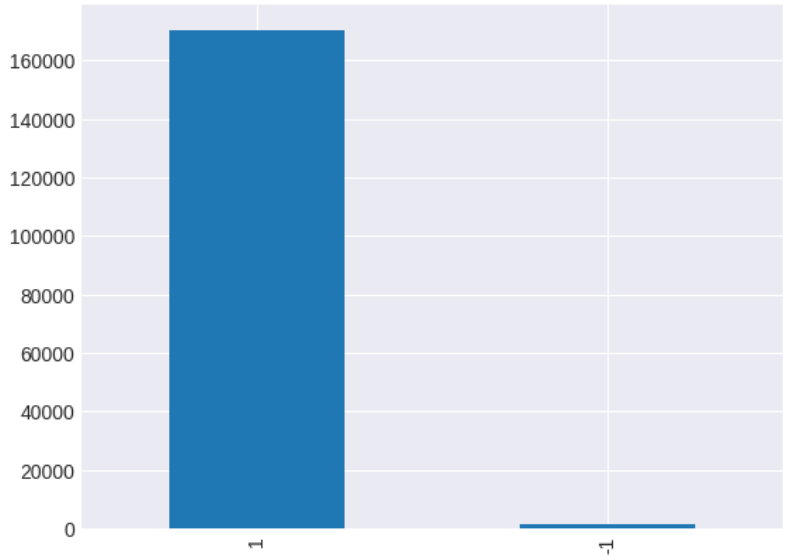
```
model = IsolationForest(random_state = 0, contamination = float(0.01))
model.fit(df_1[['value']])

df_1['score'] = model.decision_function(df_1[['value']])
df_1['anomaly_value'] = model.predict(df_1[['value']])
#df_1.head()

outliers = df_1.loc[df_1['anomaly_value'] == -1]
outlier_index = list(outliers.index)

#datapoints classified -1 are anomalous
df_1['anomaly_value'].value_counts()
```

```
 1    170303
-1      1518
Name: anomaly_value, dtype: int64
```

Αυτή τη φορά τα φυσιολογικά σημεία δεδομένων είναι 170303 και τα ανώμαλα 1518, αισθητά λιγότερα από πριν. Το ραβδόγραμμα μας είναι το εξής:

```
df_1['anomaly_value'].value_counts().plot(kind = 'bar')
```

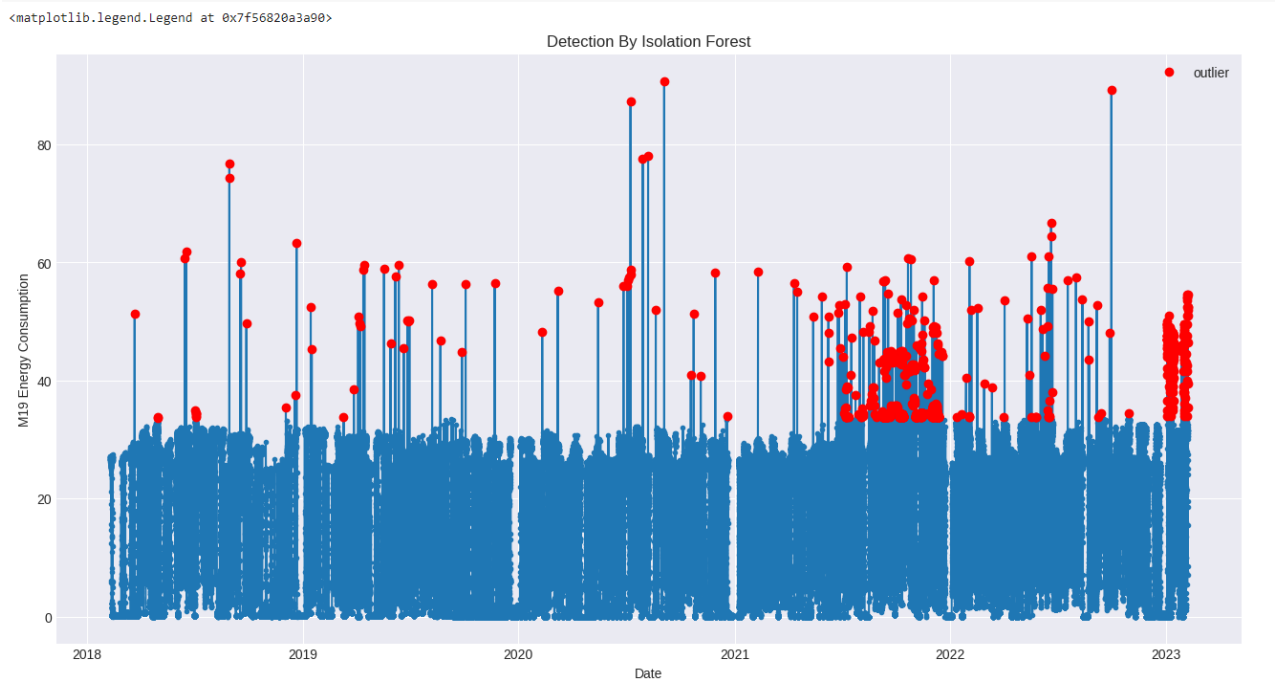
```
<Axes: >
```



Εικόνα 3-17 Απλό ραβδόγραμμα ανώμαλων σημείων (Isolation Forest) [ii]

Και φυσικά η απεικόνιση των δεδομένων μας στη χρονοσειρά:

```
plt.figure(figsize = (16, 8))

plt.plot(df_1['value'], marker = '.')
plt.plot(outliers['value'], 'o', color = 'red', label = 'outlier')
plt.title('Detection By Isolation Forest')

#plt.grid()
plt.xlabel('Date')
plt.ylabel('M19 Energy Consumption')
plt.legend()
```



Εικόνα 3-18 Χρονοσειρά M19 με επισημασμένες τις ανωμαλίες με Isolation Forest (ii)

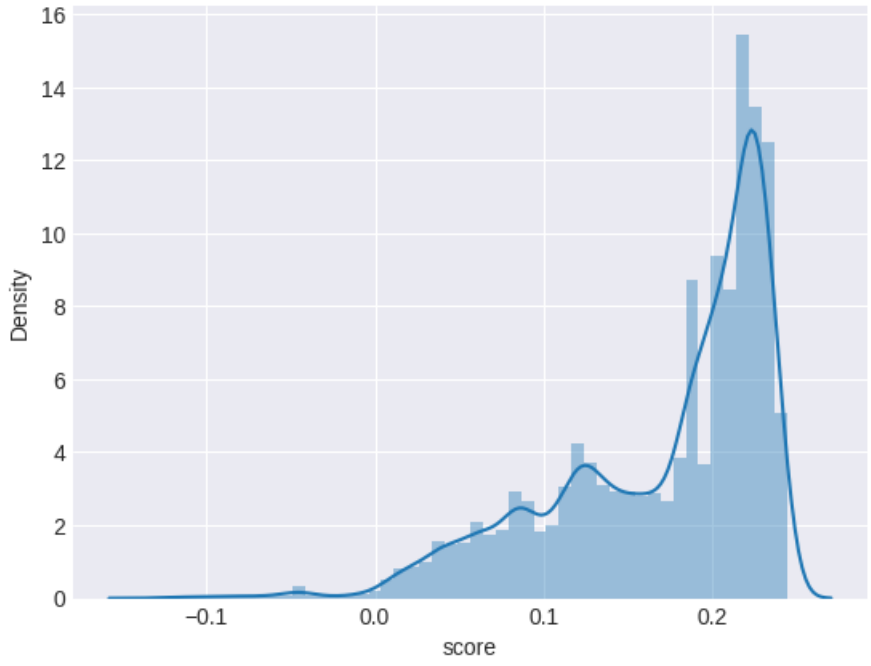
Ξεκάθαρα ο αλγόριθμος Isolation Forest με contamination 1% για τη χρονοσειρά M19, παράγει καλύτερα αποτελέσματα σε σύγκριση με contamination level 5%.

Στη συνέχεια του notebook όμως επιλέξαμε να προσθέσουμε μερικές ακομή στήλες στο dataframe, ώστε να διαπιστώσουμε τι συμβαίνει, όταν το πλήθος των χαρακτηριστικών με βάση τα οποία γίνεται το partition αυξάνονται. Συγκεκριμένα, προσθέσαμε όπως φαίνεται και από τον κώδικα που ακολουθεί το min και το max των τιμών της χρονοσειράς κατά τη διάρκεια μιας ώρας και δύο ωρών.

```
df_4 = df_1.drop(columns=['cnodeuid', 'type', 'cnt'])

# df_4.set_index('timestamp', inplace = True)

df_4['MIN2'] = df_4['value'].resample('2H').min().ffill() #hours granularity
df_4['MAX2'] = df_4['value'].resample('2H').max().ffill() #hours granularity
df_4['MIN1'] = df_4['value'].resample('1H').min().ffill() #hours granularity
df_4['MAX1'] = df_4['value'].resample('1H').max().ffill() #hours granularity
df_4 = df_4.dropna()

columns = ['value','MIN1','MAX1','MIN2','MAX2']
print(df_4.head())
model = IsolationForest(random_state = 0, contamination = 0.01)
model.fit(df_4[columns])
df_4['score'] = model.decision_function(df_4[columns])
df_4['anomaly_value'] = model.predict(df_4[columns])
print(df_4.head())

sns.distplot(df_4['score'])

outliers = df_4.loc[df_4['anomaly_value'] == -1]
outlier_index = list(outliers.index)

#datapoints classified -1 are anomalous
df_4['anomaly_value'].value_counts()
```

```
                     value                Date     score  anomaly_value  \
timestamp
2018-02-09 12:00:00  27.000 2018-02-09 12:00:00  0.204700              1
2018-02-09 14:00:00  26.750 2018-02-09 14:00:00  0.209394              1
2018-02-09 16:00:00  24.875 2018-02-09 16:00:00  0.213038              1
2018-02-09 18:00:00  27.000 2018-02-09 18:00:00  0.204700              1
2018-02-09 20:00:00  26.500 2018-02-09 20:00:00  0.222873              1

                      MIN2    MAX2    MIN1    MAX1
timestamp
2018-02-09 12:00:00  26.500  27.125  26.500  27.000
2018-02-09 14:00:00  24.125  27.000  26.625  27.000
2018-02-09 16:00:00  24.875  27.000  24.875  27.000
2018-02-09 18:00:00  26.500  27.000  26.500  27.000
2018-02-09 20:00:00  26.500  27.125  26.500  27.125
                     value                Date     score  anomaly_value  \
timestamp
2018-02-09 12:00:00  27.000 2018-02-09 12:00:00  0.214557              1
2018-02-09 14:00:00  26.750 2018-02-09 14:00:00  0.216492              1
2018-02-09 16:00:00  24.875 2018-02-09 16:00:00  0.228545              1
2018-02-09 18:00:00  27.000 2018-02-09 18:00:00  0.213235              1
2018-02-09 20:00:00  26.500 2018-02-09 20:00:00  0.223162              1

                      MIN2    MAX2    MIN1    MAX1
timestamp
2018-02-09 12:00:00  26.500  27.125  26.500  27.000
2018-02-09 14:00:00  24.125  27.000  26.625  27.000
2018-02-09 16:00:00  24.875  27.000  24.875  27.000
2018-02-09 18:00:00  26.500  27.000  26.500  27.000
2018-02-09 20:00:00  26.500  27.125  26.500  27.125
 1    21275
-1      215
Name: anomaly_value, dtype: int64
```

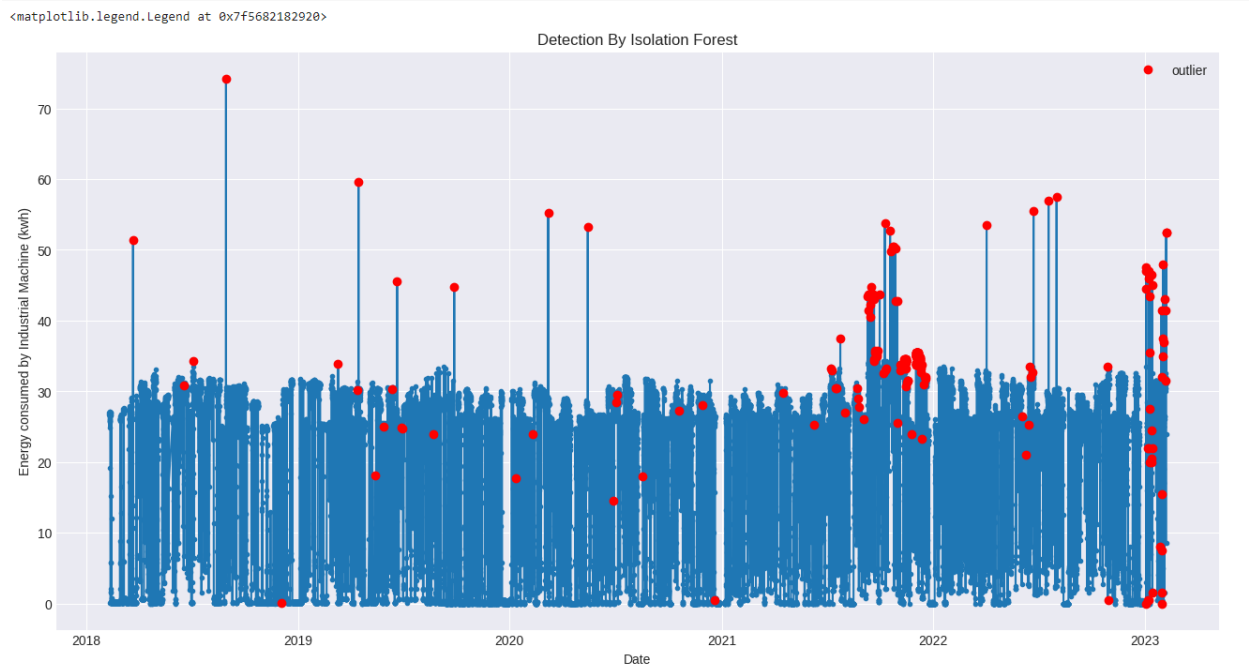
Επιλέξαμε να φτιάξουμε κι ένα διάγραμμα πυκνότητας του score.



Εικόνα 3-19 Διάγραμμα πυκνότητας score

Καθώς επίσης κι ένα διάγραμμα στο οποίο αεπικονίζονται με κόκκινο τα ανώμαλα σημεία, τα οποία εντόπισε ο αλγόριθμος Isolation Forest με contamination 1% και features τα value, MIN1, MAX1, MIN2 και MAX2.

```
plt.figure(figsize = (16, 8))

plt.plot(df_4['value'], marker = '.')
plt.plot(outliers['value'], 'o', color = 'red', label = 'outlier')
plt.title('Detection By Isolation Forest')

#plt.grid()
plt.xlabel('Date')
plt.ylabel('Energy consumed by Industrial Machine (kwh)')
plt.legend()
```



Εικόνα 3-20 Χρονοσειρά M19 με επισημασμένες τις ανωμαλίες με Isolation Forest (iii)

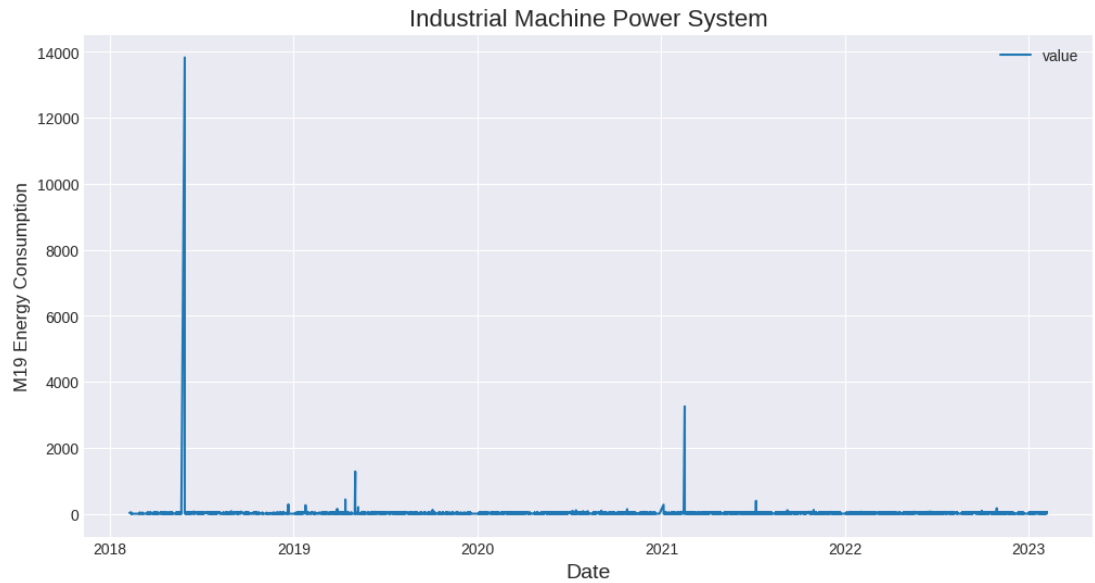
Παρατηρούμε ότι τελικά με αυτές τις παραμέτρους ο αλγόριθμος μας δίνει πολύ καλά αποτελέσματα.

Απομένει να δούμε τι αποτελέσματα δίνει το Isolation Forest όταν δεν έχουμε προεπεξεργαστεί τα δεδομένα μας, αφαιρώντας κάποιες ακραίες τιμές της χρονοσειράς M19 εκ των προτέρων.

Η χρονοσειρά μας όπως έχουμε δείξει και πιο πριν έχει αυτή τη μορφή:

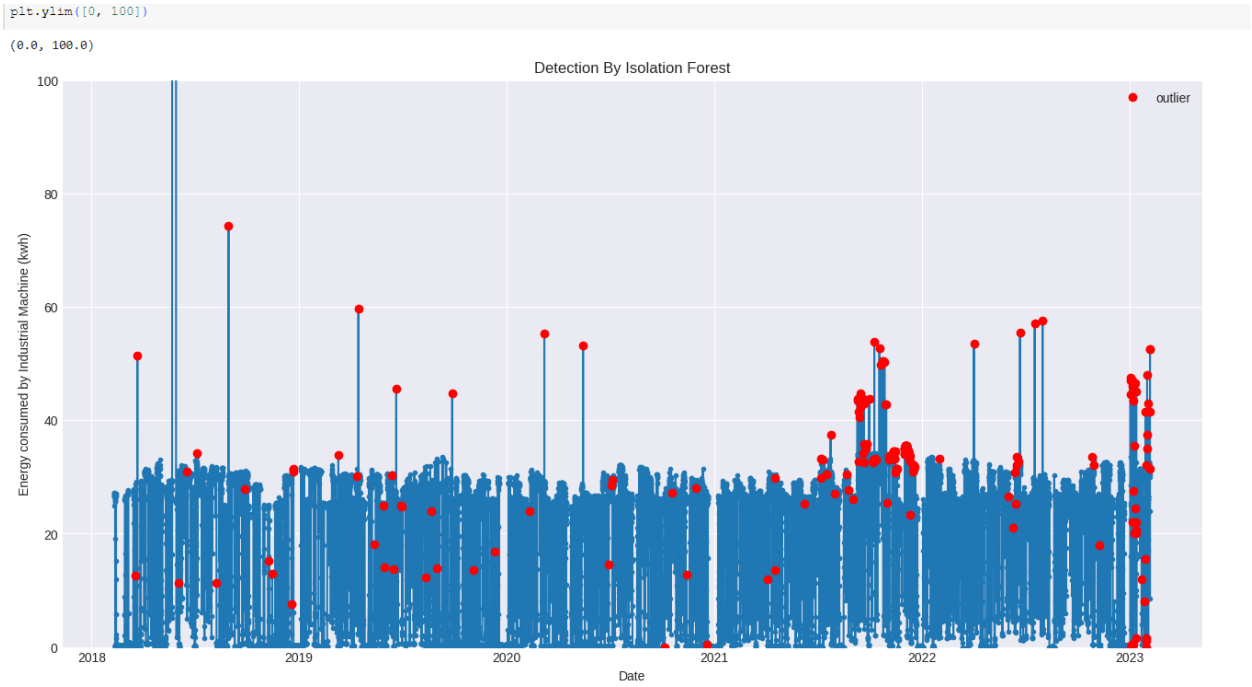
```
plot_neutral(df_2)
```

	count	mean	std	min	25%	50%	75%	max
value	171839.0	18.699019	36.57633	0.0	2.0	25.25	28.375	13828.375



Εικόνα 3-21 Χρονοσειρά M19 χωρίς προεπεξεργασία (ii)

Και ύστερα από την εφαρμογή Isolation Forest με contamination 1% και features τα value, MIN1, MAX1, MIN2 και MAX2, το διάγραμμα στο οποίο είναι σημειωμένες οι ανώμαλες τιμές είναι:



Εικόνα 3-22 Χρονοσειρά M19 με επισημασμένες τις ανωμαλίες με Isolation Forest (iv)

Να σημειωθεί ότι η γραμμή κώδικα `plt.ylim([0, 100])` προστέθηκε ώστε να είναι πιο κατανοητό το διάγραμμα διότι λόγω του εξαιρετικά μεγάλου outlier που υπάρχει δεν θα μπορούσαμε να διακρίνουμε τα υπόλοιπα ανώμαλα σημεία.

3.3.7 LOF

Καταπιαστήκαμε και με μια άλλη κλασική μέθοδο ανίχνευσης ανωμαλιών, το Local Outlier Factor, την οποία έχουμε εξηγήσει εκτενώς σε προηγούμενο κεφάλαιο.

Ακολουθεί μια εξήγηση των παραμέτρων που χρησιμοποιούνται στον αλγόριθμο LOF:

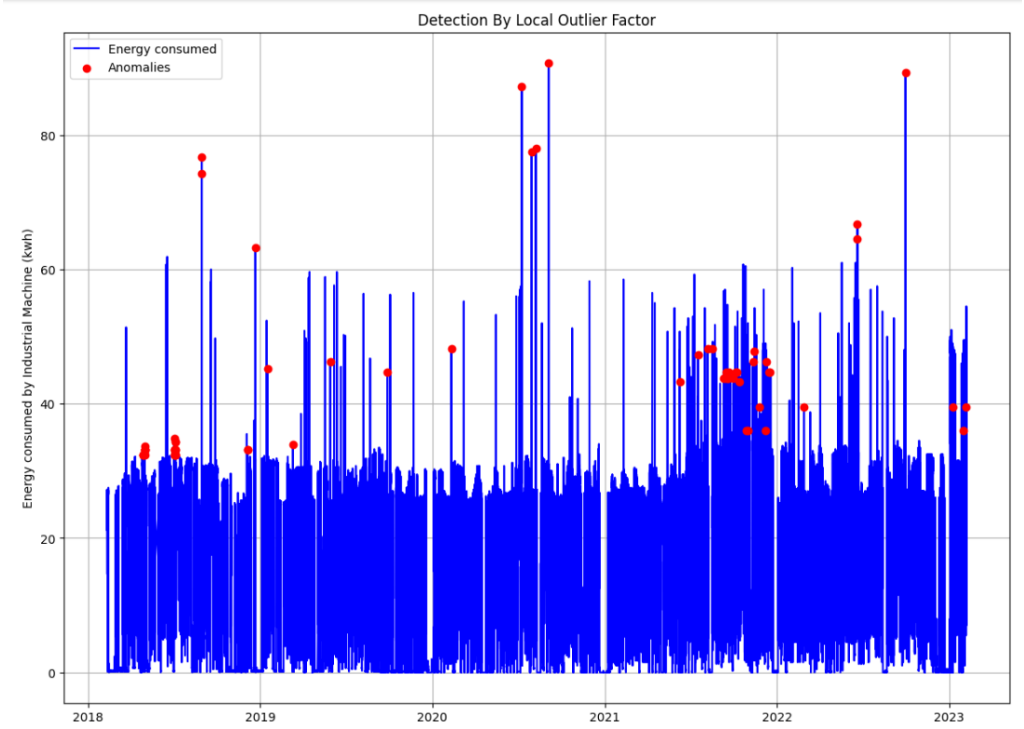
n_neighbors: Ο αριθμός των γειτόνων που χρησιμοποιούνται για τον υπολογισμό της τοπικής πυκνότητας κάθε σημείου δεδομένων. Η αύξηση αυτής της τιμής θα κάνει τον αλγόριθμο λιγότερο ευαίσθητο σε μεμονωμένα σημεία δεδομένων, αλλά μπορεί επίσης να οδηγήσει σε απώλεια ευαισθησίας σε τοπικά ακραία σημεία. Ένας καλός εμπειρικός κανόνας είναι να ορίσουμε αυτήν την παράμετρο στη μικρότερη τιμή που εξακολουθεί να καταγράφει την τοπική δομή των δεδομένων.

contamination: Η αναμενόμενη αναλογία ακραίων τιμών στα δεδομένα. Αυτή η παράμετρος χρησιμοποιείται για να ορίσει ένα όριο στη βαθμολογία LOF, κάτω από την οποία τα σημεία δεδομένων θεωρούνται ακραία. Η προεπιλεγμένη τιμή μόλυνσης είναι 0,1, η οποία προϋποθέτει ότι το 10% των σημείων δεδομένων είναι ακραίες τιμές. Ωστόσο, προσαρμόζουμε αυτήν την παράμετρο με βάση τα συγκεκριμένα χαρακτηριστικά των δεδομένων μας.

Γενικά, θα πρέπει πρώτα να ορίσουμε την παράμετρο n_neighbors με βάση την τοπική δομή των δεδομένων μας και, στη συνέχεια, να προσαρμόσουμε το contamination για να επιτύχουμε ένα επιθυμητό επίπεδο ευαισθησίας σε ακραίες τιμές. Ωστόσο, είναι σημαντικό να σημειωθεί ότι ο αλγόριθμος LOF δεν είναι πάντα εγγυημένο ότι θα λειτουργεί καλά σε όλους τους τύπους δεδομένων και μπορεί να χρειαστεί να πειραματιστούμε με διαφορετικές παραμέτρους και μεθόδους ανίχνευσης ακραίων τιμών για να βρούμε την καλύτερη προσέγγιση για τη συγκεκριμένη περίπτωση χρήσης.

```
import pandas as pd
from sklearn.neighbors import LocalOutlierFactor

def detect_anomalies_with_local_outlier(series):
    lof = LocalOutlierFactor(n_neighbors=10, contamination='auto')
    #lof = LocalOutlierFactor(n_neighbors=40, contamination=0.01)
    X = series.values.reshape(-1,1)
    y_pred = lof.fit_predict(X)
    anomalies = X[y_pred==-1]
    return pd.Series(anomalies.flatten(), index=series.index[y_pred==-1])



# Detect anomalies using the Isolation Forest algorithm
anomalies = detect_anomalies_with_local_outlier(series)

# Plot the original series and the detected anomalies
plt.subplots(figsize=(14, 10))
plt.plot(df['timestamp'], df['value'], color='blue', label='Temperature Readings')
plt.scatter(anomalies.index, anomalies.values, color='red', label='Anomalies',zorder=2000)
plt.legend()
plt.title('Machine Temperature Anomaly Detection - Local Outlier Factor')
plt.xlabel('Date')
plt.ylabel('Temperature (Celsius)')
plt.grid()
plt.show()
```

Το διάγραμμα στο οποίο είναι σημειωμένες οι ανώμαλες τιμές είναι:

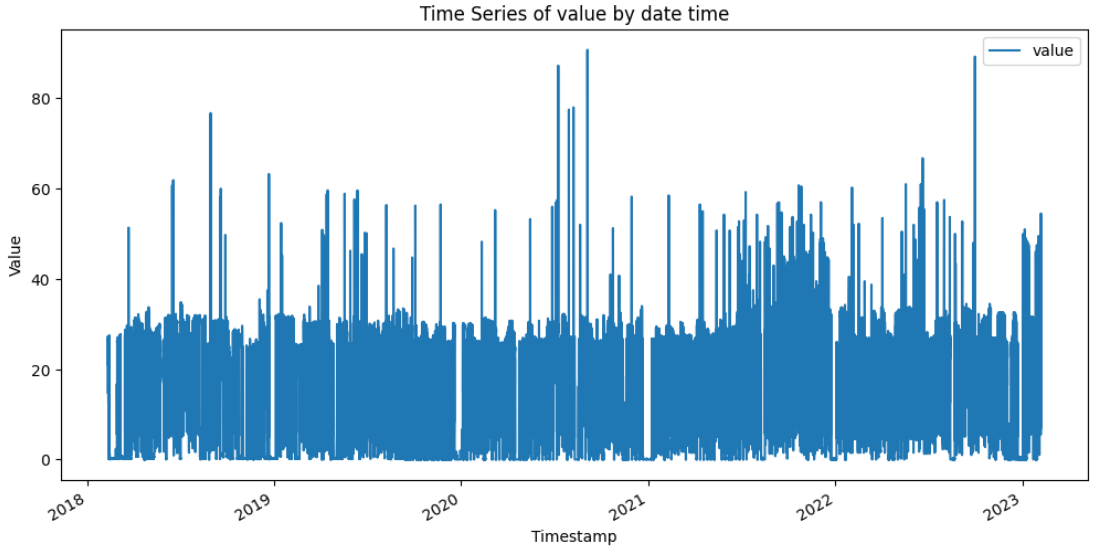


3.3.8 LSTM

Χρησιμοποιήσαμε LSTM (Long Short-Term Memory) έναν τύπο επαναλαμβανόμενου νευρωνικού δικτύου (RNN) που έχει σχεδιαστεί ειδικά για να χειρίζεται μακροπρόθεσμες εξαρτήσεις σε διαδοχικά δεδομένα.

```
print(df.head(3))
df.shape
df.plot(x='timestamp', y='value', figsize=(12,6))
plt.xlabel('Timestamp')
plt.ylabel('Value')
plt.title('Time Series of value by date time')
```

```
   cnodeuid           timestamp   value type  cnt
0     76470 2018-02-09 10:30:00  27.000   \N    1
1     76470 2018-02-09 10:45:00  26.250   \N    1
2     76470 2018-02-09 11:00:00  27.125   \N    1
Text(0.5, 1.0, 'Time Series of value by date time')
```



Εικόνα 3-23 Χρονοσειρά M19 χωρίς προεπεξεργασία

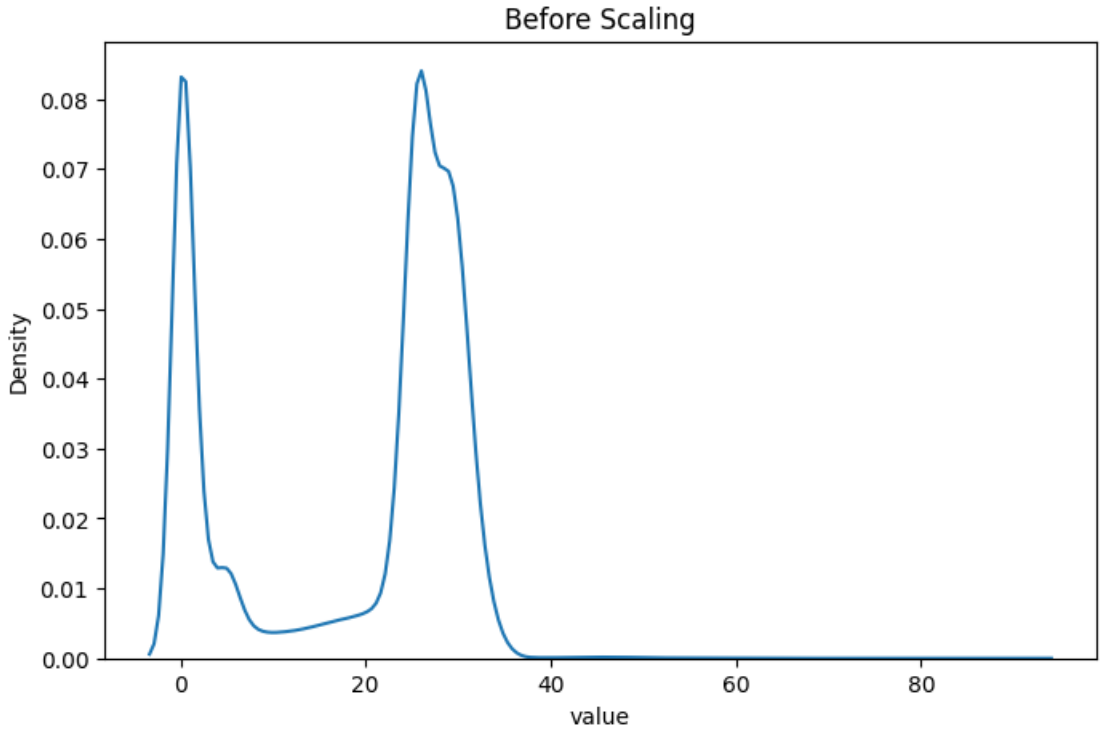
```
df.value.describe()
```

```
count    171821.000000
mean         18.575618
std          12.405307
min           0.000000
25%           2.000000
50%          25.250000
75%          28.375000
max          90.750000
Name: value, dtype: float64
```

Για να κατανοήσουμε τα δεδομένα μας επιλέξαμε εκτός από την απλή απεικόνιση της χρονοσειράς και την περιγραφή τους, να τα απεικονίσουμε χρησιμοποιώντας KDE (Kernel Density Estimate) διαγράμματα, πριν και μετά από scaling.

```
fig, (ax1) = plt.subplots(ncols=1, figsize=(8, 5))
ax1.set_title('Before Scaling')
sns.kdeplot(df['value'], ax=ax1)
```

```
<Axes: title={'center': 'Before Scaling'}, xlabel='value', ylabel='Density'>
```



Εικόνα 3-24 Πριν από scaling

Τα σημεία δεδομένων μας έχουν ελάχιστη τιμή 0 και μέγιστη 90.75, το οποίο είναι ένα αρκετά μεγάλο εύρος. Για να κανονικοποιήσουμε τα δεδομένα μας, θα εφαρμόσουμε scaling με την εξής φόρμουλα $(x - Min)/(Max - Min)$

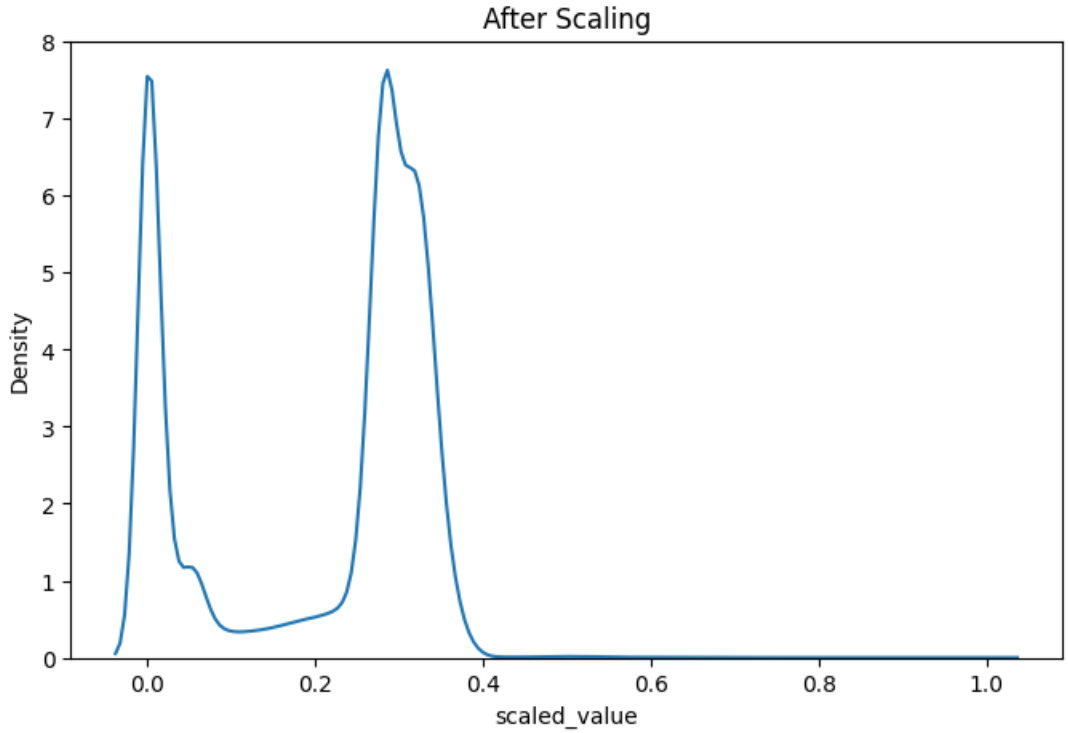
```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler(feature_range = (0, 1))
df['scaled_value'] = pd.DataFrame(scaler.fit_transform(pd.DataFrame(df['value'])),columns=['value'])
print('Shape:' , df.shape[0])
df.head(5)
```

```
Shape: 171821
```

	cnodeuid	timestamp	value	type	cnt	scaled_value
0	76470	2018-02-09 10:30:00	27.000	\N	1	0.297521
1	76470	2018-02-09 10:45:00	26.250	\N	1	0.289256
2	76470	2018-02-09 11:00:00	27.125	\N	1	0.298898
3	76470	2018-02-09 11:15:00	26.625	\N	1	0.293388
4	76470	2018-02-09 11:30:00	26.750	\N	1	0.294766

```
fig, (ax1) = plt.subplots(ncols=1, figsize=(8, 5))
ax1.set_title('After Scaling')
sns.kdeplot(df['scaled_value'], ax=ax1)
```

```
<Axes: title={'center': 'After Scaling'}, xlabel='scaled_value', ylabel='Density'>
```



Εικόνα 3-25 Μετά από scaling

Υπάρχουν 171821 σημεία δεδομένων στην ακολουθία και ο στόχος μας είναι να βρούμε ανώμαλα σημεία. Αυτό σημαίνει ότι προσπαθούμε να μάθουμε πότε τα σημεία δεδομένων είναι μη φυσιολογικά. Εάν μπορούμε να προβλέψουμε ένα σημείο δεδομένων τη χρονική στιγμή T με βάση τα ιστορικά δεδομένα μέχρι το T-1, τότε συγκρίνοντας την αναμενόμενη τιμή με την πραγματική τιμή, μπορούμε να δούμε αν είμαστε εντός του αναμενόμενου εύρους τιμών για το χρόνο T. Με το να προβλέψουμε ότι η ενεργειακή κατανάλωση του M19 έχει τιμη y_{pred}, μπορούμε να συγκρίνουμε το y_{pred} με το πραγματικό y_{actual}. Η διαφορά μεταξύ τους δίνει το σφάλμα, και όταν λαμβάνουμε τα σφάλματα όλων των σημείων στην ακολουθία, καταλήγουμε με μια κατανομή σφαλμάτων.

Αυτό το πετύχαμε μέσω ενός διαδοχικού μοντέλου χρησιμοποιώντας το Keras. Το μοντέλο αποτελείται από 2 στρώματα LSTM και ένα dense layer. Το επίπεδο LSTM λαμβάνει ως είσοδο τα δεδομένα χρονοσειράς και εκπαιδεύεται στο να μαθαίνει τις αξίες σε σχέση με τον χρόνο. Το επόμενο στρώμα (dense layer - πλήρως συνδεδεμένο στρώμα) παίρνει ως είσοδο την έξοδο από το στρώμα LSTM και το μετατρέπει σε πλήρως συνδεδεμένο. Στη συνέχεια, εφαρμόζεται μια σιγμοειδής συνάρτηση ενεργοποίησης στο dense layer έτσι ώστε η τελική έξοδος να είναι μεταξύ 0 και 1. Χρησιμοποιείται επίσης το adam optimizer και το μέσο τετραγωνικό σφάλμα (MSE – Mean Squared Error) ως συνάρτηση απώλειας.

```
time_steps = 48
metric = 'mean_absolute_error'

model = Sequential()
model.add(LSTM(units=32, activation='tanh', input_shape=(time_steps, 1), return_sequences=True))
model.add(LSTM(units=32, activation='tanh', return_sequences=True))

model.add(Dense(1, activation='sigmoid'))

model.compile(optimizer='adam', loss='mean_absolute_error', metrics=[metric])
print(model.summary())
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm (LSTM)                 (None, 48, 32)            4352

 lstm_1 (LSTM)               (None, 48, 32)            8320

 dense (Dense)               (None, 48, 1)             33

=================================================================
Total params: 12,705
Trainable params: 12,705
Non-trainable params: 0
_________________________________________________________________
None
```

```
sequence = np.array(df['scaled_value'])
print(sequence)
time_steps = 48
samples = len(sequence)
trim = samples % time_steps
subsequences = int(samples/time_steps)
sequence_trimmed = sequence[:samples - trim]

print(samples, subsequences)
sequence_trimmed.shape = (subsequences, time_steps, 1)
print(sequence_trimmed.shape)
```

```
[0.29752066 0.2892562  0.29889807 ... 0.34710744 0.33608815 0.3415978 ]
171821 3579
(3579, 48, 1)
```

Εκπαιδεύσαμε το μοντέλο μας για 80 epochs, χρησιμοποιώντας τα δεδομένα εκπαίδευσης και ως δεδομένα επαλήθευσης.

```
training_dataset = sequence_trimmed[:-700]
print("training_dataset: ", training_dataset.shape)

batch_size=32
epochs=80

model.fit(x=training_dataset, y=training_dataset,
                      batch_size=batch_size, epochs=epochs,
                      verbose=1, validation_data=(sequence_trimmed[-700:], sequence_trimmed[-700:]),
                      callbacks=[keras.callbacks.EarlyStopping(monitor="val_loss", patience=6, restore_best_weights=True)])
```

ndarray: training_dataset

ndarray with shape (2879, 48, 1)

```
training_dataset:  (2879, 48, 1)
Epoch 1/80
90/90 [==============================] - 11s 18ms/step - loss: 0.1586 - mean_absolute_error: 0.1586 - val_loss: 0.0811 - val_mean_absolute_error: 0.0811
Epoch 2/80
90/90 [==============================] - 1s 13ms/step - loss: 0.0640 - mean_absolute_error: 0.0640 - val_loss: 0.0478 - val_mean_absolute_error: 0.0478
```

Αφού εκπαιδεύσαμε το μοντέλο, ήρθε η ώρα να δούμε πόσο αποτελεσματικά μπορούμε να το εφαρμόσουμε για προβλέψεις και ανίχνευση ανωμαλιών σε ένα σύνολο δεδομένων δοκιμής το οποίο έχουμε χωρίσει σε υποακολουθίες του ίδιου μήκους (timesteps) με το σύνολο δεδομένων εκπαίδευσης.

Ύστερα υπολογίσαμε το ριζικό μέσο τετραγωνικό σφάλμα (RMSE – Root Mean Squared Error).

```
import math
from sklearn.metrics import mean_squared_error

sequence = np.array(df['scaled_value'])
print(sequence)
time_steps = 48
samples = len(sequence)
trim = samples % time_steps
print(samples)
subsequences = int(samples/time_steps)
sequence_trimmed = sequence[:samples - trim]

print(samples, subsequences)
sequence_trimmed.shape = (subsequences, time_steps, 1)
print(sequence_trimmed.shape)

testing_dataset = sequence_trimmed
print("testing_dataset: ", testing_dataset.shape)

testing_pred = model.predict(x=testing_dataset)
print("testing_pred: ", testing_pred.shape)

testing_dataset = testing_dataset.reshape((testing_dataset.shape[0]*testing_dataset.shape[1]), testing_dataset.shape[2])
print("testing_dataset: ", testing_dataset.shape)

testing_pred = testing_pred.reshape((testing_pred.shape[0]*testing_pred.shape[1]), testing_pred.shape[2])
print("testing_pred: ", testing_pred.shape)
errorsDF = testing_dataset - testing_pred
print(errorsDF.shape)
rmse = math.sqrt(mean_squared_error(testing_dataset, testing_pred))
print('Test RMSE: %.3f' % rmse)
```

```
[0.29752066 0.2892562  0.29889807 ... 0.34710744 0.33608815 0.3415978 ]
171821
171821 3579
(3579, 48, 1)
testing_dataset:  (3579, 48, 1)
112/112 [==============================] - 1s 4ms/step
testing_pred:  (3579, 48, 1)
testing_dataset:  (171792, 1)
testing_pred:  (171792, 1)
(171792, 1)
Test RMSE: 0.016
```

Το RMSE είναι 0.016, τιμή αρκετά χαμηλή, και αυτό φαίνεται επίσης από τη χαμηλή απώλεια στη φάση εκπαίδευσης μετά από 80 epochs:
`loss: 0.0051 - mean_absolute_error: 0.0051 - val_loss: 0.0054 - val_mean_absolute_error: 0.0054`

Έχοντας το προβλεπόμενο σύνολο δεδομένων και το σύνολο δεδομένων δοκιμών, υπολογίσαμε τη διαφορά ως *diff*, η οποία στη συνέχεια μεταβιβάστηκε μέσω διανυσματικών κανόνων. Στη συνέχεια, ταξινομήσαμε τις βαθμολογίες/διαφορές και χρησιμοποιώντας μια τιμή αποκοπής επιλέξαμε το όριο (threshold). Αυτό προφανώς μπορεί να αλλάξει ανάλογα με τις παραμέτρους μας, ιδιαίτερα την τιμή αποκοπής (η οποία τώρα είναι 0.999). Παρακάτω επίσης φαίνεται ο κώδικας για τον υπολογισμό του ορίου.

```
#based on cutoff after sorting errors
dist = np.linalg.norm(testing_dataset - testing_pred, axis=-1)
#dist = np.linalg.norm((testing_dataset - testing_pred) / (testing_dataset+testing_pred), axis=-1)

scores =dist.copy()
print(scores.shape)
scores.sort()
cutoff = int(0.999 * len(scores))
print(cutoff)
#print(scores[cutoff:])
threshold= scores[cutoff]
print(threshold)
```

```
(171792,)
171620
0.1505192071199417
```

Πήραμε 0.151 ως όριο και οποιαδήποτε μεγαύτερη τιμή θεωρείται ανωμαλία.
Παρακάτω φτιάξαμε μια γραφική παράσταση του συνόλου δεδομένων δοκιμής (testing dataset) και του αντίστοιχου προβλεπόμενου συνόλου δεδομένων (predicted dataset).

```
plt.figure(figsize=(14,8))
plt.plot(testing_dataset, color='green')
plt.plot(testing_pred, color='yellow')
```

```
[<matplotlib.lines.Line2D at 0x7fc54e841030>]
```

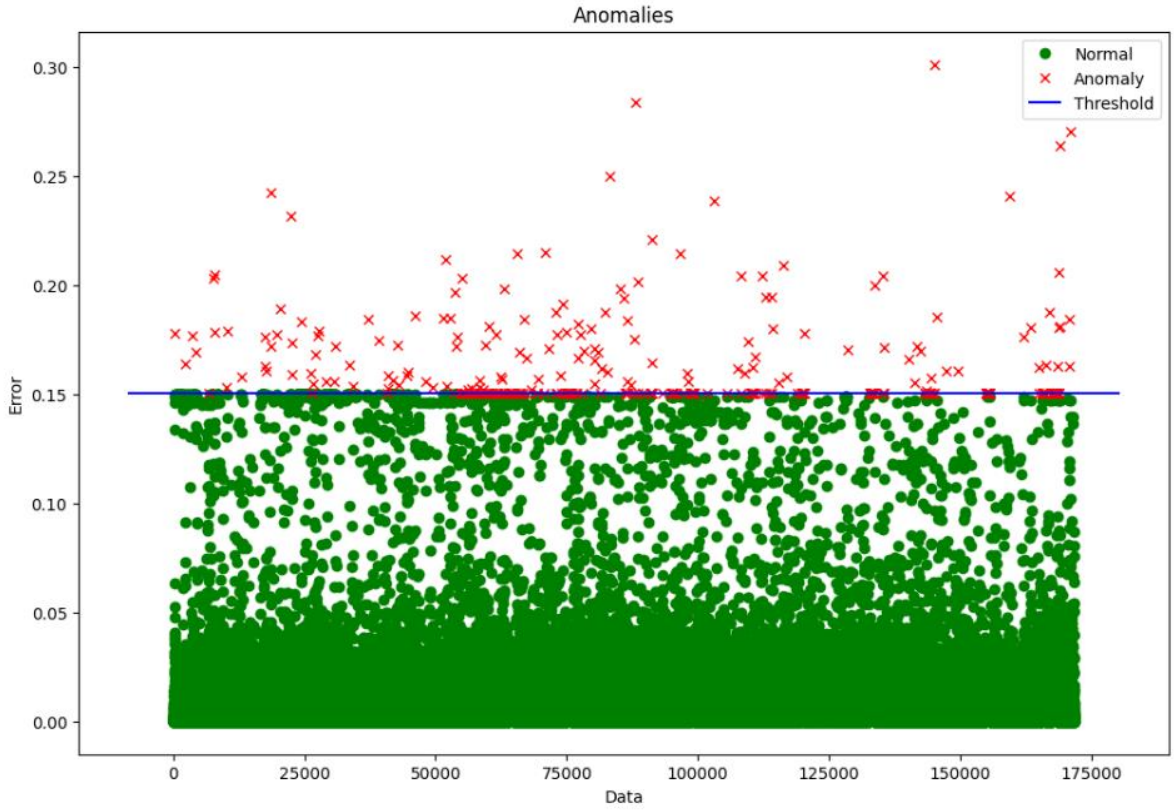
Εικόνα 3-16 Κοινή απεικόνιση testing & prdicted dataset

Ύστερα ταξινομήσαμε τα σημεία δεδομένων ως ανώμαλα ή κανονικά.

```
#label the records anomalies or not based on threshold
z = zip(dist >= threshold, dist)

y_label=[]
error = []
for idx, (is_anomaly, dist) in enumerate(z):
    if is_anomaly:
        y_label.append(1)
    else:
        y_label.append(0)
    error.append(dist)
```

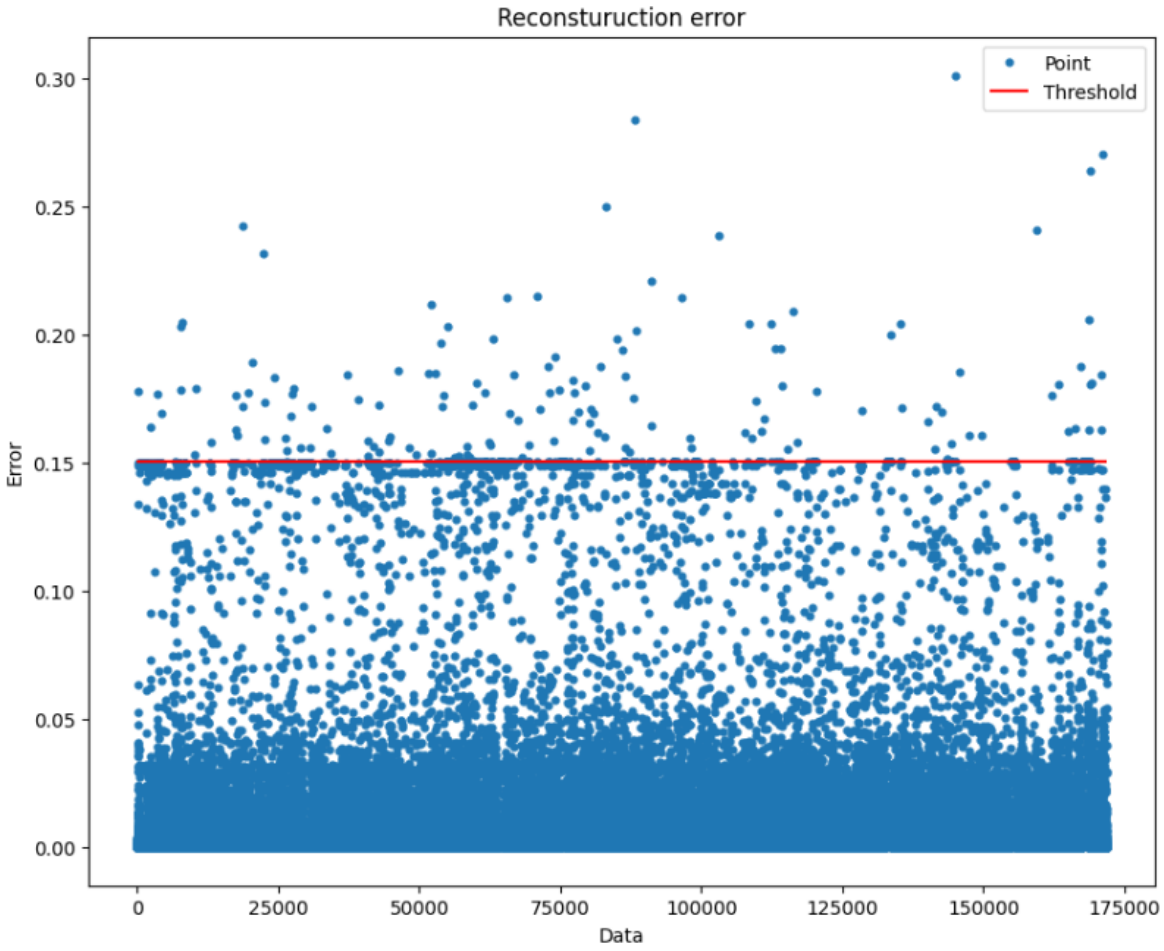
Και τα οπτικοποιήσαμε σε συνάρτηση με το όριο:

```
viz = Visualization()
viz.draw_anomaly(y_label, error, threshold)
```



Εικόνα 3-27 Οπτικοποίηση Ανωμαλιών

```
viz.draw_error(error, threshold)
```



Εικόνα 3-28 Reconstruction Error

Τέλος απεικονίσαμε πανω στο διάγραμμα της χρονοσειράς τα ανώμαλα σημεία, αφού υπολογίσαμε το πλήθος τους.

```
adf = pd.DataFrame({'Datetime': df.iloc[:-trim]['timestamp'],
                    'observation': df.iloc[:-trim]['value'],
                    'scaled_value': df.iloc[:-trim]['scaled_value'],
                    'error': error, 'anomaly': y_label,
                    'predicted_scaled':testing_pred.reshape(-1),
                    'predicted':scaler.inverse_transform(testing_pred).reshape(-1)})
adf.head(5)
```

	Datetime	observation	scaled_value	error	anomaly	predicted_scaled	predicted
0	2018-02-09 10:30:00	27.000	0.297521	0.002355	0	0.295166	26.786310
1	2018-02-09 10:45:00	26.250	0.289256	0.002105	0	0.287151	26.058968
2	2018-02-09 11:00:00	27.125	0.298898	0.000277	0	0.299175	27.150154
3	2018-02-09 11:15:00	26.625	0.293388	0.000115	0	0.293273	26.614555
4	2018-02-09 11:30:00	26.750	0.294766	0.001181	0	0.295947	26.857176

```
adf['anomaly'].value_counts()
```

```
0    171400
1       392
Name: anomaly, dtype: int64
```

```
anomaliesDF = adf.query('anomaly == 1')
anomaliesDF
```

	Datetime	observation	scaled_value	error	anomaly	predicted_scaled	predicted
251	2018-02-12 01:15:00	19.625	0.216253	0.177886	1	0.038368	3.481872
2294	2018-03-05 13:00:00	16.250	0.179063	0.163964	1	0.015099	1.370250
3603	2018-03-19 04:15:00	19.625	0.216253	0.177026	1	0.039228	3.559923
4409	2018-03-27 15:00:00	16.125	0.177686	0.169350	1	0.008335	0.756443
6816	2018-04-21 16:45:00	0.000	0.000000	0.150519	1	0.150519	13.659618
...	...	...	...	...	...	...	...

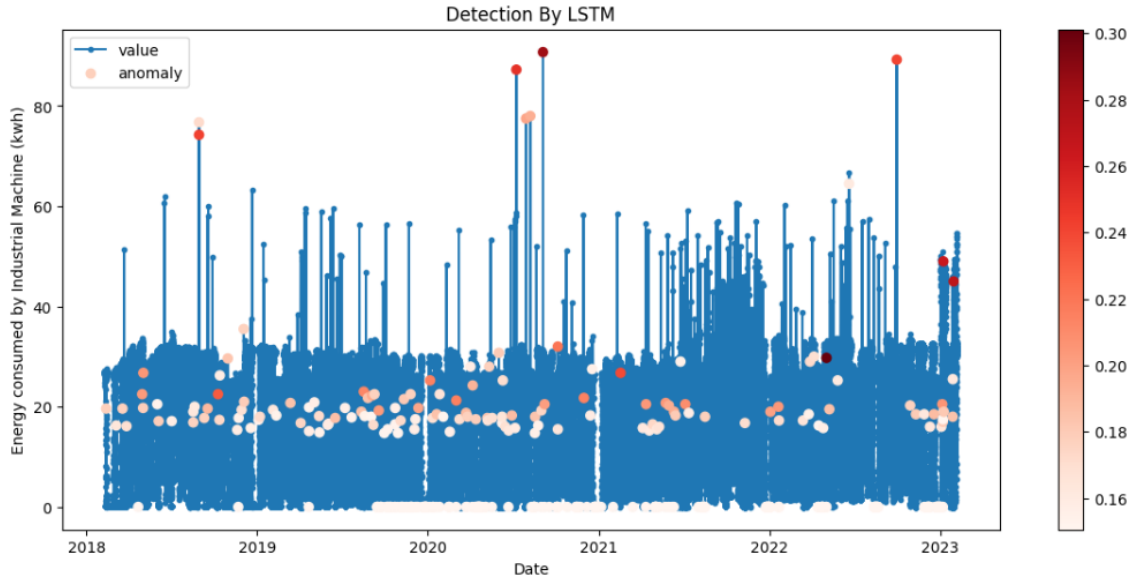
```
anomaliesDF[anomaliesDF['Datetime']>pd.Timestamp('2019-01-04')]
```

	Datetime	observation	scaled_value	error	anomaly	predicted_scaled	predicted
30631	2019-01-04 01:45:00	17.375	0.191460	0.155915	1	0.035545	3.225682
30918	2019-01-07 01:30:00	18.000	0.198347	0.171701	1	0.026646	2.418103
33588	2019-02-04 04:00:00	19.375	0.213499	0.163187	1	0.050311	4.565762
34254	2019-02-11 02:30:00	18.250	0.201102	0.153627	1	0.047475	4.308312
37153	2019-03-13 07:15:00	20.750	0.228650	0.184118	1	0.044532	4.041299
...	...	...	...	...	...	...	...
169003	2023-01-08 04:45:00	49.000	0.539945	0.264045	1	0.275900	25.037920
169095	2023-01-09 03:45:00	19.000	0.209366	0.181320	1	0.028046	2.545213
170924	2023-01-28 05:00:00	18.000	0.198347	0.184278	1	0.014069	1.276769
170948	2023-01-28 11:00:00	25.500	0.280992	0.162523	1	0.118469	10.751018
171114	2023-01-30 04:30:00	45.000	0.495868	0.270390	1	0.225477	20.462063

363 rows × 7 columns

```
plt.plot(df['timestamp'], df['value'], marker = '.', label = 'value')
plt.scatter(anomaliesDF['Datetime'], anomaliesDF['observation'], label = 'anomaly',zorder=1000, c=anomaliesDF['error'], cmap='Reds')
#plt.plot(adf['Datetime'], adf['predicted'], color = 'green', label = 'predicted')
#plt.plot(anomaliesDF['Datetime'], anomaliesDF['predicted'], 'x', color = 'orange', label = 'predicted')
plt.title('Detection By LSTM')
plt.colorbar()

plt.xlabel('Date')
plt.ylabel('Energy consumed by Industrial Machine (kwh)')
plt.legend()
```

```
<matplotlib.legend.Legend at 0x7fc5ed207c70>
```



Εικόνα 3-29 Χρονοσειρά M19 με επισημασμένες τις ανωμαλίες με LSTM

3.3.9 RNN – Autoencoder

Τέλος, εφαρμόσαμε ανίχνευση ανωμαλιών στη χρονοσειρά M19, χρησιμοποιώντας RNN-Autoencoder, δηλαδή αυτοκωδικοποιητή με επανλαμβανόμενα νευρωνικά δίκτυα, έχοντας προεπεξεργαστεί τη χρονοσειρά.

Ένας RNN-Autoencoder, συντομογραφία του Recurrent Neural Network Autoencoder, είναι ένας τύπος αρχιτεκτονικής νευρωνικών δικτύων που χρησιμοποιείται για μάθηση χωρίς επίβλεψη (unsupervised learning) και μείωση διαστάσεων (dimensionality reduction) διαδοχικών δεδομένων. Συνδυάζει τις δυνατότητες επαναλαμβανόμενων νευρωνικών δικτύων (RNN) και αυτοκωδικοποιητών. Ακολουθεί μια επισκόπηση βήμα προς βήμα του τρόπου λειτουργίας ενός RNN-Autoencoder: (Tang, et al., 2019)

1. Προετοιμασία Δεδομένων (Data Preparation):

    - Διαδοχικά δεδομένα (Sequential Data): Προετοιμάζουμε τα διαδοχικά δεδομένα, όπως χρονοσειρές ή δεδομένα κειμένου, που θα χρησιμοποιηθούν ως είσοδος στον αυτόματο κωδικοποιητή RNN.
    - Κανονικοποίηση δεδομένων: Κανονικοποιούμε τα δεδομένα για να διασφαλίσουμε ότι όλα τα χαρακτηριστικά βρίσκονται σε παρόμοια κλίμακα, κάτι που βοηθά στην εκπαίδευση του δικτύου.

2. Κωδικοποιητής (Encoder):

    - Κωδικοποίηση εισόδου: Τα διαδοχικά δεδομένα τροφοδοτούνται στο επίπεδο κωδικοποιητή RNN, το οποίο επεξεργάζεται τα δεδομένα βήμα προς βήμα.
    - Αναπαράσταση κρυφής κατάστασης: Το επίπεδο RNN εξάγει μια αναπαράσταση κρυφής κατάστασης σε κάθε χρονικό βήμα, η οποία καταγράφει τις χρονικές εξαρτήσεις και συνοψίζει τις πληροφορίες στην ακολουθία εισόδου.

3. Στρώμα συμφόρησης (Bottleneck Layer):

    - Μείωση διαστάσεων: Η αναπαράσταση κρυφής κατάστασης από τον κωδικοποιητή περνά μέσα από ένα στρώμα συμφόρησης, το οποίο μειώνει τη διάσταση των δεδομένων.
    - Συμπίεση: Το στρώμα συμφόρησης συμπιέζει τις πληροφορίες διατηρώντας παράλληλα σημαντικά χαρακτηριστικά της ακολουθίας εισόδου.

4. Αποκρυπτογράφος (Decoder):

    - Ανακατασκευή: Η συμπιεσμένη αναπαράσταση από το bottleneck layer περνά μέσα από το στρώμα αποκωδικοποιητή RNN, το οποίο στοχεύει στην ανακατασκευή της αρχικής ακολουθίας.
    - Παραγωγή εξόδου: Σε κάθε χρονικό βήμα, ο αποκωδικοποιητής παράγει μια έξοδο που αντιπροσωπεύει την ανακατασκευασμένη τιμή για αυτό το χρονικό βήμα.

5. Υπολογισμός απώλειας (Loss Calculation):

    - Σφάλμα ανακατασκευής: Συγκρίνεται η έξοδος του αποκωδικοποιητή με την αρχική ακολουθία εισόδου για να υπολογιστεί το σφάλμα ανακατασκευής.

- Λειτουργία απώλειας: Χρησιμοποιείται μια κατάλληλη συνάρτηση απώλειας, όπως το μέσο τετραγωνικό σφάλμα (MSE), για να μετρηθεί η διαφορά μεταξύ της αρχικής εισόδου και της ανακατασκευασμένης εξόδου.

6. Εκπαίδευση (Training):

   - Backpropagation: Εκτελείται backpropagation για να ενημερωθούν τα βάρη και οι προκαταλήψεις του RNN-Autoencoder, ελαχιστοποιώντας το σφάλμα ανακατασκευής.
   - Επαναλήψεις Εκπαίδευσης: Επανάληψη της διαδικασίας εκπαίδευσης για πολλαπλές επαναλήψεις ή εποχές μέχρι το δίκτυο να συγκλίνει ή να επιτύχει ένα ικανοποιητικό επίπεδο ανακατασκευής.

7. Ανίχνευση ανωμαλιών:

   - Κατώφλι σφάλματος ανακατασκευής (Reconstruction Error Threshold): Μόλις εκπαιδευτεί ο αυτόματος κωδικοποιητής RNN, μπορούν να εντοπιστούν ανωμαλίες ή ακραίες τιμές συγκρίνοντας το σφάλμα ανακατασκευής αόρατων δεδομένων με ένα προκαθορισμένο όριο.
   - Αναγνώριση ανωμαλίας: Τα σημεία δεδομένων με σφάλματα ανακατασκευής που υπερβαίνουν το όριο επισημαίνονται ως ανωμαλίες ή ύποπτα σημεία δεδομένων.

Εκπαιδεύοντας τον RNN-Autoencoder στην ακολουθία εισόδου και βελτιστοποιώντας τη διαδικασία ανακατασκευής, το δίκτυο μαθαίνει να καταγράφει τα βασικά μοτίβα και τις χρονικές εξαρτήσεις στα δεδομένα. Στη συνέχεια, μπορεί να ανιχνεύσει ανωμαλίες ή ανωμαλίες σε νέες, αόρατες ακολουθίες που βασίζονται σε αποκλίσεις από τα μαθησιακά μοτίβα κατά τη φάση της ανασυγκρότησης.

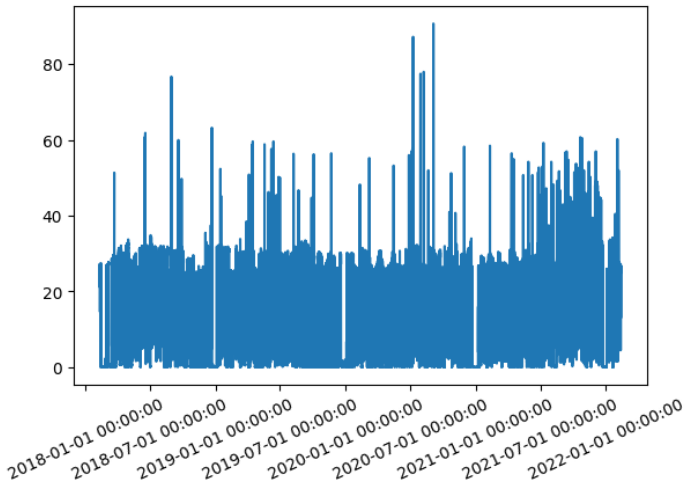
Για τα δεδομένα της χρονοσειράς M19, ακολουθήσαμε την προαναφερθείσα διαδικασία.

Αρχικά χωρίσαμε τα δεδομένα μας σε train data (80%) και test data (20%).

```
from sklearn.model_selection import train_test_split

# Split data into train and test sets
train_size = int(len(df) * 0.8)  # 80% for training, 20% for testing
train_data = df.iloc[:train_size]
test_data = df.iloc[train_size:]
```

Train Data:

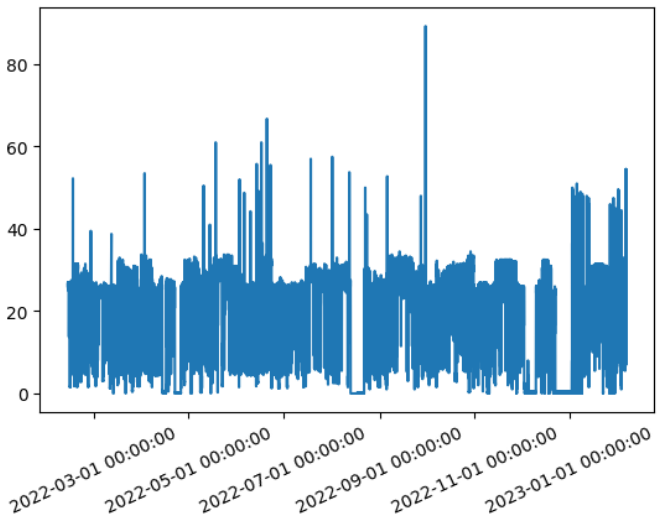
```
plot_values(train_data)
```



Εικόνα 3-30 Train data

Test Data:

```
plot_values(test_data)
```



Εικόνα 3-31 Test data

Προετοιμάσαμε τα δεδομένα εκπαίδευσης. Συγκεκριμένα πήραμε τις τιμές από τα δεδομένα εκπαίδευσης και τα κανονικοποιήσαμε.

```
def get_value_from_df(df):
    return df.value.to_list()


def normalize(values):
    mean = np.mean(values)
    values -= mean
    std = np.std(values)
    values /= std
    return values, mean, std


# Get the `value` column from the training dataframe.
training_value = get_value_from_df(train_data)

# Normalize `value` and save the mean and std we get,
# for normalizing test data.
training_value, training_mean, training_std = normalize(training_value)
len(training_value)
```

```
137456
```

Έχουμε όπως φαίνεται 137456 σημεία δεδομένων στο test set μας.

Ύστερα δημιουργήσαμε τις ακολουθίες οι οποίες θα αποτελέσουν την είσοδο του κωδικοποιητή:

```
TIME_STEPS = 288


def create_sequences(values, time_steps=TIME_STEPS):
    output = []
    for i in range(len(values) - time_steps):
        output.append(values[i : (i + time_steps)])
    # Convert 2D sequences into 3D as we will be feeding this into
    # a convolutional layer.
    return np.expand_dims(output, axis=2)


x_train = create_sequences(training_value)
print("Training input shape: ", x_train.shape)
```

```
Training input shape:  (137168, 288, 1)
```

Στη συνέχεια ορίσαμε το μοντέλο του αυτοκωδικοποιητή συνελικτικής ανακατασκευής (convolutional reconstruction autoencoder model) . Το μοντέλο παίρνει είσοδο μεγέθους *(batch_size, sequence_length, num_features)* και επιστρέφει έξοδο ίδιου μεγέθους. Στη δική μας περίπτωση *sequence_length = 288* και *num_features = 1*.

```
n_steps = x_train.shape[1]
n_features = x_train.shape[2]

keras.backend.clear_session()
model = keras.Sequential(
    [
        layers.Input(shape=(n_steps, n_features)),
        layers.Conv1D(filters=32, kernel_size=15, padding='same', data_format='channels_last',
            dilation_rate=1, activation="linear"),
        layers.LSTM(
            units=25, activation="tanh", name="lstm_1", return_sequences=False
        ),
        layers.RepeatVector(n_steps),
        layers.LSTM(
            units=25, activation="tanh", name="lstm_2", return_sequences=True
        ),
        layers.Conv1D(filters=32, kernel_size=15, padding='same', data_format='channels_last',
            dilation_rate=1, activation="linear"),
        layers.TimeDistributed(layers.Dense(1, activation='linear'))
    ]
)
model.compile(optimizer=keras.optimizers.Adam(learning_rate=0.001), loss="mse")
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv1d (Conv1D)             (None, 288, 32)           512

 lstm_1 (LSTM)               (None, 25)                5800

 repeat_vector (RepeatVecto  (None, 288, 25)           0
 r)

 lstm_2 (LSTM)               (None, 288, 25)           5100

 conv1d_1 (Conv1D)           (None, 288, 32)           12032

 time_distributed (TimeDist  (None, 288, 1)            33
 ributed)

=================================================================
Total params: 23477 (91.71 KB)
Trainable params: 23477 (91.71 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Εκπαιδεύσαμε το μοντέλο με είσοδο το x_train και target επίσης το x_train, καθώς έχουμε ένα μοντέλο ανακατασκευής.

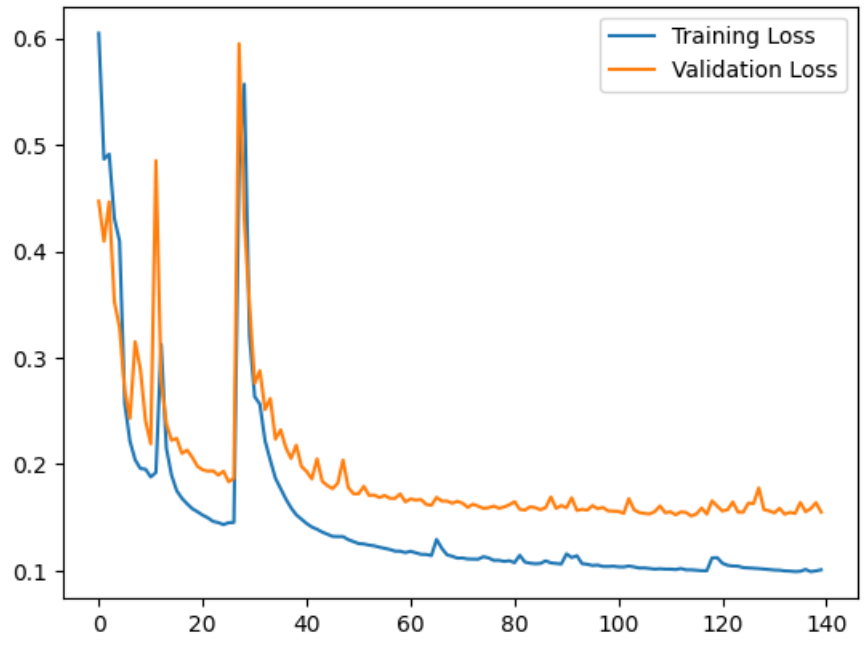
```
history = model.fit(
    x_train,
    x_train,
    epochs=200,
    batch_size=128,
    validation_split=0.1,
    callbacks=[
        keras.callbacks.EarlyStopping(monitor="val_loss", patience=25, mode="min", restore_best_weights=True)
    ],
)
```

```
Epoch 1/200
965/965 [==============================] - 41s 22ms/step - loss: 0.6048 - val_loss: 0.4470
Epoch 2/200
965/965 [==============================] - 21s 22ms/step - loss: 0.4865 - val_loss: 0.4094
Epoch 3/200
965/965 [==============================] - 20s 21ms/step - loss: 0.4912 - val_loss: 0.4464
Epoch 4/200
965/965 [==============================] - 24s 25ms/step - loss: 0.4308 - val_loss: 0.3528
Epoch 5/200
965/965 [==============================] - 22s 23ms/step - loss: 0.4096 - val_loss: 0.3284
```

Για να δούμε πώς πήγε η εκπαίδευση απεικονίσαμε στο ίδιο διάγραμμα τα training loss και validation loss:

```
plt.plot(history.history["loss"], label="Training Loss")
plt.plot(history.history["val_loss"], label="Validation Loss")
plt.legend()
```

```
<matplotlib.legend.Legend at 0x7f06d44c2980>
```



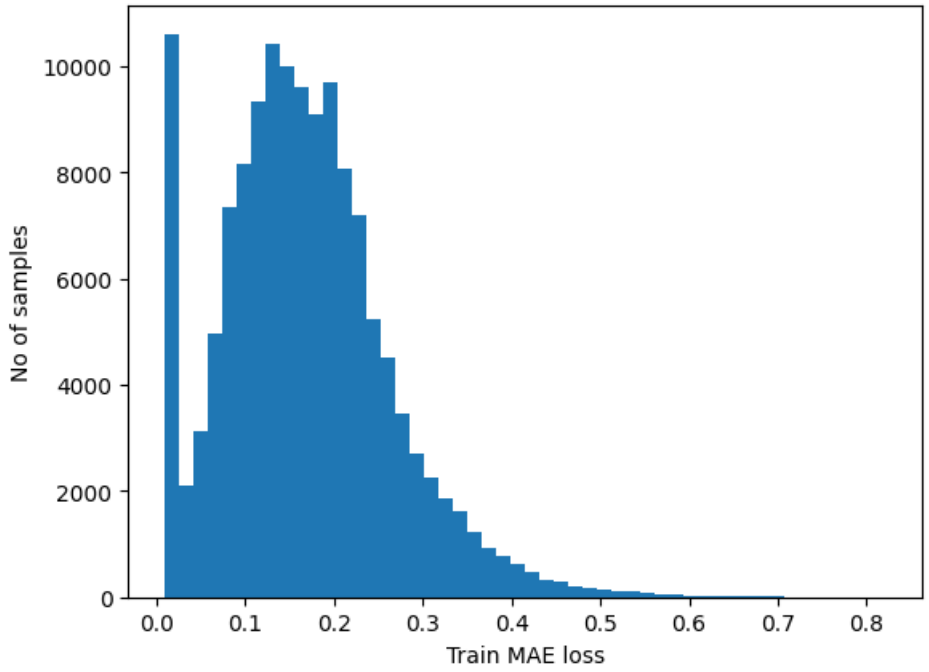
Εικόνα 3-32 Training loss & validation loss

Προχωρήσαμε στην ανίχνευση ανωμαλιών προσδιορίζοντας πόσο καλά το μοντέλο μας μπορεί να ανακατασκευάσει τα δεδομένα εισόδου. Βρήκαμε την απώλεια MAE (mean absolute error) σε δείγματα εκπαίδευσης και τη μέγιστη τιμή απώλειας MAE. Αυτό είναι το χειρότερο αποτέλεσμα του μοντέλου μας προσπαθώντας να ανακατασκευάσει ένα δείγμα. Το καταστήσαμε όριο για τον εντοπισμό ανωμαλιών.

Εάν η απώλεια ανακατασκευής για ένα δείγμα είναι μεγαλύτερη από αυτήν την τιμή κατωφλίου, τότε μπορούμε να συμπεράνουμε ότι το μοντέλο βλέπει ένα μοτίβο με το οποίο δεν είναι εξοικειωμένο. Χαρακτηρίζουμε αυτό το δείγμα ως ανωμαλία.

```
# Get train MAE loss.
x_train_pred = model.predict(x_train)
train_mae_loss = np.mean(np.abs(x_train_pred - x_train), axis=1)

print(train_mae_loss)

plt.hist(train_mae_loss, bins=50)
plt.xlabel("Train MAE loss")
plt.ylabel("No of samples")
plt.show()

# Get reconstruction loss threshold.
# threshold = np.max(train_mae_loss)
threshold = np.percentile(train_mae_loss, 95)
print("Reconstruction error threshold: ", threshold)
```

```
4287/4287 [==============================] - 37s 8ms/step
[[0.11034596]
 [0.11562245]
 [0.11679995]
 ...
 [0.1648115 ]
 [0.15639171]
 [0.15802742]]
```

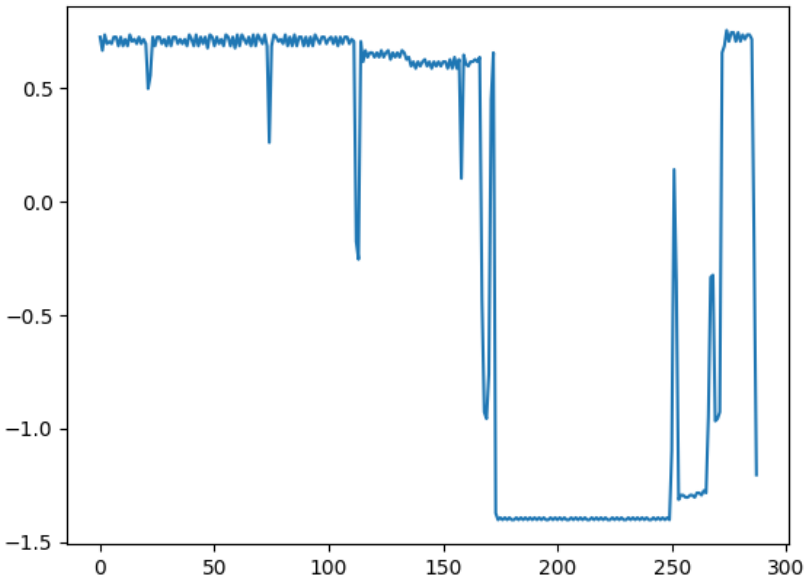



```
Reconstruction error threshold:  0.3388265901394023
```

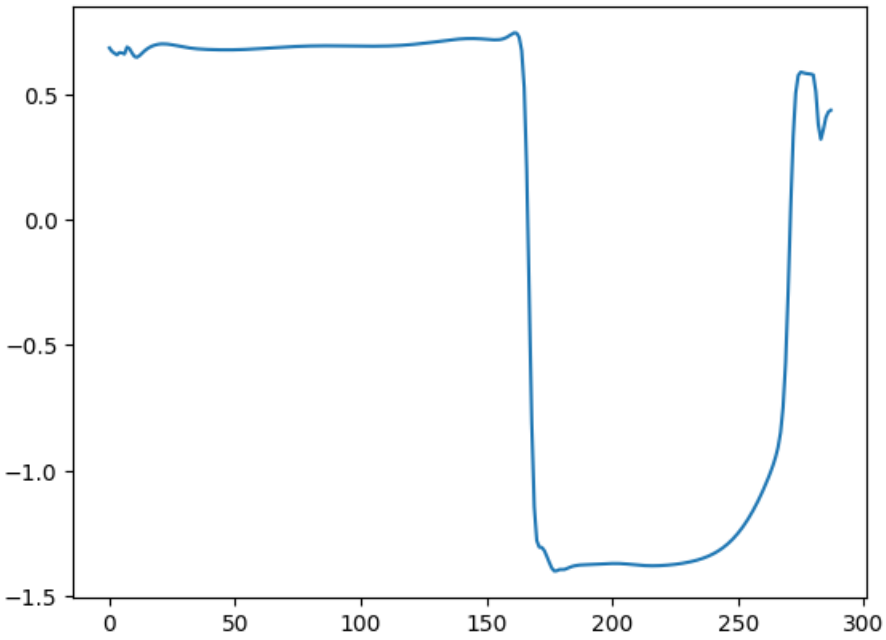
Εικόνα 3-33 Train MAE loss

Έχει ενδιαφέρον να δούμε πώς ανακατασκεύσε το μοντέλο μας το πρώτο δείγμα.

```
# Checking how the first sequence is learnt
plt.plot(x_train[0])
plt.show()
plt.plot(x_train_pred[0])
plt.show()
```



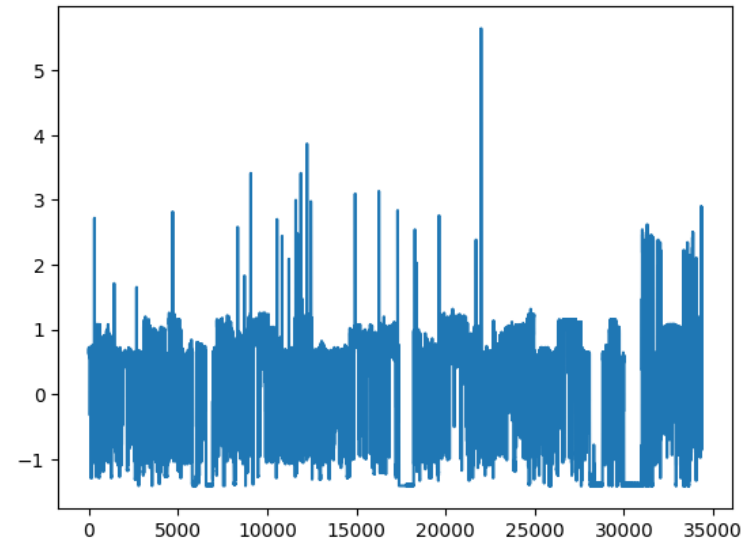
Εικόνα 3-34 Πρώτο δείγμα x_train



Εικόνα 3-35 Ανακατασκεύη πρώτου δείγματος

Προετοιμάσαμε τα δεδομένα ελέγχου:

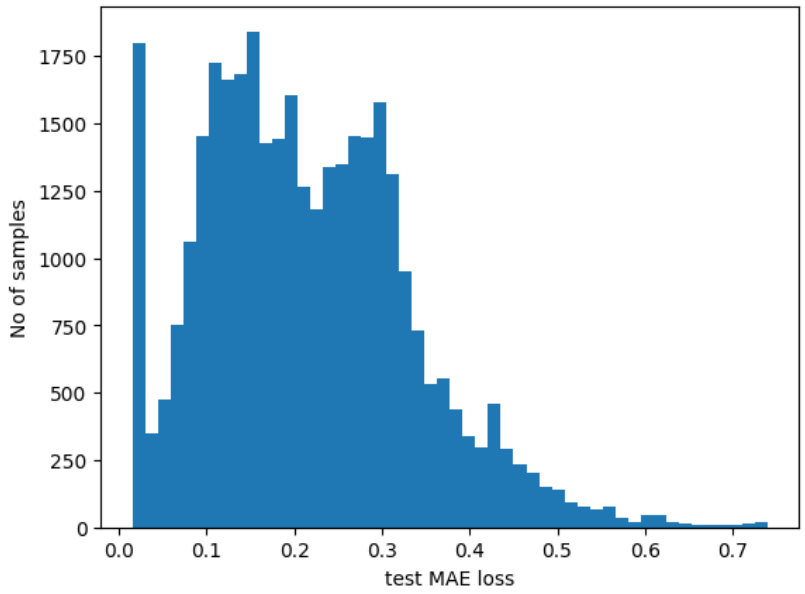
```
def normalize_test(values, mean, std):
    values -= mean
    values /= std
    return values


test_value = get_value_from_df(test_data)
test_value = normalize_test(test_value, training_mean, training_std)
plt.plot(test_value.tolist())
plt.show()

# Create sequences from test values.
x_test = create_sequences(test_value)
print("Test input shape: ", x_test.shape)

# Get test MAE loss.
x_test_pred = model.predict(x_test)
test_mae_loss = np.mean(np.abs(x_test_pred - x_test), axis=1)
test_mae_loss = test_mae_loss.reshape((-1))

plt.hist(test_mae_loss, bins=50)
plt.xlabel("test MAE loss")
plt.ylabel("No of samples")
plt.show()

# Detect all the samples which are anomalies.
anomalies = (test_mae_loss > threshold).tolist()
print("Number of anomaly samples: ", np.sum(anomalies))
print("Indices of anomaly samples: ", np.where(anomalies))
```




```
Test input shape:  (34077, 288, 1)
1065/1065 [==============================] - 10s 10ms/step
```

Εικόνα 3-36 Test Value

```
Test input shape:  (34077, 288, 1)
1065/1065 [==============================] - 10s 10ms/step
```




```
Number of anomaly samples:  4721
Indices of anomaly samples:  (array([  181,   182,   183, ..., 33840, 33841, 33842]),)
```

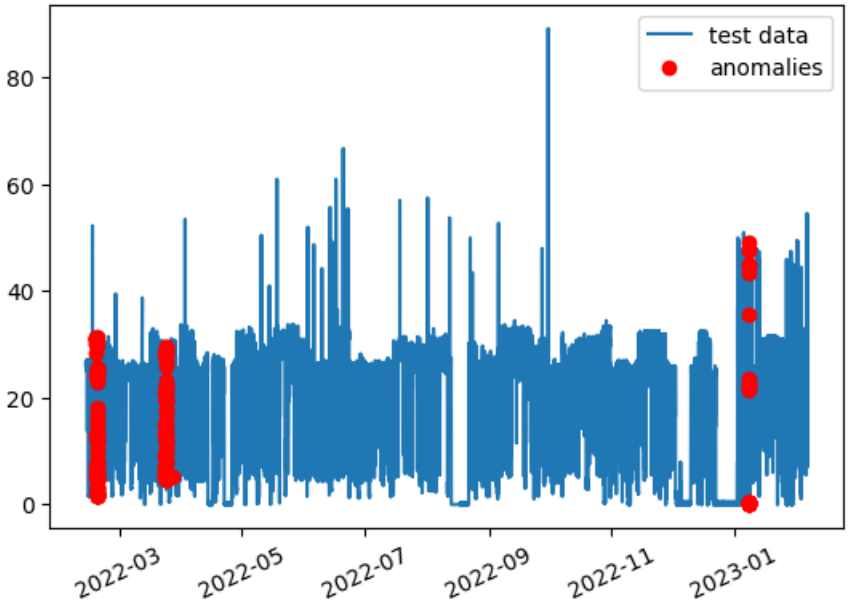
Εικόνα 3-37 Test MAE loss

Γνωρίζοντας τα δείγματα των δεδομένων που είναι ανωμαλίες, βρήκαμε τις αντίστοιχες χρονικές στιγμές από το αρχικό test set. Ένα σημείο είναι μη φυσιολογικό αν τα δείγματα $(i - timesteps + 1)$ μέχρι i είναι μη φυσιολογικά.

```
# data i is an anomaly if samples [(i - timesteps + 1) to (i)] are anomalies
anomalous_data_indices = []
for data_idx in range(TIME_STEPS - 1, len(test_value) - TIME_STEPS + 1):
    time_series = range(data_idx - TIME_STEPS + 1, data_idx)
    if all([anomalies[j] for j in time_series]):
        anomalous_data_indices.append(data_idx)
```

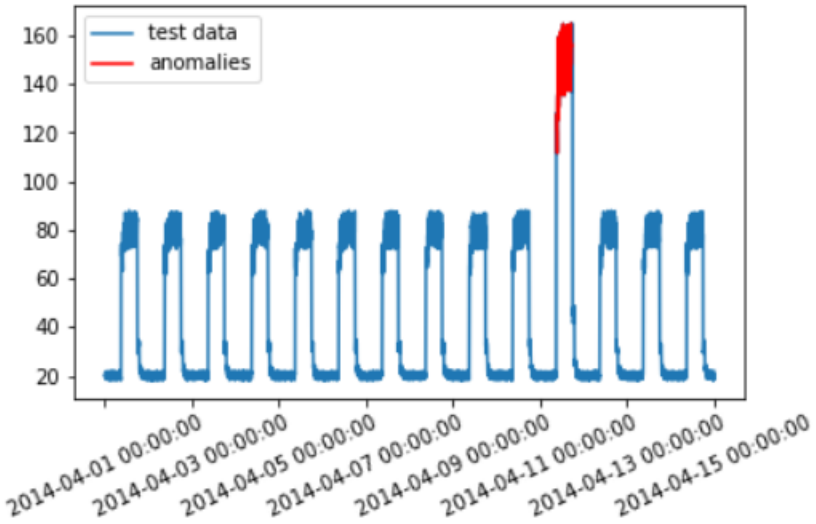
Τέλος απεικονίσαμε πανω στο διάγραμμα τα ανώμαλα σημεία.

```
df_subset = test_data.iloc[anomalous_data_indices, :]
plt.subplots_adjust(bottom=0.2)
plt.xticks(rotation=25)
ax = plt.gca()
# xfmt = md.DateFormatter("%Y-%m-%d %H:%M:%S")
# ax.xaxis.set_major_formatter(xfmt)

dates = test_data["timestamp"].to_list()
values = test_data["value"].to_list()
plt.plot(dates, values, label="test data")

dates = df_subset["timestamp"].to_list()
values = df_subset["value"].to_list()
plt.plot(dates, values, 'o', label="anomalies", color="r")

plt.legend()
plt.show()
```



Εικόνα 3-38 Χρονοσειρά M19 με επισημασμένες τις ανωμαλίες με RNN – Autoencoder

Παρατηρούμε ότι το συγκεκριμένο μοντέλο δεν μας δίνει καλά αποτελέσματα και χρειάζεται περισσότερη μελέτη και δοκιμές για να εντοπίσουμε γιατί δεν εκπαιδεύεται καλά στα δεδομένα μας. Σε αλλό dataset βέβαια μας δίνει καλύτερα αποτελέσματα και λειτουργεί πολύ πιο αποδοτικά, καθώς έχει καταφέρει να εντοπίσει μοτίβα στα δεδομένα, όπως φανερώνει και το διάγραμμα.



Εικόνα 3-39 Αποδοτική ανίχνευση ανωμαλιών με RNN – Autoencoder σε διαφορετικά δεδομένα

3.4 Soft / Hard voting

Soft voting και hard voting είναι δύο μέθοδοι που χρησιμοποιούνται στη μηχανική μάθηση, ιδίως στον τομέα της ταξινόμησης, για να συνδυάσουν τις προβλέψεις πολλών μεμονωμένων μοντέλων σε μια ενιαία τελική πρόβλεψη.

Hard voting:

Στο hard voting κάθε μεμονωμένο μοντέλο του συνόλου κάνει μια πρόβλεψη και η τελική πρόβλεψη καθορίζεται με ψηφοφορία πλειοψηφίας. Η ετικέτα κλάσης που λαμβάνει τις περισσότερες ψήφους από τα μεμονωμένα μοντέλα επιλέγεται ως τελική πρόβλεψη. Σε αυτή την προσέγγιση, κάθε μοντέλο έχει ίση βαρύτητα στη διαδικασία λήψης αποφάσεων.

Για παράδειγμα, ας υποθέσουμε ότι έχουμε ένα σύνολο τριών μοντέλων: Το μοντέλο Α προβλέπει την κλάση 1, το μοντέλο Β προβλέπει την κλάση 2 και το μοντέλο Γ προβλέπει την κλάση 1. Η τελική πρόβλεψη θα είναι η κλάση 1, καθώς έλαβε την πλειοψηφία των ψήφων από τα μοντέλα.

Το hard voting είναι αποτελεσματικό όταν τα μεμονωμένα μοντέλα στο σύνολο είναι διαφορετικά και κάνουν ανεξάρτητα σφάλματα. Λειτουργεί καλά όταν τα μοντέλα έχουν παρόμοια επίπεδα απόδοσης και δεν είναι επιρρεπή σε υπερβολική προσαρμογή (overfitting).

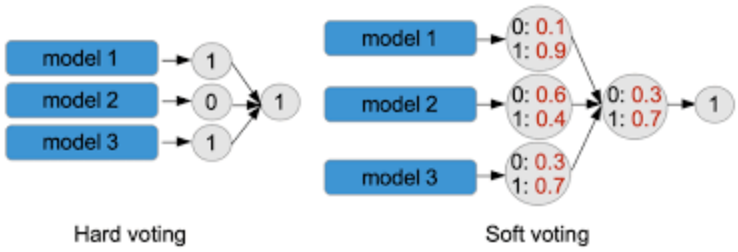
Soft voting:

Στο soft voting κάθε μεμονωμένο μοντέλο στο σύνολο προβλέπει τις πιθανότητες ή τα confidence scores για κάθε κλάση αντί για τις πραγματικές ετικέτες κλάσεων. Στη συνέχεια, η τελική πρόβλεψη υπολογίζεται με τη μέση τιμή ή τη λήψη του σταθμισμένου μέσου όρου αυτών των πιθανοτήτων σε όλα τα μοντέλα. Η κλάση με την υψηλότερη μέση πιθανότητα επιλέγεται ως τελική πρόβλεψη.

Για παράδειγμα, ας υποθέσουμε ότι έχουμε ένα σύνολο τριών μοντέλων: Το μοντέλο Α προβλέπει πιθανότητες [0.6, 0.4] για την κλάση 1 και την κλάση 2, το μοντέλο Β προβλέπει πιθανότητες [0.3, 0.7] και το μοντέλο Γ προβλέπει πιθανότητες [0.8, 0.2]. Με soft voting, η τελική πρόβλεψη θα είναι η κλάση 1 επειδή οι μέσες πιθανότητες για την κλάση 1 είναι υψηλότερες από εκείνες για την κλάση 2.

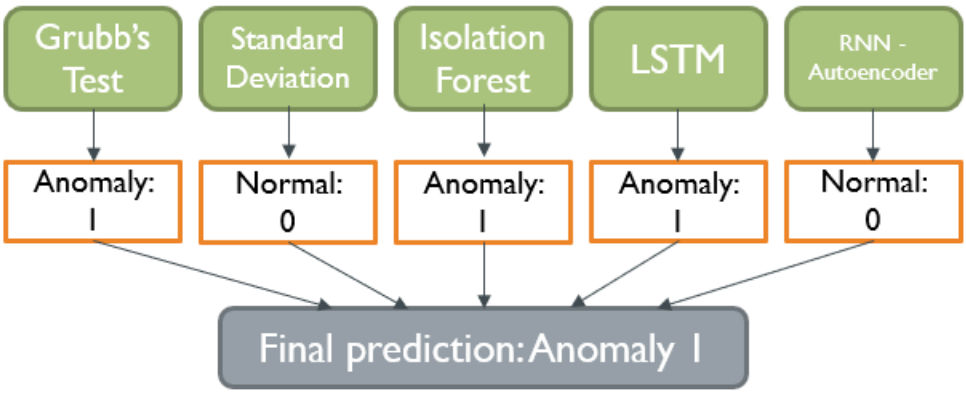
Το soft voting λαμβάνει υπόψη τα επίπεδα εμπιστοσύνης των επιμέρους μοντέλων στις προβλέψεις τους. Είναι χρήσιμο όταν τα μοντέλα παρέχουν εκτιμήσεις πιθανοτήτων και μπορούν να μεταφέρουν την αβεβαιότητα των προβλέψεών τους. Λαμβάνοντας υπόψη τις πιθανότητες, μπορεί να παρέχει πιο ακριβείς προβλέψεις.

Τόσο το hard voting όσο και το soft voting εφαρμόζονται σε σενάρια μάθησης, όπου συνδυάζονται πολλαπλά μοντέλα για να βελτιωθεί η συνολική απόδοση και η γενίκευση. Η επιλογή μεταξύ των δύο εξαρτάται από τη φύση του προβλήματος, τα χαρακτηριστικά των μοντέλων και τη διαθεσιμότητα των εκτιμήσεων πιθανοτήτων. Έτσι λοιπόν και στη δική μας περίπτωση η είναι χρήσιμη η εφαρμογή είτε soft είτε hard voting, μιας και χρησιμοποιούμε για τον εντοπισμό ακραίων τιμών και ανωμαλιών διάφορες μεθόδους και τεχνικές, από στατιστικές μεθόδους μέχρι αλγορίθμους βασισμένους στο deep learning.

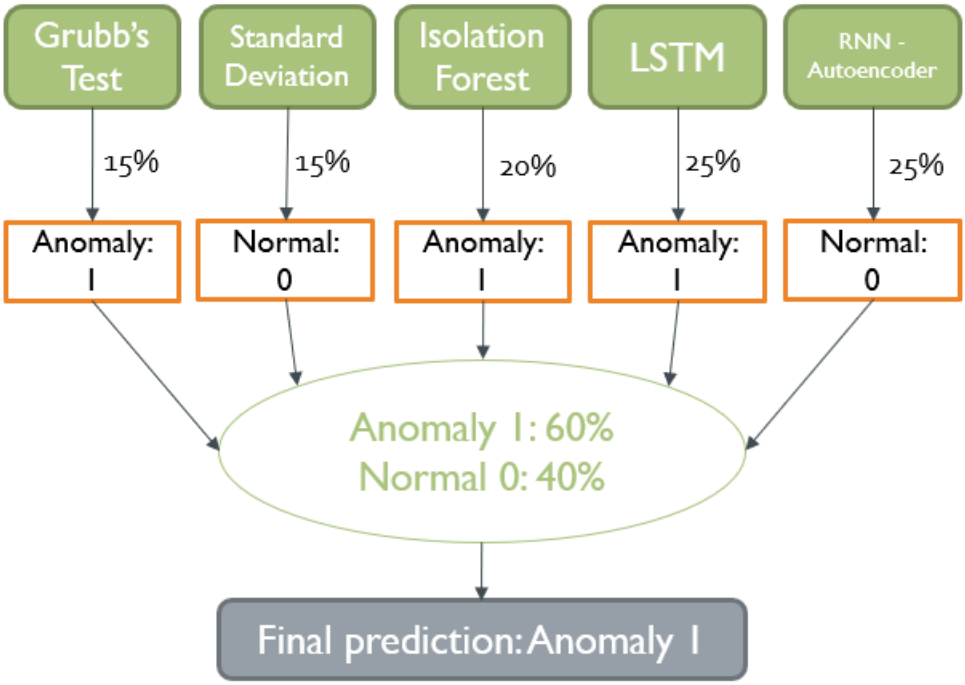


Εικόνα 3-39 Οπτική επεξήγηση hard/soft voting

Ιδανικά θα θέλαμε να κρατήσουμε έναν περιττό αριθμό μοντέλων, ας πούμε πέντε και να εξάγουμε συμπέρασμα για το αν μία τιμή από τα σημεία δεδομένων είναι ανώμαλη μέσω του συνδυασμού τους. Σε περίπτωση εφαρμογής hard voting, αυτές οι μέθοδοι έχουν ισάξια βάρη 0.2 (20%) και λαμβάνουμε υπόψη την ψήφο της πλειοψηφίας μεταξύ των μοντέλων. Μια προτεινόμενη διαδικασία για τη δική μας περίπτωση είναι η εξής. Αν τέσσερα ή περισσότερα μοντέλα επισημάνουν ένα σημείο δεδομένων ώς ανώμαλο, τότε το κάνουμε κατευθείαν label ως ανωμαλία. Αν τρία μοντέλα επισημάνουν ένα σημείο δεδομένων ώς ανώμαλο, τότε αυτό είναι μια πιθανή ανωμαλία και το κάνουμε label ως warning. Αν λιγότερα απο τρία μοντέλα επισημάνουν ένα σημείο ως ανώμαλο, τότε δεν μας επηρεάζει και το αφήνουμε ως έχει.

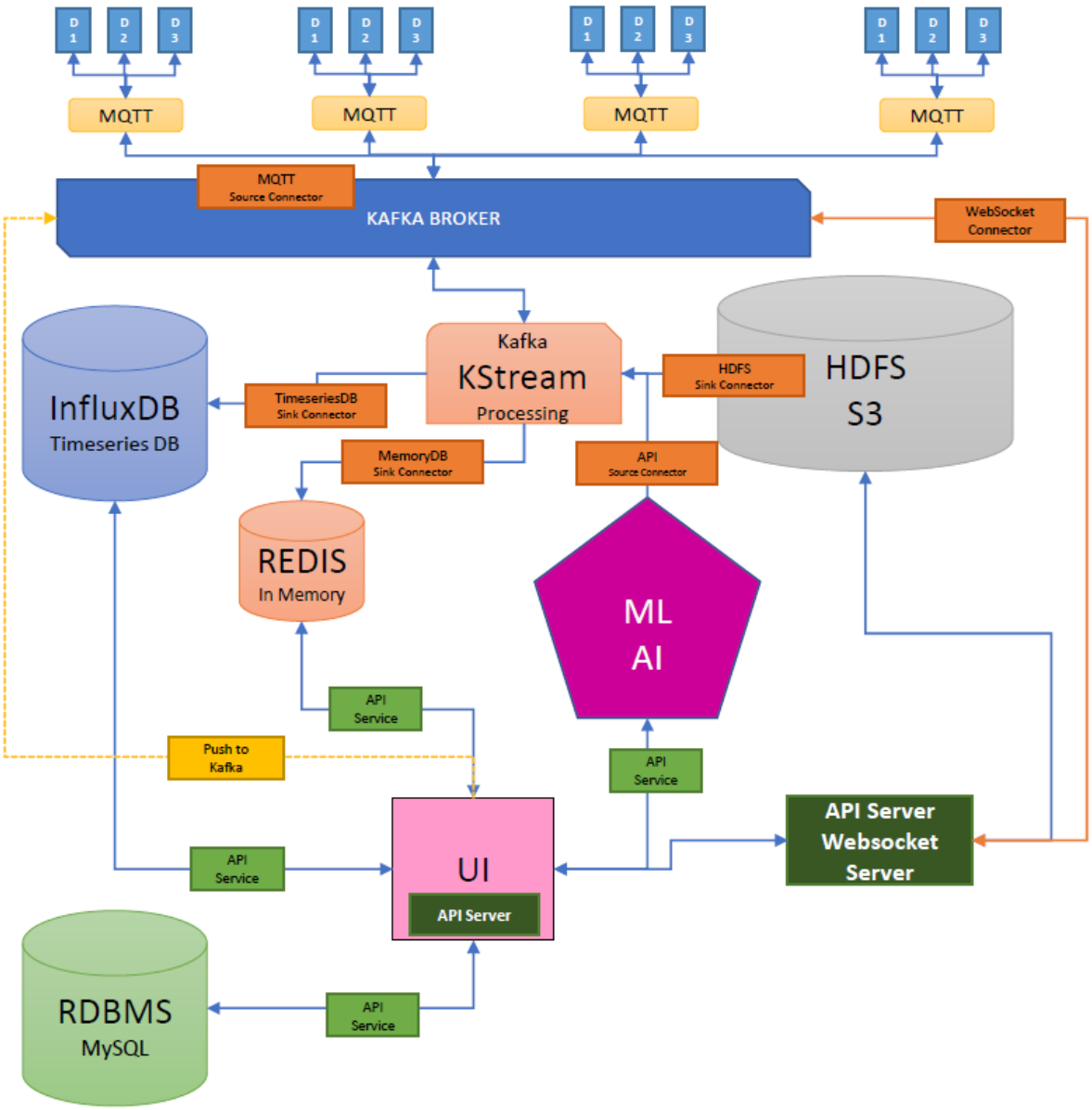


Σε περίπτωση εφαρμογής soft voting όπως έχουμε αναφέρει ήδη κάθε μοντέλο μπορεί να επηρεάσει διαφορετικά το τελικό αποτέλεσμα λόγω των διαφορετκών βαρών των μοντέλων. Νωρίτερα στην εργασία πραγματοποιήσαμε ανίχνευση ανωμαλιών με στατιστικές μεθόδους (standard deviation, IQR, z-score, Grubb's test κοκ), με κλασικές τεχνικές (isolation forest) καθώς και με πιο εξειδικευμένες τεχνικές (LSTM, RNN-Autoencoder). Προφανώς αυτές οι τεχνικές δεν έχουν τις ίδιες επιδόσεις και γι' αυτό παίζει σημαντικό ρόλο η κατάλληλη επιλογή βαρών. Αν υποθέσουμε ότι κρατάμε τις δύο αποδοτικότερες στατιστικές μεθόδους (Model A, Model B), το isolation forest με περισσότερα από ένα features και μετά από fine tuning παραμέτρων (Model C), το LSTM (Model D) και το RNN-Autoencoder (Model E), τότε μία πρόταση για ενδεικτικά βάρη είναι: Model A → 0.15, Model B → 0.15, Model C → 0.2, Model D → 0.25, Model E → 0.25. Ανώμαλο θα θεωρήσουμε ένα σημείο που θα επισημανθεί με πιθανότητα μεγαλύτερη ή ίση του 0.65 (65%), warning θα θεωρήσουμε ένα σημείο που θα επισημανθεί με πιθανότητα μικρότερη του 0.65 (65%) και μεγαλύτερη ή ίση του 0.5 (50%), ενώ οτιδήποτε χαμηλότερο του του 0.5 (50%) δεν μας επηρεάζει.



3.5 Ενδεικτική αρχιτεκτονική πλαισίου για την υλοποίηση των προαναφερθέντων μοντέλων

Σχεδιάσμε μια ενδεικτική αρχιτεκτονική πλαισίου η οποία υλοποιεί τα προαναφερθέντα.



Εικόνα 3-39 Προτεινόμενη IoT Αρχιτεκτονική

Στην προτεινόμενη αρχιτεκτονική, οι συσκευές επικοινωνούν στέλνοντας και λαμβάνοντας πληροφορίες μέσω του MQTT, ενός ελαφρού πρωτοκόλλου ανταλλαγής μηνυμάτων που χρησιμοποιείται συνήθως σε συστήματα IoT. Το Kafka ενεργεί ως κεντρικός μεσίτης (broker) μηνυμάτων, λαμβάνοντας μηνύματα από το MQTT και διευκολύνοντας την περαιτέρω επεξεργασία.

Οι δαίμονες (deamons) λαμβάνουν μηνύματα MQTT και τα διανέμουν σε διάφορα στοιχεία (components) για επεξεργασία. Ένα από αυτά τα συστατικά είναι το Kafka Streams, το οποίο χρησιμεύει ως επίπεδο επεξεργασίας δεδομένων πραγματικού χρόνου. Στη θέση του Kafka Streams μπορεί να μπει Apache Flink ή κάποιο άλλο framework κατανεμημένης επεξεργασίας. Εφαρμόζει μετασχηματισμούς και υπολογισμούς στα δεδομένα καθώς αυτά ρέουν μέσω του συστήματος.

Το Kafka Streams γράφει τα επεξεργασμένα δεδομένα σε διάφορα συστήματα αποθήκευσης χρησιμοποιώντας συνδετήρες Kafka (Kafka connectors). Ένας προορισμός είναι μια time series data base όπως το InfluxDB, το οποίο χρησιμεύει ως hot storage για τη διατήρηση των μηνυμάτων για διάρκεια ενός έτους. Ένας άλλος προορισμός είναι μια in-memory data base όπως η RedisDB, στην οποία αποθηκεύονται μόνο οι τελευταίες τιμές για γρήγορη προσβασιμότητα μέσω API.

Επιπλέον, το Kafka Streams γράφει όλα τα δεδομένα, συμπεριλαμβανομένων τόσο των ακατέργαστων όσο και των επεξεργασμένων, σε ένα κατανεμημένο σύστημα διαμοιρασμού αρχείων (cold storage), όπως το S3 ή το Hadoop DFS. Τα δεδομένα αυτά παραμένουν διαθέσιμα και για εργασίες μηχανικής μάθησης μέσω μιας διεπαφής χρήστη ML (UI).

Το UI αλληλεπιδρά με τις βάσεις δεδομένων διαβάζοντας και γράφοντας δεδομένα μέσω υπηρεσιών API. Συγκεκριμένα, διαβάζει δεδομένα από το HDFS μέσω ενός διακομιστή API και επικοινωνεί με τα μοντέλα ML ενεργοποιώντας ενέργειες μέσω μιας άλλης υπηρεσίας API. Ένας σύνδεσμος WebSocket χρησιμοποιείται για την ανάκτηση ζωντανών δεδομένων από τον Kafka broker και την παράδοσή τους στο UI σε πραγματικό χρόνο μέσω ενός διακομιστή WebSocket (server-client communication).

Το UI μπορεί να δρομολογήσει ενέργειες του χρήστη και να στείλει εργασίες στο στοιχείο ML. Στη συνέχεια, το ML component εκπαιδεύει μοντέλα και τα αποθηκεύει στο HDFS. Η υπηρεσία ML API, η οποία χρησιμοποιεί την υπηρεσία TensorFlow Service συσκευασμένη σε δοχεία Docker, μπορεί να εξυπηρετεί αυτά τα μοντέλα για εκτέλεση σε πραγματικό χρόνο ή προγραμματισμένη εκτέλεση.

Επιπλέον υπάρχει μια RDBMS βάση για την αποθήκευση των business δεδομένων της εφαρμογής (logins κοκ).

Όλες οι υπηρεσίες, εκτός από τις βάσεις δεδομένων και τις stateful εφαρμογές, αναγκαστικά χρησιμοποιούν το Docker. Στις περιπτώσεις όπου οι stateful εφαρμογές είναι containerized, το σύστημα HDFS χρησιμοποιείται για αποθήκευση, όπως στην υπηρεσία ML. Για την επίτευξη επεκτασιμότητας και υψηλής απόδοσης, όλες οι υπηρεσίες API τοποθετούνται πίσω από έναν εξισορροπιστή φορτίου (load balancer).

Όλα τα συστήματα και τα εργαλεία που χρησιμοπούνται στην προτεινόμενη αρχιτεκτονική υποστηρίζουν γρήγορες ταχύτητες, είναι κατάλληλα για χρήση σε μεγάλους όγκους δεδομένων και χαρακτηρίζονται από την εύκολη επεκτασιμότητά τους. Επιπλέον με τη χρήση αυτών των συστημάτων επιτυγχάνεται μια αμφίδρομη επικοινωνία χρηστών και συσκευών. Έτσι, αυτή η αρχιτεκτονική παρέχει ένα συνεκτικό και κλιμακούμενο πλαίσιο για την επεξεργασία δεδομένων IoT, συνδυάζοντας τα MQTT, Kafka, Kafka Streams, διάφορα συστήματα αποθήκευσης, βάσεις δεδομένων, στοιχεία ML και διεπαφές χρήστη, όλα ενορχηστρωμένα με χρήση containers και υποστηριζόμενα από υπηρεσίες API.

Συγκεντρωτικά βλέπουμε στον παρακάτω πίνακα:

	Scalable	High Speed	Big Data
MQTT	X	X	
KAFKA	X	X	X
Kstreams	X	X	X
InfluxDB	X	X	X
RedisDB	X	X	
HDFS	X		X

ML	X		X
Websocket	X	X	
API	X		X
RDBMS		X	

3.6 Roadmap

Σε συνέχεια της έρευνας μας πέρα από την παρούσα εργασία παρουσιάζουν ενδιαφέρον τα εξής:

- Πραγματοποίηση ανίχνευσης ανωμαλιών υλοποιώντας αλγορίθμους forecasting
- Συνδυασμός όλων των μεθόδων για την παραγωγή καλύτερων αποτελεσμάτων
- Διαχωρισμός και αυτόματη επιλογή καλύτερων μοντέλων (auto-ML)
- Υλοποίηση της προτεινόμενης αρχιτεκτονικής πλαισίου

Επίλογος

Η εργασία με θέμα "Αναγνώριση μη φυσιολογικών προτύπων και τιμών σε συστήματα IoT για την ανίχνευση βλαβών" διείσδυσε στον περίπλοκο κόσμο των συστημάτων IoT και στην κρίσιμη ανάγκη για αξιόπιστους μηχανισμούς ανίχνευσης σφαλμάτων. Μέσω της ανάπτυξης ενός ολοκληρωμένου πλαισίου, η έρευνα αυτή είχε ως στόχο να αντιμετωπίσει τις προκλήσεις που σχετίζονται με την αναγνώριση μη φυσιολογικών μοτίβων και τιμών σε περιβάλλοντα IoT, οδηγώντας τελικά σε αποτελεσματικότερη ανίχνευση σφαλμάτων και βελτιωμένη αξιοπιστία του συστήματος.

Το ταξίδι που ξεκίνησε η παρούσα διατριβή ξεκίνησε με την αναγνώριση του μετασχηματιστικού χαρακτήρα του Διαδικτύου των Πραγμάτων, το οποίο έχει φέρει επανάσταση στη συνδεσιμότητα και την ανταλλαγή δεδομένων μεταξύ αμέτρητων συσκευών. Καθώς τα συστήματα IoT συνεχίζουν να αναπτύσσονται σε διάφορους τομείς, η απαίτηση για αποτελεσματικούς μηχανισμούς ανίχνευσης σφαλμάτων έχει γίνει υψίστης σημασίας. Με την αναγνώριση μη φυσιολογικών προτύπων και τιμών, τα πιθανά σφάλματα μπορούν να εντοπιστούν προληπτικά, ελαχιστοποιώντας τον χρόνο διακοπής λειτουργίας του συστήματος και βελτιστοποιώντας τη λειτουργική αξιοπιστία.

Το προτεινόμενο πλαίσιο ενσωμάτωσε έναν συνδυασμό αλγορίθμων μηχανικής μάθησης, στατιστικής ανάλυσης και τεχνικών ανίχνευσης ανωμαλιών. Μέσω μιας εκτεταμένης ανάλυσης των αρχιτεκτονικών συστημάτων IoT, των πρωτοκόλλων και των μορφών δεδομένων, εντοπίστηκαν πιθανές πηγές σφαλμάτων και τα αντίστοιχα μοτίβα τους. Αυτή η ανάλυση άνοιξε το δρόμο για τη διερεύνηση ενός φάσματος αλγορίθμων μηχανικής μάθησης, συμπεριλαμβανομένων τεχνικών με επίβλεψη, χωρίς επίβλεψη και με ημι-επίβλεψη, για την ακριβή διάκριση μεταξύ κανονικών και μη κανονικών προτύπων ή τιμών.

Για την καταγραφή των χρονικών εξαρτήσεων και των αλληλεπιδράσεων εντός των δεδομένων IoT, χρησιμοποιήθηκαν μέθοδοι στατιστικής ανάλυσης. Αυτές οι τεχνικές βελτίωσαν περαιτέρω την ικανότητα του πλαισίου να εντοπίζει ανωμαλίες και να αναγνωρίζει αποκλίσεις από την αναμενόμενη συμπεριφορά, διευκολύνοντας την ακριβέστερη ανίχνευση σφαλμάτων. Η αξιολόγηση του πλαισίου με τη χρήση ενός πραγματικού συνόλου δεδομένων IoT που ελήφθη από ένα βιομηχανικό περιβάλλον, συγκεκριμένα από ένα εργοστάσιο παραγωγής πλαστικών, παρείχε απτές αποδείξεις για την αποτελεσματικότητά του στην ανίχνευση μη φυσιολογικών προτύπων και τιμών, ελαχιστοποιώντας παράλληλα τα ψευδώς θετικά και τα ψευδώς αρνητικά αποτελέσματα.

Επιπλέον, η εργασία πρότεινε μια απλή αρχιτεκτονική IoT που θα μπορούσε να ενσωματώσει το πλαίσιο που αναπτύχθηκε και να αυτοματοποιήσει τη διαδικασία ανίχνευσης ανωμαλιών. Αυτή η ενσωμάτωση γεφύρωσε το χάσμα μεταξύ θεωρίας και πρακτικής εφαρμογής, αναδεικνύοντας τις δυνατότητες του πλαισίου για ανάπτυξη στον πραγματικό κόσμο.

Συμπερασματικά, η έρευνα που διεξήχθη στο πλαίσιο της παρούσας εργασίας συνέβαλε σημαντικά στην πρόοδο της αξιοπιστίας και των δυνατοτήτων ανίχνευσης σφαλμάτων του IoT. Με την αναγνώριση μη φυσιολογικών προτύπων και τιμών στα συστήματα IoT, μπορούν να δρομολογηθούν έγκαιρες ενέργειες παρέμβασης και συντήρησης για την ελαχιστοποίηση των διαταραχών του συστήματος, τη βελτίωση της λειτουργικής απόδοσης και τη βελτίωση της συνολικής εμπειρίας των χρηστών στο διασυνδεδεμένο τοπίο IoT.

Καθώς το IoT συνεχίζει να εξελίσσεται, η μελλοντική έρευνα μπορεί να βασιστεί στα θεμέλια που θέτει η παρούσα εργασία, διερευνώντας προηγμένες τεχνικές ανίχνευσης ανωμαλιών, βελτιστοποιώντας την απόδοση του πλαισίου και προσαρμόζοντάς το σε διάφορους τομείς εφαρμογών IoT. Η συνεχής βελτίωση και εφαρμογή των μηχανισμών ανίχνευσης σφαλμάτων είναι καθοριστικής σημασίας για τη βιώσιμη ανάπτυξη και την ευρεία υιοθέτηση των συστημάτων IoT, εξασφαλίζοντας την αξιόπιστη λειτουργία τους και τον θετικό αντίκτυπο στις βιομηχανίες και τις κοινωνίες παγκοσμίως.

Βιβλιογραφία

Aggarwal, C. C., 2017. *Outlier Analysis.* 2 ed. s.l.:Springer Cham.

Atzori, L., Iera, A. & Morabito, G., 2010. The Internet of Things: A survey. *Computer Networks,* 54(15).

Borgia, E., 2014. The Internet of Things vision: Key features, applications and open issues. *Computer Communications.*

Brownlee, J., 2018. *A Gentle Introduction to Statistical Data Distributions..* s.l.:s.n.

Burns, B. et al., 2013. *Kubernetes: A Platform for Automating Deployment, Scaling, and Operations of Application Containers.* Proceedings of the 19th ACM SIGOPS Symposium on Operating Systems Principles, s.n.

Erl, T., Puttini, R. & Mahmood, Z., 2013. *Cloud Computing: Concepts, Technology & Architecture.* s.l.:Prentice Hall.

Feasel, K., 2022. *Finding Ghosts in Your Data: Anomaly Detection Techniques with Examples in Python.* 1 ed. s.l.:Apress Berkeley, CA.

Grubbs, F. E., 1969. Procedures for Detecting Outlying Observations in Samples.. *Technometrics,* Issue 11.

Gubbi, J., Buyya, R., Marusic, S. & Palaniswami, M., 2013. Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Generation Computer Systems,* 29(7).

Hofstee, H. P., 2020. *Edge Computing: The Next Frontier in High-Performance Computing.* s.l.:CRC Press.

Jukan, A. & Ganchev, I., 2017. Anomaly detection and fault diagnosis in IoT systems: Methods, architectures, and opportunities. *IEEE Internet of Things Journal.*

Kafka®, A., 2021. *Apache Software Foundation.* [Online]
Available at: https://kafka.apache.org/

Ranger, S., 2020. *What is the IoT? Everything you need to know about the Internet of Things right now.* [Online]
Available at: https://www.zdnet.com/article/what-is-the-internet-of-things-everything-you-need-to-know-about-the-iot-right-now/

Shvachko, K., Kuang, H., Radia, S. & Chansler, R., 2010. *The Hadoop Distributed File System.* s.l., 2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST).

Solomon, H., Marmol, J. & Golub, J., 2014. *Docker: Lightweight Linux Containers for Consistent Development and Deployment.* s.l., Proceedings of the Linux Symposium.

Sridhar, A. & Suman, K. A., 2019. *Beginning Anomaly Detection Using Python-Based Deep Learning.* 1 ed. s.l.:Apress Berkeley, CA.

Tang, Y., Liu, A., Zou, T. & & Xiang, Y., 2019. Anomaly detection for industrial Internet of Things systems: A deep learning approach. *IEEE Transactions on Industrial Informatics.*

Yan, Z., Zhang, P. & Vasilakos, A., 2014. A survey on trust management for Internet of Things. *Journal of Network and Computer Applications,* Volume 42.

Zhou, B., Chen, L., Wang, C. & Wang, X., 2018. Anomaly detection in IoT systems: A survey.. *Journal of Network and Computer Applications.*