ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΙΓΑΙΟΥ
ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΑΚΩΝ ΚΑΙ ΕΠΙΚΟΙΝΩΝΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

Ανακάλυψη ανωμαλιών σε δεδομένα χρονοσειρών

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

Βασίλειου Πετρόπουλου

Επιβλέπων : Μαραγκουδάκης Εμμανουήλ

Σάμος, Μάιος 2020

Η σελίδα αυτή είναι σκόπιμα λευκή.

Πρόλογος και ευχαριστίες

Η παρούσα εργασία με τίτλο « ***Ανακάλυψη ανωμαλιών σε δεδομένα χρονοσειρών*** » εκπονήθηκε από τον φοιτητή Βασίλειο Πετρόπουλο σε μερική εκπλήρωση των απαιτήσεων για το δίπλωμα του Μηχανικού Πληροφοριακών και Επικοινωνιακών Συστημάτων, με εποπτεία του καθηγητή του τμήματος κ. Μανόλη Μαραγκουδάκη. Είναι αποτέλεσμα έντονης ζύμωσης γύρω από το θέμα, το χρονικό διάστημα από τον Σεπτέμβριο 2019 έως τον Μάιο 2020.

Η εργασία για λειτουργικούς και πρακτικούς λόγους διαχωρίστηκε σε πέντε (5) επί μέρους στάδια. Τα αναφέρω επιγραμματικά και θα αναλυθούν στη συνέχεια.

(α) *Μελέτη του τι είναι η ανακάλυψη ανωμαλιών σε χρονοσειρές, (β) μελέτη για το ποιοι αλγόριθμοι υπάρχουν (οι 5 ποιο σημαντικοί) και τα βασικά τους σημεία, (γ) συλλογή κώδικα για κάθε αλγόριθμο, (δ) δημιουργία εφαρμογής που ενσωματώνει όλους αυτούς του αλγορίθμους (σε python) και (ε) πειράματα με γνωστές χρονοσειρές που έχουν ανωμαλίες και συγκριτική αντιπαράθεση τους.*

Επιπροσθέτως θα ήθελα εκφράσω τις ευχαριστίες μου στον επιβλέποντα καθηγητή μου, κ. Μανόλη Μαραγκουδάκη που μου έδωσε την ευκαιρία να ασχοληθώ με ένα τόσο ενδιαφέρον θέμα. Ακόμα τον ευχαριστώ που στάθηκε αρωγός σε όλα τα στάδια της ολοκλήρωσης αυτής της εργασίας, με τη βοήθεια και τις χρήσιμες συμβουλές του.

Περισσότερο από όλους, οφείλω να ευχαριστήσω την οικογένεια μου, διότι χωρίς εκείνους η απόκτηση του διπλώματος θα ήταν για μένα ένα πολύ δύσκολο εγχείρημα. Τους ευχαριστώ που στάθηκαν δίπλα μου όλα αυτά τα χρόνια και για την υπομονή που υπέδειξαν, μέχρι την επιστροφή μου στην οικογενειακή εστία.

του

ΒΑΣΙΛΕΙΟΥ ΠΕΤΡΟΠΟΥΛΟΥ

Τμήμα Μηχανικών Πληροφοριακών και Επικοινωνιακών Συστημάτων

ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΙΓΑΙΟΥ

Η σελίδα αυτή είναι σκόπιμα λευκή.

Πίνακας περιεχομένων

Λίστα Εικόνων

Ακρωνύμια

TS	Time Series
SVM	Support-Vector Machine
PAA	Piecewise Aggregate Approximation
S.A.X.	Symbolic Aggregate Approximation
OC-SVM	One Class-Support Vector Machine
RBF	Radial Basis Function
MSLE	Mean Squared Logarithmic Error
Znorm	Z normalization
AI	Artificial Intelligence
ML	Machine Learning
ΕΦΓ	Επεξεργασία Φυσικής Γλώσσας
NLP	Natural Language Processing
KNN	K Nearest Neighbors
NN	Neural Network
JSP	Java Server Pages
ΜΜΜ	Μέσα Μαζικής Μεταφοράς

Περίληψη

Η αναγκαιότητα των εφαρμογών της τεχνητής νοημοσύνης (AI) είναι κάτι που μεγαλώνει όσο η κοινωνία εξελίσσεται. Είναι πλέον ένα ισχυρό κομμάτι της καθημερινότητάς μας και την συναντάμε σε εφαρμογές και λειτουργίες, πολλές φορές χωρίς να το καταλαβαίνουμε. Όταν τίθεται θέμα ασφάλειας προσωπικών δεδομένων ή όταν πρόκειται για θέματα υγείας η τεχνητή νοημοσύνη αποτελεί το καταλληλότερο μέσο και πολλές φορές χωρίς να το ξέρουμε υπάρχει ακόμα και σε μια απλή εξέταση. Μεταξύ πολλών άλλων, τεχνολογίες AI είναι η αναγνώριση προσώπου και εικόνας (πολλά έξυπνα τηλέφωνα ξεκλειδώνουν πια αναγνωρίζοντας το πρόσωπο του κατόχου τους, το Facebook αναγνωρίζει στα πρόσωπα μιας φωτογραφίας που ανεβάζει ο χρήστης τους φίλους του και προτείνει tag), η αναγνώριση ασυνήθιστων κινήσεων στο δίκτυο μπορεί να προβλέψει μια εν δυνάμει απάτη πιστωτικών καρτών, η αυτόματη μετάφραση κειμένου, η εμφάνιση στοχευμένων διαφημιστικών μηνυμάτων, οι προσωποποιημένες προτάσεις ειδήσεων, ή το Google maps, που εντοπίζει διευθύνσεις και παρέχει δεδομένα κυκλοφορίας.

Γίνεται επομένως αντιληπτό, ότι η τεχνητή νοημοσύνη μας βοηθά στην ποιότητα της ζωής μας. Ένας ακόμα τομέας που έχει ισχυρή παρουσία είναι η αναγνώριση ανωμαλιών. Όταν αυτές οι μη- κανονικές τιμές που δεν συμβαδίζουν με το υπόλοιπο σύνολο των δεδομένων μπορούν να βρεθούν αυτοματοποιημένα χωρίς να παιδεύεται ο άνθρωπος ή ακόμα και χωρίς να το περιμένει (από απλές ενημερώσεις στο κινητό για το αν θα βρέξει σε 1 ώρα από τώρα, μέχρι τις αλλαγές και τις προβλέψεις των τιμών του χρηματιστηρίου), τότε η ζωή γίνεται πιο εύκολη και ο άνθρωπος εστιάζει σε πράγματα που είναι πραγματικά άξιο να ξοδεύει το χρόνο του. Αυτό κάνουν και οι μέθοδοι αναγνώρισης ανωμαλιών, μερικές εκ των οποίων έχουν χρησιμοποιηθεί και αναλυθεί στην παρούσα διπλωματική. Για να γίνει πιο σαφές το περιεχόμενό της, υλοποιήθηκε μια εφαρμογή web που ενσωματώνει κατά κύριο λόγο λειτουργίες για αναγνώριση ανωμαλιών σε δεδομένα χρονοσειρών.

Keywords: *Τεχνητή νοημοσύνη, ανακάλυψη ανωμαλιών, χρονοσειρές, δεδομένα, χαρακτηριστικά, αναγνώριση προσώπου, έξυπνη αναζήτηση, στοχευμένες διαφημίσεις, προσωποποιημένες προτάσεις, διαδικτυακή εφαρμογή.*

Abstract

The necessity of artificial intelligence (AI) applications is growing as society evolves. It is indeed now, a powerful part of our daily routine and we find it in applications and functions of our everyday life, some times without even realizing it. When it comes to personal data security or health issues, artificial intelligence is the most appropriate mean to use, and many times without knowing it is there, at a simple health test. Among other things, AI technologies are face and image recognition (many smartphones now unlock by recognizing their owner's face, Facebook recognizes the faces in a photo that the user uploads his friends and suggests a tag), the recognition of unusual network movement may foresee a potential credit card fraud, automatic text translation, the appearance of targeted advertising messages, personalized news suggestions, or Google maps, which identifies addresses and provides traffic data.

It is therefore understood that artificial intelligence helps us in our life's quality. Another area that AI has a really strong presence is anomaly detection/recognition. When these abnormal values that do not match the rest of the data can be found automatically without effort (simple mobile notifications with the weather forecast about the rain 1 hour from now or changes and forecasts of stock market prices), then life becomes easier and man focuses on things that are really worth spending his time on. This is what anomaly detection methods do, some of which have been used and analyzed in this Thesis. To make its content easier to understand, a web application has been implemented that mainly includes features for recognizing anomalies in time series data.

Keywords: *Artificial Intelligence, anomaly detection, Time series, data, features, face recognition, smart search, targeted advertising messages, personalized suggestions, web application.*

1 *Εισαγωγή*

1.1 Καθορισμός του προβλήματος

Χρονοσειρά είναι μια ακολουθία δεδομένων διακριτού χρόνου. Δηλαδή είναι δεδομένα πάνω στη συμπεριφορά μιας μεταβλητής σε διαδοχικές χρονικές περιόδους και παρουσιάζονται πολύ συχνά μέσω γραφημάτων.
Στην εξόρυξη δεδομένων, η ανίχνευση ανωμαλιών είναι η αναγνώριση σπάνιων αντικειμένων, γεγονότων ή παρατηρήσεων που εγείρουν υποψίες και διαφέρουν σημαντικά από την πλειονότητα των υπόλοιπων δεδομένων.

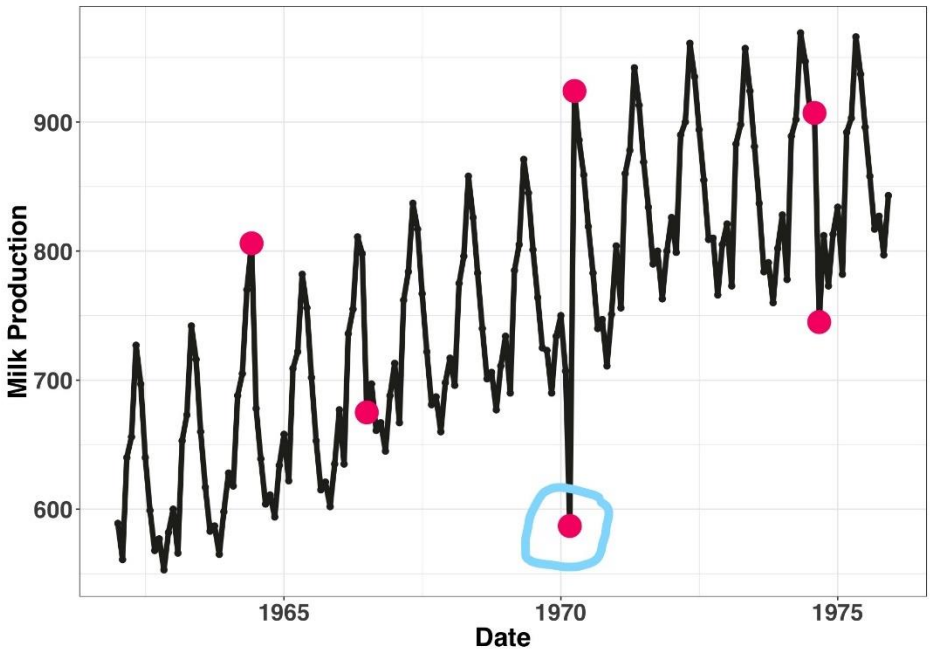


Figure 1: Milk production ts

Στην παραπάνω εικόνα απεικονίζεται η παραγωγή γάλατος από το 1960 μέχρι το 1976 περίπου. Κάθε σημείο (μαύρες κουκίδες) της χρονοσειράς είναι η τιμή που πήρε την δεδομένη χρονική στιγμή. Οι κόκκινες κουκίδες είναι διαφορετικές μη αναμενόμενες τιμές που πήρε

συγκεκριμένες χρονικές στιγμές η παραγωγή, κοινώς ανωμαλίες. Για παράδειγμα στις αρχές του 1970 (η κόκκινη κουκίδα που είναι κυκλωμένη με μπλε χρώμα) η τιμή αυτή μας δείχνει ότι η παραγωγή γάλατος (λίγο κάτω από 600) έχει πέσει πάρα πολύ σε σχέση με την αμέσως προηγούμενη μέτρηση (λίγο πάνω από 700). Αυτό πρακτικά μας δείχνει ότι εκείνη την εποχή η παραγωγή γάλατος ήταν εξαιρετικά μειωμένη, πράγμα που μπορεί να επηρέασε και άλλους τομείς της καθημερινότητας. Ίσως την αγορά. (Μειωμένη παραγωγή συνήθως σημαίνει αύξηση της τιμής πώλησης ώστε να μην έχουν μειωμένα έσοδα οι παραγωγοί.)

Αξίζει να σημειωθεί, ότι η μελέτη των χρονοσειρών είναι συνδεδεμένη με πολλούς τομείς της καθημερινότητάς μας. Πολλοί επιστήμονες ή ακόμα και απλοί καθημερινοί άνθρωποι μελετούν καθημερινά τις χρονοσειρές για να εξάγουν κάποια συμπεράσματα για να βελτιώσουν την ποιότητα της ζωής τους.

Χρηματιστήριο: Δεν χρειάζεται ιδιαίτερη γνώση για να καταλάβει κανείς ότι οι τιμές που παρουσιάζονται στα διαγράμματα του χρηματιστηρίου είναι με βάση τη χρονική περίοδο. Κάποιος που παρακολουθεί το χρηματιστήριο, διαβάζει αναλύσεις και μελετά τα διαγράμματα των τιμών για κάποια χρονική περίοδο, αποφασίζει βλέποντας την χρονοσειρά αν θα αγοράσει, αν θα πουλήσει ή αν θα μείνει αμέτοχος. Καταλαβαίνει επίσης πότε η μετοχή έχει πτώση ή άνοδο και αν αυτό είναι φυσιολογικό με βάση τις προηγούμενες μετρήσεις. Οι πιο εξοικειωμένοι και κάποιοι ειδικοί ενδεχομένως μπορούν ακόμα και να προβλέψουν την πιθανή κίνηση της τιμής στο μέλλον με βάση τα έως τώρα δεδομένα που παρατηρούν από τη χρονοσειρά.

Οικονομία και τράπεζες: Στον τομέα αυτόν οι χρονοσειρές είναι εξίσου ένα σημαντικό κομμάτι. Μια αδικαιολόγητη για παράδειγμα αλλαγή στον τρόπο που παρουσιάζονται οι τιμές με βάση το χρόνο, δηλαδή μια αύξηση του μεγέθους της τιμής που μετράει τη χρήση του δικτύου των συναλλαγών σε μία συγκεκριμένη χρονική στιγμή, μπορεί να κρύβει μια εν δυνάμει απάτη πιστωτικών καρτών.

Υγεία: Στο κομμάτι της υγείας υπάρχουν συλλογές χρονοσειρών για μελέτη, που περιλαμβάνουν εκτός των άλλων, για κάθε γενικό νοσοκομείο ποσοστό καταλήψεως κλινών, μέσο όρο παραμονής των ασθενών, εσωτερικοί ασθενείς ανά μονάδα, μελέτες δηλαδή που προσφέρουν στον αναγνώστη τη γνώση που χρειάζεται να έχουν σε περίπτωση που χρειαστεί να νοσηλευτούν ή να χειρουργηθούν. Ακόμα υπάρχουν χρονοσειρές που αποτυπώνουν τον πληθυσμό των ασθενών ανά γιατρό, νοσοκόμο ή κλίνη η ακόμα και το ποσό των συνολικών δαπανών των υπηρεσιών υγείας. Όλα αυτά οδηγούν τους αρμόδιους στην καλύτερη διαχείριση ιατρικών αποθεμάτων, προσωπικού και χρήματων με στόχο την καλύτερη εξυπηρέτηση του κάθε ανθρώπου ξεχωριστά και την βέλτιστη αντιμετώπιση υγειονομικών κρίσεων. Τρανό παράδειγμα αποτελεί η πρόσφατη πανδημία του Κορωνοϊού που μελετάται καθημερινά μέσω στατιστικών χρονοσειρών με σκοπό την εύρεση φαρμάκων και εμβολίου. Οι επιστήμονες χωρίς αυτά τα στατιστικά θα βάδιζαν στα τυφλά.(Τέτοιες χρονοσειρές είναι η επίδραση ενός φαρμάκου σε κάποιο ποσοστό ανθρώπων, οι μέρες που κάνει ένας ασθενής να νοσήσει αφού προσβληθεί από τον ίο ή ο χρόνος ανάρρωσης και ο τρόπος μετάδοσης του ιού ανά εποχή.)

Περιβάλλον: Η ποιότητα της ζωής του σύγχρονου ανθρώπου εξαρτάται στον μεγαλύτερο βαθμό από το περιβάλλον που ζει. Είναι λοιπόν αναγκαίο να μελετώνται οι περιβαλλοντικές παράμετροι που την επηρεάζουν άμεσα. Η μελέτη των χρονοσειρών που αποτυπώνουν για παράδειγμα την

ποιότητα του αέρα σε ορισμένες περιπτώσεις μπορεί να οδηγήσουν τους επιστήμονες στο συμπέρασμα ότι υπάρχει μια ρύπανση ή κάποια διαρροή αερίου από εργοστάσια που χρήζει αντιμετώπισης για να μην πλήττεται η υγεία του ανθρώπου. Ένα τέτοιο project έχει αναλάβει και το πανεπιστήμιο Θεσσαλίας, το τμήμα μηχανολόγων μηχανικών που έχει εγκαταστήσει ένα δίκτυο σταθμών μέτρησης ατμοσφαιρικής ρύπανσης και παρακολουθεί την ημερήσια διακύμανση των αιωρούμενων σωματιδίων PM10 και εξάγει τα αποτελέσματα καθημερινά με μετρήσεις ανά 5 λεπτά σε μορφή χρονοσειρών.

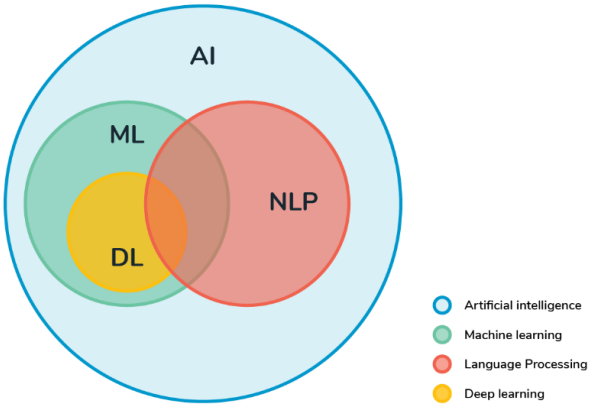
Γεωπονία: Στον τομέα της γεωπονίας αυτό που είναι αξιοσημείωτο είναι ότι πολλοί νέοι αγρότες αξιοποιώντας την τεχνολογία και την τεχνητή νοημοσύνη μελετούν με ειδικούς αισθητήρες που τοποθετούνται στα χωράφια τη σύσταση του χώματος, τα ποσοστά θρέψης του φυτού ανάλογα με τις ουσίες που προσλαμβάνει και πολλές ακόμα μετρήσεις, οι οποίες προβάλλονται στον άνθρωπο με τη μορφή χρονοσειρών ώστε να είναι πιο ακριβής και εύκολη η μελέτη των αποτελεσμάτων. Με τον τρόπο αυτόν ο αγρότης είναι σε θέση να γνωρίζει ποια χρονική περίοδο και κάτω από ποιες περιβαλλοντικές συνθήκες η παραγωγή του παρουσίασε την επιθυμητή απόδοση, έτσι μελλοντικά προσαρμόζει το χρόνο και την ποσότητα της φροντίδας (πότισμα, λίπανση) πιο κοντά σε περιόδους που την προηγούμενη χρονιά η παραγωγή έδειξε πιο υγιής και αυξήθηκε σημαντικά .

Καθημερινότητα: Η μελέτη των χρονοσειρών αποτελεί αναπόσπαστο κομμάτι της καθημερινής μας ρουτίνας. Πολλές εφαρμογές που χρησιμοποιούμε καθημερινά μας κοινοποιούν αποτελέσματα εξαγόμενα από χρονοσειρές. Ένα απλό παράδειγμα τέτοιων εφαρμογών είναι οι εφαρμογές για τα Μέσα μαζικής μεταφοράς (ΜΜΜ) που χρησιμοποιούν χιλιάδες άνθρωποι καθημερινά. Εάν παρατηρήσει κανείς τους χάρτες που παρουσιάζουν πολύ πιθανό να δει ότι σε συγκεκριμένες χρονικές περιόδους και σε συγκεκριμένες στάσεις το λεωφορείο για παράδειγμα αργεί λίγο παραπάνω. Αυτό προκύπτει μέσα από ανάλυση μιας χρονοσειράς που καταγράφει τη διέλευση των οχημάτων. Έτσι σύμφωνα με παλιά δεδομένα το σύστημα αναγνωρίζει ότι αφού τις προηγούμενες μέρες (λόγω αυξημένης κίνησης σε ώρες αιχμής σε συγκεκριμένα σημεία για παράδειγμα), είναι πολύ μεγάλη η πιθανότητα ο δρόμος αυτός να έχει και πάλι μία καθυστέρηση στη συγκεκριμένη στάση για τη συγκεκριμένη χρονική περίοδο. Ακόμη ένα κομμάτι αξιοποίησης των αποτελεσμάτων που περιέχει μια χρονοσειρά είναι και συγκεκριμένες συμπεριφορές της αγοράς. Για παράδειγμα αν μελετήσει κάποιος μία χρονοσειρά που περιέχει τιμές για κάθε μήνα για μια περίοδο πολλών χρόνων θα παρατηρήσει ότι υπάρχει μια τάση κάθε χρόνο τον μήνα Δεκέμβριο οι τιμές να ανεβαίνουν λόγω Χριστουγέννων. Έτσι μπορεί ο καθένας να κάνει τον προϋπολογισμό του με βάση αυτή τη μελέτη και να κρατήσει κάποια έξτρα χρήματα για την περίοδο αυτή.

Είναι επομένως προς όφελος του ανθρώπου να μελετά χρονοσειρές βασισμένες σε καθημερινά γεγονότα που μπορεί να κάνουν τη ζωή του πιο εύκολη, να βελτιώσουν πτυχές της καθημερινότητάς του η να «προβλέψουν» γεγονότα με βάση παλαιότερες συμπεριφορές. Το να μελετάμε γεγονότα που έχουν αποτυπωθεί χρονικά με μαθηματική ακρίβεια όχι μόνο διευκολύνει τον τρόπο ζωής, αλλά κάνει και πιο ασφαλή την συμβίωση μας.

1.2 Τεχνητή νοημοσύνη και μηχανική μάθηση.

Ο όρος **τεχνητή νοημοσύνη** αναφέρεται στον κλάδο της επιστήμης των υπολογιστών ο οποίος ασχολείται με τη σχεδίαση και την υλοποίηση υπολογιστικών συστημάτων που μιμούνται στοιχεία της ανθρώπινης συμπεριφοράς τα οποία υπονοούν έστω και στοιχειώδη ευφυΐα: μάθηση, προσαρμοστικότητα, εξαγωγή συμπερασμάτων, κατανόηση από συμφραζόμενα, επίλυση προβλημάτων κλπ. [1] Καθιστά τις μηχανές ικανές να μαθαίνουν από την εμπειρία, να προσαρμόζονται σε νέα εισαγόμενα δεδομένα και να εκτελούν ανθρωπομορφικά έργα. Τα περισσότερα παραδείγματα AI (AI από Artificial Intelligence) για τα οποία ακούτε σήμερα –από τους υπολογιστές που παίζουν σκάκι έως τα αυτο-οδηγούμενα αυτοκίνητα– βασίζονται σε μεγάλο βαθμό στο deep learning και την επεξεργασία φυσικής γλώσσας (ΕΦΓ). Υπό μια απλή και εκλαϊκευμένη ερμηνεία είναι «η επιστήμη που κάνει τις μηχανές πιο έξυπνες». Μέσω της τεχνητής νοημοσύνης οι ιστοσελίδες, το διαδίκτυο, οι ηλεκτρονικοί υπολογιστές, οι τηλεοράσεις, ένα ρολόι ή ένα κινητό τηλέφωνο αποκτούν τη δυνατότητα να αναπτύσσουν μια κάποια μορφή… σκέψης.



Εικόνα 1: Τεχνητή Νοημοσύνη

1.2.1 Γιατί είναι σημαντική η τεχνητή νοημοσύνη (AI); [2]

- **Το AI αυτοματοποιεί την επαναληπτική μάθηση και την ανακάλυψη μέσω δεδομένων.** Αντί των αυτοματοποιημένων χειροκίνητων έργων, το AI εκτελεί συχνά, μεγάλου όγκου μηχανογραφημένα έργα, αξιόπιστα και χωρίς κόπο. Για αυτόν τον τύπο της αυτοματοποίησης, η ανθρώπινη έρευνα εξακολουθεί να είναι καίριας σημασίας για να εγκατασταθεί το σύστημα και να τεθούν οι κατάλληλες ερωτήσεις.
- **Το AI προσθέτει ευφυΐα** στα υπάρχοντα προϊόντα. Στις περισσότερες περιπτώσεις, το AI δεν πωλείται ως μεμονωμένη εφαρμογή. Πιθανότερο είναι οι ικανότητες του AI να βελτιώνουν τα προϊόντα που ήδη χρησιμοποιείτε, όπως η πρόσθεση της ψηφιακής βοηθού Siri ως λειτουργία στα προϊόντα Apple νέας γενιάς. Η αυτοματοποίηση, οι πλατφόρμες συνομιλίας και οι έξυπνες μηχανές μπορούν να συνδυαστούν με μεγάλες ποσότητες δεδομένων προκειμένου να βελτιώνουν πολλές τεχνολογίες στο σπίτι και τον χώρο εργασίας από την ασφάλεια πληροφοριών έως την ανάλυση.

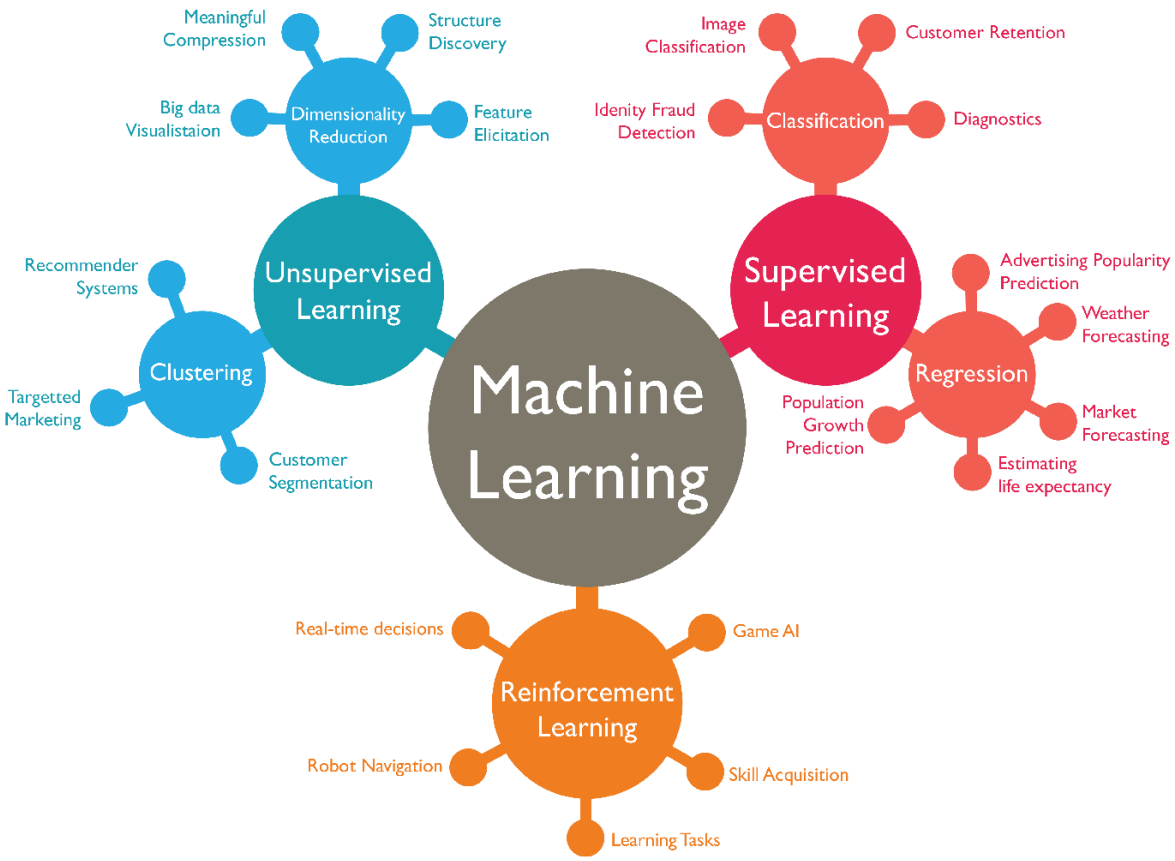
- **Το AI προσαρμόζεται μέσω προοδευτικών αλγορίθμων εκμάθησης (learning algorithms)** ώστε να αφεθούν τα δεδομένα να κάνουν τον προγραμματισμό. Το AI βρίσκει δομή και κανονικότητες στα δεδομένα οπότε ο αλγόριθμος αποκτά μια δεξιότητα: Ο αλγόριθμος ταξινομεί ή κατηγοριοποιεί. Έτσι, όπως ο αλγόριθμος μπορεί να διδάξει τον εαυτό του τον τρόπο να παίζει σκάκι, μπορεί και να τον διδάξει ποιο προϊόν να συστήσει στη συνέχεια διαδικτυακά. Και τα μοντέλα προσαρμόζονται όταν τους δίνονται νέα δεδομένα. Η οπισθοδιάδοση (back–propagation) είναι μια τεχνική AI που επιτρέπει στο μοντέλο να προσαρμόζεται, μέσω εκπαίδευσης και προστιθέμενων δεδομένων όταν η πρώτη απάντηση δεν είναι η ενδεδειγμένη.
- **Το AI αναλύει περισσότερα και βαθύτερα δεδομένα** με χρήση neural networks (νευρωνικά δίκτυα) που διαθέτουν πολλά κρυφά επίπεδα. Η κατασκευή ενός συστήματος ανίχνευσης απάτης με πέντε κρυφά επίπεδα ήταν σχεδόν αδύνατη λίγα χρόνια νωρίτερα. Όλα αυτά έχουν αλλάξει με την απίστευτη ισχύ των υπολογιστών και το μέγεθος των δεδομένων. Χρειάζεσαι μια πληθώρα δεδομένων για να εκπαιδεύσεις μοντέλα μάθησης σε βάθος, επειδή μαθαίνουν απευθείας από τα δεδομένα. Με όσο περισσότερα δεδομένα τα τροφοδοτείς, τόσο πιο ακριβή γίνονται.
- **Το AI επιτυγχάνει απίστευτη ακρίβεια** μέσω deep neural networks – κάτι που προηγουμένως ήταν αδύνατο. Για παράδειγμα, οι αλληλεπιδράσεις μας με το Google Search και το Google Photos βασίζονται στο deep learning – και συνεχίζουν να γίνονται πιο ακριβή όσο περισσότερο τα χρησιμοποιείτε. Στον ιατρικό τομέα, τεχνικές AI όπως το deep learning, η ταξινόμηση εικόνων και η αναγνώριση αντικειμένων μπορούν τώρα να χρησιμοποιηθούν για την ανίχνευση καρκίνου σε απεικονίσεις με μαγνητική τομογραφία, με ακρίβεια παρόμοια με αυτή καταρτισμένων ακτινολόγων.
- **Το AI αξιοποιεί στο έπακρο τα δεδομένα.** Όταν οι αλγόριθμοι είναι αυτο-εκπαιδευόμενοι, τα ίδια τα δεδομένα μπορούν να καταστούν πνευματική ιδιοκτησία. Οι απαντήσεις βρίσκονται στα δεδομένα, απλά, πρέπει να εφαρμόσετε μεθόδους τεχνητής νοημοσύνης για να τις ανακτήσετε.

Πηγή: [3]

1.2.2 Πώς ακριβώς λειτουργεί η Τεχνητή Νοημοσύνη (AI);

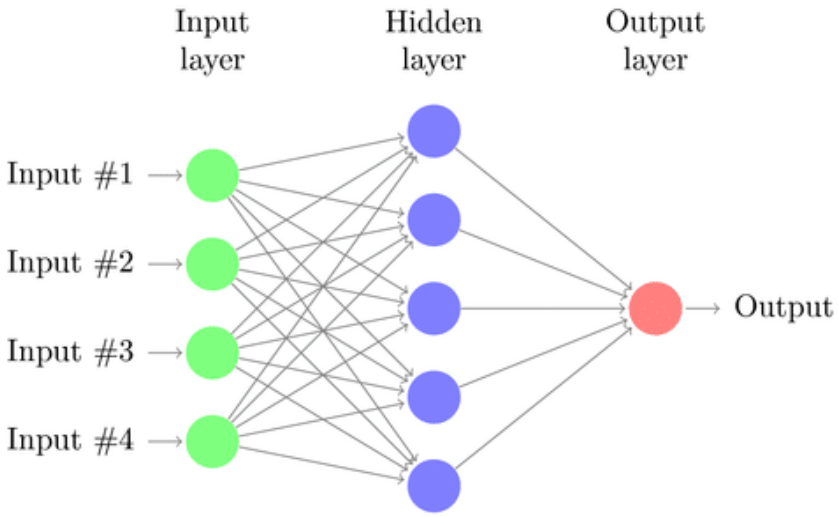
Η τεχνητή νοημοσύνη λειτουργεί με συνδυασμό μεγάλων ποσοτήτων δεδομένων με γρήγορους, επαναληπτικής διαδικασίας και ευφυείς αλγορίθμους, επιτρέποντας στο λογισμικό να μαθαίνει αυτόματα από μορφές ή χαρακτηριστικά των δεδομένων. Αποτελεί σημείο τομής μεταξύ πολλαπλών επιστημών όπως της πληροφορικής, της ψυχολογίας, της φιλοσοφίας, της νευρολογίας, της γλωσσολογίας και της επιστήμης μηχανικών, με στόχο τη σύνθεση ευφυούς συμπεριφοράς, με στοιχεία συλλογιστικής, μάθησης και προσαρμογής στο περιβάλλον, ενώ συνήθως εφαρμόζεται σε μηχανές ή υπολογιστές ειδικής κατασκευής. Η τεχνητή νοημοσύνη είναι ένα πεδίο μελέτης που περιλαμβάνει πολλές θεωρίες, μεθόδους και τεχνολογίες, καθώς και τα παρακάτω κύρια υποπεδία:

1. **Machine learning (μηχανική μάθηση):** αυτοματοποιεί την κατασκευή αναλυτικών μοντέλων. Χρησιμοποιεί μεθόδους από τα νευρωνικά δίκτυα (neural networks), τη στατιστική και τη φυσική για την εύρεση κρυφών γνώσεων εντός των δεδομένων χωρίς να έχει προγραμματιστεί εμφανώς για το πού να εξετάσει ή τι να συμπεράνει.



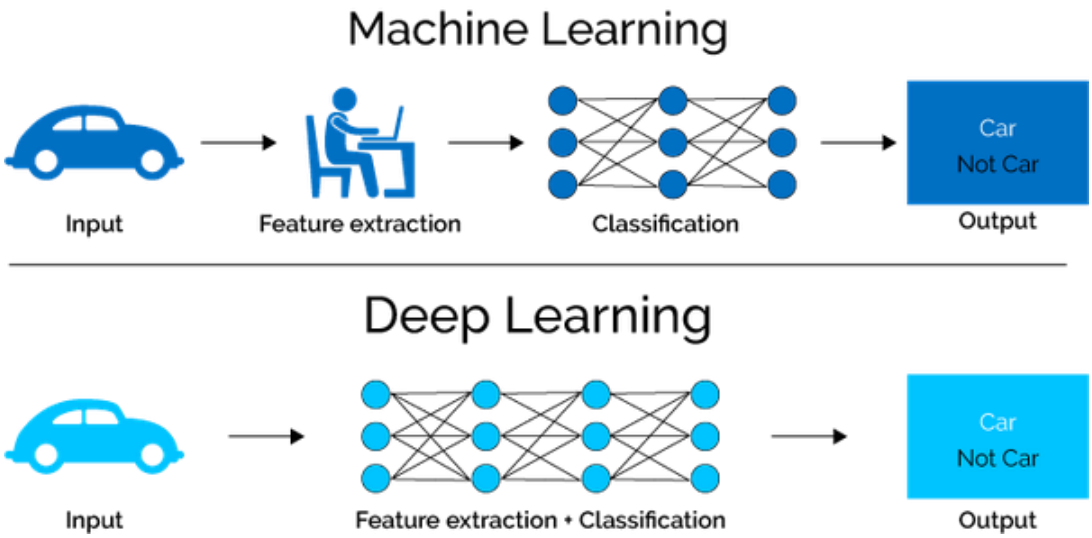
Εικόνα 2: Machine Learning

2. **Neural network (νευρωνικό δίκτυο):** είναι ένας τύπος μηχανικής μάθησης που αποτελείται από αλληλοσυνδεόμενες μονάδες (όπως οι νευρώνες) που επεξεργάζονται τις πληροφορίες ανταποκρινόμενες σε εξωτερικές εισαγωγές δεδομένων, προωθώντας πληροφορίες μεταξύ κάθε μονάδας. Η διαδικασία απαιτεί πολλαπλές διελεύσεις στα δεδομένα προκειμένου να βρεθούν συνδέσεις και να γίνει εξαγωγή νοήματος από ακαθόριστα δεδομένα.



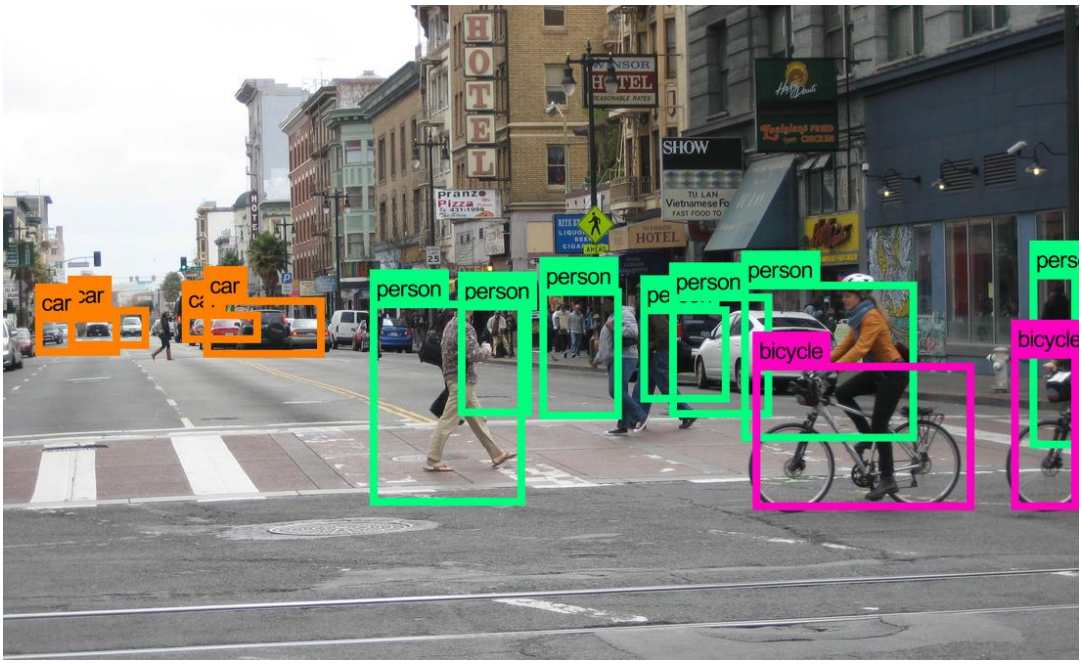
Εικόνα 3: Νευρωνικό Δίκτυο

3. **Deep learning (σε βάθος μάθηση):** Χρησιμοποιεί τεράστια neural networks με πολλά επίπεδα μονάδων επεξεργασίας, αξιοποιώντας τις εξελίξεις στην υπολογιστική ισχύ και τις βελτιωμένες τεχνικές εκπαίδευσης για την μάθηση πολύπλοκων μορφών σε μεγάλες ποσότητες δεδομένων. Οι κοινές εφαρμογές της περιλαμβάνουν την αναγνώριση εικόνας και ομιλίας.



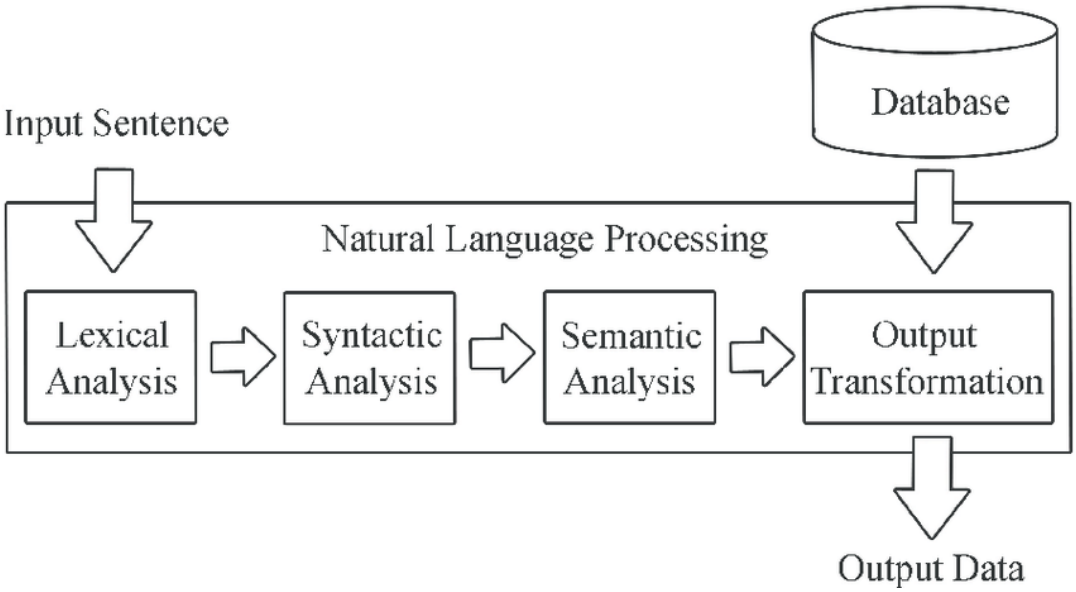
Εικόνα 4: Deep Learning

4. **Computer vision** : βασίζεται στην αναγνώριση μορφών (pattern recognition) και στο deep learning ώστε να αναγνωρίζεται τι υπάρχει σε μια εικόνα ή ένα βίντεο. Όταν οι μηχανές μπορούν να επεξεργαστούν, να αναλύσουν και να κατανοήσουν εικόνες μπορούν να συλλάβουν εικόνες ή βίντεο σε πραγματικό χρόνο και να ερμηνεύσουν τα περιβάλλοντά τους. Χρησιμοποιείται σε περιπτώσεις βιομετρικών σαρώσεων, αναγνώριση προσώπου, ίριδας, δαχτυλικού αποτυπώματος κτλ.



Εικόνα 5: Computer Vision

5. **Natural language processing** (NLP) - (επεξεργασία φυσικής γλώσσας ή ΕΦΓ): είναι η ικανότητα των υπολογιστών να αναλύουν, να κατανοούν και να παράγουν ομιλούμενη γλώσσα, συμπεριλαμβανομένης της ομιλίας. Το επόμενο στάδιο στην ΕΦΓ είναι η φυσική γλωσσική αλληλεπίδραση, η οποία επιτρέπει στους ανθρώπους να επικοινωνούν με υπολογιστές χρησιμοποιώντας την κανονική, καθημερινή γλώσσα για την εκτέλεση καθηκόντων.



Εικόνα 6: NLP

Πηγή: [3]

1.2.3 Machine learning (μηχανική μάθηση)

Μηχανική μάθηση [4] είναι υποπεδίο της επιστήμης των υπολογιστών που αναπτύχθηκε από τη μελέτη της αναγνώρισης προτύπων και της υπολογιστικής θεωρίας μάθησης στην τεχνητή νοημοσύνη. Διερευνά τη μελέτη και την κατασκευή αλγορίθμων που μπορούν να μαθαίνουν από τα δεδομένα και να κάνουν προβλέψεις σχετικά με αυτά. Γεννήθηκε από τη θεωρία ότι οι υπολογιστές μπορούν να μάθουν χωρίς να προγραμματίζονται για την εκτέλεση συγκεκριμένων εργασιών. Τέτοιοι αλγόριθμοι λειτουργούν κατασκευάζοντας μοντέλα από πειραματικά δεδομένα, προκειμένου να κάνουν προβλέψεις βασιζόμενες στα δεδομένα ή να εξάγουν αποφάσεις που εκφράζονται ως το αποτέλεσμα.

Η επαναληπτική πτυχή της μηχανικής μάθησης είναι σημαντική επειδή καθώς τα μοντέλα εκτίθενται σε νέα δεδομένα, είναι σε θέση να προσαρμοστούν ανεξάρτητα. Μαθαίνουν από προηγούμενους υπολογισμούς να παράγουν αξιόπιστες, επαναλαμβανόμενες αποφάσεις και αποτελέσματα. Κατά την τελευταία δεκαετία, η μηχανική μάθηση μας έδωσε αυτοκίνητα αυτο-οδηγούμενα, αναγνώριση ομιλίας, αποτελεσματική αναζήτηση στο διαδίκτυο και μια πολύ βελτιωμένη κατανόηση του ανθρώπινου γονιδιώματος.

Μερικά ευρέως δημοσιευμένα παραδείγματα εφαρμογών μηχανικής μάθησης είναι:

- ✓ Το self-driving αυτοκίνητο της Google.

- ✓ Διαδικτυακές προτάσεις ανάλογα τη χρήση. Όπως για παράδειγμα οι προτάσεις ταινιών με βάση αυτά που έχει δει ο χρήστης στο Netflix.
- ✓ Ανίχνευση απάτης σε τραπεζικές συναλλαγές.
- ✓ Η ολοκλήρωση φράσεων ή η επιλογή των σημείων στίξεως όταν γράφουμε ένα μήνυμα ή ένα email.
- ✓ Αυτόματη μετάφραση κειμένων.

Στη μηχανική μάθηση, ένας στόχος (target) ονομάζεται ετικέτα (label) ενώ στη στατιστική ο στόχος ονομάζεται εξαρτημένη μεταβλητή. Από την άλλη μεριά μια μεταβλητή στα στατιστικά ονομάζεται χαρακτηριστικό (feature) στη μηχανική μάθηση και ένας μετασχηματισμός (στα στατιστικά) ονομάζεται δημιουργία χαρακτηριστικών (feature creation) στη μηχανική μάθηση. [5]

1.2.3.1 Μέθοδοι μηχανικής μάθησης

Δύο από τις πιο διαδεδομένες μεθόδους μηχανικής μάθησης είναι η εποπτευόμενη μάθηση και η μη εποπτευόμενη μάθηση, αλλά υπάρχουν και άλλες μέθοδοι μηχανικής μάθησης. Η κάθε μία είναι κατάλληλη και συμπεριφέρεται καλύτερα για συγκεκριμένα tasks, έτσι είναι στην ευχέρεια του προγραμματιστή ποια θα χρησιμοποιήσει και γιατί.

1.2.3.1.1 Εποπτευόμενη μάθηση (supervised learning)

Οι αλγόριθμοι εκπαιδεύονται χρησιμοποιώντας labeled παραδείγματα, όπως μια είσοδο όπου είναι γνωστή η μία επιθυμητή έξοδο. Για παράδειγμα, ένα παράδειγμα θα μπορούσε να φέρει σημεία δεδομένων είτε "F" (Failed) είτε "R" (Runs) ή ακόμα και True/False ή 0/1, οτιδήποτε προδιαγράφει το σωστό στόχο ως αποτέλεσμα. Ο αλγόριθμος εκμάθησης λαμβάνει ένα σύνολο εισόδων μαζί με τις αντίστοιχες σωστές εξόδους και ο μαθαίνει συγκρίνοντας την πραγματική του έξοδο με τις σωστές εξόδους για να βρει λάθη. Στη συνέχεια τροποποιεί το μοντέλο και μέσω μεθόδων όπως η ταξινόμηση (classification), η παλινδρόμηση (regression) και η πρόβλεψη (prediction), η εποπτευόμενη μάθηση χρησιμοποιεί μοτίβα για να προβλέψει τις τιμές της ετικέτας σε πρόσθετα δεδομένα χωρίς ετικέτα (unlabeled). Η εποπτευόμενη μάθηση χρησιμοποιείται συνήθως σε εφαρμογές όπου ιστορικά δεδομένα προβλέπουν πιθανά μελλοντικά γεγονότα. Για παράδειγμα πόση ώρα θα σου πάρει να επιστρέψεις σπίτι από τη δουλειά, με δεδομένα της κίνησης των προηγούμενων ημερών.

1.2.3.1.1.1 Κατηγοριοποίηση (Classification):

Η ταξινόμηση είναι ένας τύπος εποπτευόμενης μάθησης. Καθορίζει την κλάση στην οποία ανήκουν τα στοιχεία δεδομένων και χρησιμοποιείται καλύτερα όταν η έξοδος έχει πεπερασμένες και διακριτές τιμές. Προβλέπει επίσης μια κλάση για μια μεταβλητή εισόδου. Είναι μια διαδικασία που σχετίζεται με την κατηγοριοποίηση, τη διαδικασία στην οποία οι ιδέες και τα αντικείμενα αναγνωρίζονται, διαφοροποιούνται και κατανοούνται. Οι τεχνικές ταξινόμησης προβλέπουν διακριτές αποκρίσεις - για παράδειγμα, εάν ένα email είναι γνήσιο ή ανεπιθύμητο, ή εάν ένας όγκος είναι καρκινικός ή καλοήθης. Τα μοντέλα ταξινόμησης ταξινομούν τα δεδομένα εισόδου σε κατηγορίες. Οι τυπικές εφαρμογές περιλαμβάνουν ιατρική απεικόνιση, αναγνώριση ομιλίας και βαθμολογία πίστωσης. Οι συνηθισμένοι αλγόριθμοι για την εκτέλεση της ταξινόμησης είναι SVM, decision trees, k-nearest neighbor, Naïve Bayes και neural networks.

Υπάρχουν 2 τύποι ταξινόμησης:

- Binomial
- Multi-Class

Χρησιμοποιήστε την ταξινόμηση εάν τα δεδομένα σας μπορούν να επισημανθούν, να κατηγοριοποιηθούν ή να διαχωριστούν σε συγκεκριμένες ομάδες ή τάξεις.

Πηγή: [6]

1.2.3.1.1.2 Παλινδρόμηση (regression):

Τα μοντέλα παλινδρόμησης χρησιμοποιούνται για να προβλέψουν μια συνεχή τιμή. Η πρόβλεψη των τιμών ενός σπιτιού δεδομένου των χαρακτηριστικών του σπιτιού όπως το μέγεθος, η τιμή κ.λπ. είναι ένα από τα κοινά παραδείγματα παλινδρόμησης. Είναι μια εποπτευόμενη τεχνική. Οι τεχνικές παλινδρόμησης προβλέπουν συνεχείς αποκρίσεις - για παράδειγμα, αλλαγές στη θερμοκρασία ή διακυμάνσεις στη ζήτηση ισχύος. Οι τυπικές εφαρμογές περιλαμβάνουν πρόβλεψη φορτίου ηλεκτρικής ενέργειας και αλγοριθμικές συναλλαγές.

Τύποι Παλινδρόμησης:

1. **Simple Linear Regression:** Εδώ προβλέπουμε μια μεταβλητή στόχου Y με βάση τη μεταβλητή εισόδου X. Θα πρέπει να υπάρχει γραμμική σχέση μεταξύ της μεταβλητής στόχου και της πρόβλεψης και έτσι έρχεται το όνομα Linear Regression. Για παράδειγμα σκεφτείτε το ενδεχόμενο να προβλέψετε τον μισθό ενός υπαλλήλου με βάση την ηλικία του.

$$Y = a + bX$$

2. **Polynomial Regression:** Στην πολυωνυμική παλινδρόμηση, μετατρέπουμε τα αρχικά χαρακτηριστικά σε πολυωνυμικά χαρακτηριστικά ενός δεδομένου βαθμού και στη συνέχεια εφαρμόζουμε το Linear Regression σε αυτό. Το παραπάνω γραμμικό μοντέλο Y = a + bX μετατρέπεται σε κάτι σαν:

$$Y = a+bX+cX^2$$

3. **Support Vector Regression:** Στο SVR, εντοπίζουμε ένα hyperplane με μέγιστο περιθώριο έτσι ώστε ο μέγιστος αριθμός σημείων δεδομένων να βρίσκεται εντός αυτού του περιθωρίου. Τα SVR είναι σχεδόν παρόμοια με τον αλγόριθμο ταξινόμησης SVM. Αντί να ελαχιστοποιήσουμε το ποσοστό σφάλματος όπως στην απλή γραμμική παλινδρόμηση, προσπαθούμε να προσαρμόσουμε το σφάλμα εντός ενός συγκεκριμένου ορίου. Στόχος μας στο SVR είναι βασικά να εξετάσουμε τα σημεία που βρίσκονται εντός του περιθωρίου.
4. **Decision Tree Regression:** Τα δέντρα αποφάσεων μπορούν να χρησιμοποιηθούν για ταξινόμηση και παλινδρόμηση. Σε δέντρα αποφάσεων, σε κάθε επίπεδο, πρέπει να προσδιορίσουμε το χαρακτηριστικό διαχωρισμού. Στην περίπτωση παλινδρόμησης, ο αλγόριθμος ID3 μπορεί να χρησιμοποιηθεί για τον προσδιορισμό του κόμβου διαχωρισμού μειώνοντας την τυπική απόκλιση. Ένα δέντρο αποφάσεων δημιουργείται με διαίρεση των δεδομένων σε υποσύνολα που περιέχουν παρουσίες με παρόμοιες τιμές (ομοιογενείς). Η τυπική απόκλιση χρησιμοποιείται για τον υπολογισμό της ομοιογένειας ενός αριθμητικού δείγματος. Εάν το αριθμητικό δείγμα είναι εντελώς ομοιογενές, η τυπική απόκλιση είναι μηδέν.

5. **Random Forest Regression:** Είναι μια συνολική προσέγγιση όπου λαμβάνουμε υπόψη τις προβλέψεις πολλών δέντρων παλινδρόμησης αποφάσεων.
    - Επιλέγουμε τυχαία σημεία K.
    - Προσδιορίζουμε το n όπου n είναι ο αριθμός των παλινδρομικών δέντρων αποφάσεων που θα δημιουργηθούν. Επαναλαμβάνουμε τα βήματα 1 και 2 για να δημιουργήσουμε πολλά δέντρα παλινδρόμησης.
    - Ο μέσος όρος κάθε κλάδου αντιστοιχεί στον κόμβο των φύλλων σε κάθε δέντρο αποφάσεων.
    - Για την πρόβλεψη εξόδου για μια μεταβλητή, λαμβάνεται υπόψη ο μέσος όρος όλων των προβλέψεων όλων των δένδρων αποφάσεων.

    Αποτρέπει το overfitting δημιουργώντας τυχαία υποσύνολα των χαρακτηριστικών και δημιουργώντας μικρότερα δέντρα χρησιμοποιώντας αυτά τα υποσύνολα.

Χρησιμοποιήστε τεχνικές παλινδρόμησης εάν εργάζεστε με ένα εύρος δεδομένων ή εάν η φύση της απόκρισης σας είναι ένας πραγματικός αριθμός, όπως η θερμοκρασία ή ο χρόνος μέχρι την αποτυχία ενός εξοπλισμού.

Πηγή: [7]

1.2.3.1.1.3 Πρόβλεψη (prediction):

Αναφέρεται στην έξοδο ενός αλγορίθμου αφού έχει εκπαιδευτεί σε ένα ιστορικό σύνολο δεδομένων και εφαρμόζεται σε νέα δεδομένα και προβλέπει την πιθανότητα ενός συγκεκριμένου αποτελέσματος. Ο αλγόριθμος θα δημιουργήσει πιθανές τιμές για μια άγνωστη μεταβλητή για κάθε εγγραφή στα νέα δεδομένα, επιτρέποντας στον κατασκευαστή μοντέλων να προσδιορίσει ποια θα είναι πιθανότατα αυτή η τιμή. Οι προβλέψεις μοντέλου μηχανικής μάθησης επιτρέπουν στις επιχειρήσεις να κάνουν πολύ ακριβείς εικασίες σχετικά με τα πιθανά αποτελέσματα μιας ερώτησης βάσει ιστορικών δεδομένων, τα οποία μπορεί να αφορούν όλα τα είδη πραγμάτων - πιθανότητα απώλειας πελατών, πιθανή δόλια δραστηριότητα και πολλά άλλα.

Πηγή: [8]

1.2.3.1.2 Μη εποπτευόμενη μάθηση (unsupervised learning)

Χρησιμοποιείται σε δεδομένα που δεν έχουν ετικέτες από παλιές παρατηρήσεις. Με λίγα λόγια το σύστημα δεν γνωρίζει τη σωστή απάντηση. Ο αλγόριθμος πρέπει να καταλάβει τι εμφανίζεται και να εξερευνήσει τα δεδομένα ώστε να βρει κάποια δομή. Για παράδειγμα έστω ότι αναγνωρίζουμε σαν όχημα μόνο τα IX αυτοκίνητα υποθέτοντας ότι δεν έχουν εφευρεθεί ακόμα άλλα είδη οχήματος. Ξαφνικά μια μέρα βλέπουμε μπροστά μας ένα φορτηγό και καθώς δεν το έχουμε ξαναδεί δεν μας είναι γνώριμο. Σκεπτόμενοι όμως τα δεδομένα που γνωρίζουμε από το ήδη γνωστό σε εμάς IX αυτοκίνητο (πόρτες, παρμπρίζ,4 ρόδες) το αναγνωρίζουμε και αυτό σαν όχημα. Οι δημοφιλέστερες τεχνικές περιλαμβάνουν nearest-neighbor mapping, k-means clustering και αρκετοί άλλοι. Αυτοί οι αλγόριθμοι χρησιμοποιούνται συχνά για την σύσταση στοιχείων και τον εντοπισμό ακραίων δεδομένων (outliers) ή αλλιώς ανωμαλιών.

1.2.3.1.2.1 Συσταδοποίηση (Clustering):

Ας υποθέσουμε ότι δίνετε ένα σύνολο δεδομένων όπου κάθε παρατηρούμενο παράδειγμα έχει ένα σύνολο χαρακτηριστικών, αλλά δεν έχει ετικέτες (labels). Ένα από τα πιο απλά που μπορούμε να κάνουμε σε ένα σύνολο δεδομένων χωρίς ετικέτες είναι να βρούμε ομάδες δεδομένων στο σύνολό του dataset που είναι παρόμοιες μεταξύ τους - αυτό που ονομάζουμε clusters. Αυτό κάνει και ο K-means. Ένα σύμπλεγμα (cluster) αναφέρεται σε μια συλλογή σημείων δεδομένων που συγκεντρώνονται μαζί λόγω ορισμένων ομοιοτήτων. Ορίζουμε έναν αριθμό στόχου k, ο οποίος αναφέρεται στον αριθμό των κεντροειδών (centroids) που χρειάζεται το σύνολο των δεδομένων. Ένα κεντροειδές είναι η φανταστική ή πραγματική θέση που αντιπροσωπεύει το κέντρο του συμπλέγματος. Με άλλα λόγια είναι η ομαδοποίηση ενός συνόλου αντικειμένων με τέτοιο τρόπο ώστε τα αντικείμενα στην ίδια ομάδα (που ονομάζεται σύμπλεγμα) να είναι πιο παρόμοια (με κάποια έννοια) μεταξύ τους από αυτά σε άλλες ομάδες (clusters). Είναι μια κοινή τεχνική για ανάλυση στατιστικών δεδομένων, που χρησιμοποιείται σε πολλούς τομείς, όπως αναγνώριση προτύπων, ανάλυση εικόνας, ανάκτηση πληροφοριών, βιοπληροφορική, συμπίεση δεδομένων, γραφικά υπολογιστών και μηχανική μάθηση.

Πηγή: [9]

1.2.3.1.3 Ημι-εποπτευόμενη μάθηση (semisupervised learning)

Χρησιμοποιείται για τις ίδιες εφαρμογές με την εποπτευόμενη μάθηση. Ωστόσο, χρησιμοποιεί τόσο δεδομένα με ετικέτα (labeled) όσο και χωρίς (unlabeled) για εκπαίδευση. Συνήθως μια μικρή ποσότητα δεδομένων με ετικέτα με μεγάλη ποσότητα δεδομένων χωρίς ετικέτα. Αυτός ο τύπος μάθησης μπορεί να χρησιμοποιηθεί με μεθόδους όπως ταξινόμηση (classification), παλινδρόμηση (regression) και πρόβλεψη (prediction). Ένα απλό παράδειγμα τέτοιας μεθόδου μπορεί να είναι και η αναγνώριση προσώπου από web camera.

1.2.3.1.4 Μάθηση ενίσχυσης (Reinforcement learning)

Χρησιμοποιείται συχνά για ρομποτική, παιχνίδια και gps πλοήγηση. Με την μέθοδο αυτή, ο αλγόριθμος ανακαλύπτει μέσω δοκιμής και σφάλματος ποιες ενέργειες αποδίδουν καλύτερα (μεγαλύτερο reward). Αυτός ο τύπος μάθησης έχει τρία βασικά συστατικά: τον πράκτορα (agent, δηλ. τον υπεύθυνο λήψης αποφάσεων), το περιβάλλον (environment, δηλ. ό,τι αλληλοεπιδρά με τον πράκτορα) και ενέργειες (actions, δηλ. τι μπορεί να κάνει ο πράκτορας). Ο στόχος είναι ο πράκτορας να επιλέξει ενέργειες που μεγιστοποιούν το αναμενόμενο reward για ένα δεδομένο χρονικό διάστημα.

Πηγή: [5]

1.3 Χρονοσειρές

Χρονική σειρά ή χρονοσειρά είναι μια ακολουθία που λαμβάνεται σε διαδοχικά ισαπέχουσες χρονικές στιγμές. Είναι δηλαδή μια ακολουθία δεδομένων διακριτού χρόνου. Πιο συγκεκριμένα είναι μια συχνότητα από ζευγάρια, όπου το κάθε ζευγάρι αποτελείται από έναν δείκτη χρόνου (time index) και μία τιμή (value).

Πρόκειται για δεδομένα πάνω στη συμπεριφορά μιας ή πολλών μεταβλητών σε διαδοχικές χρονικές περιόδους.

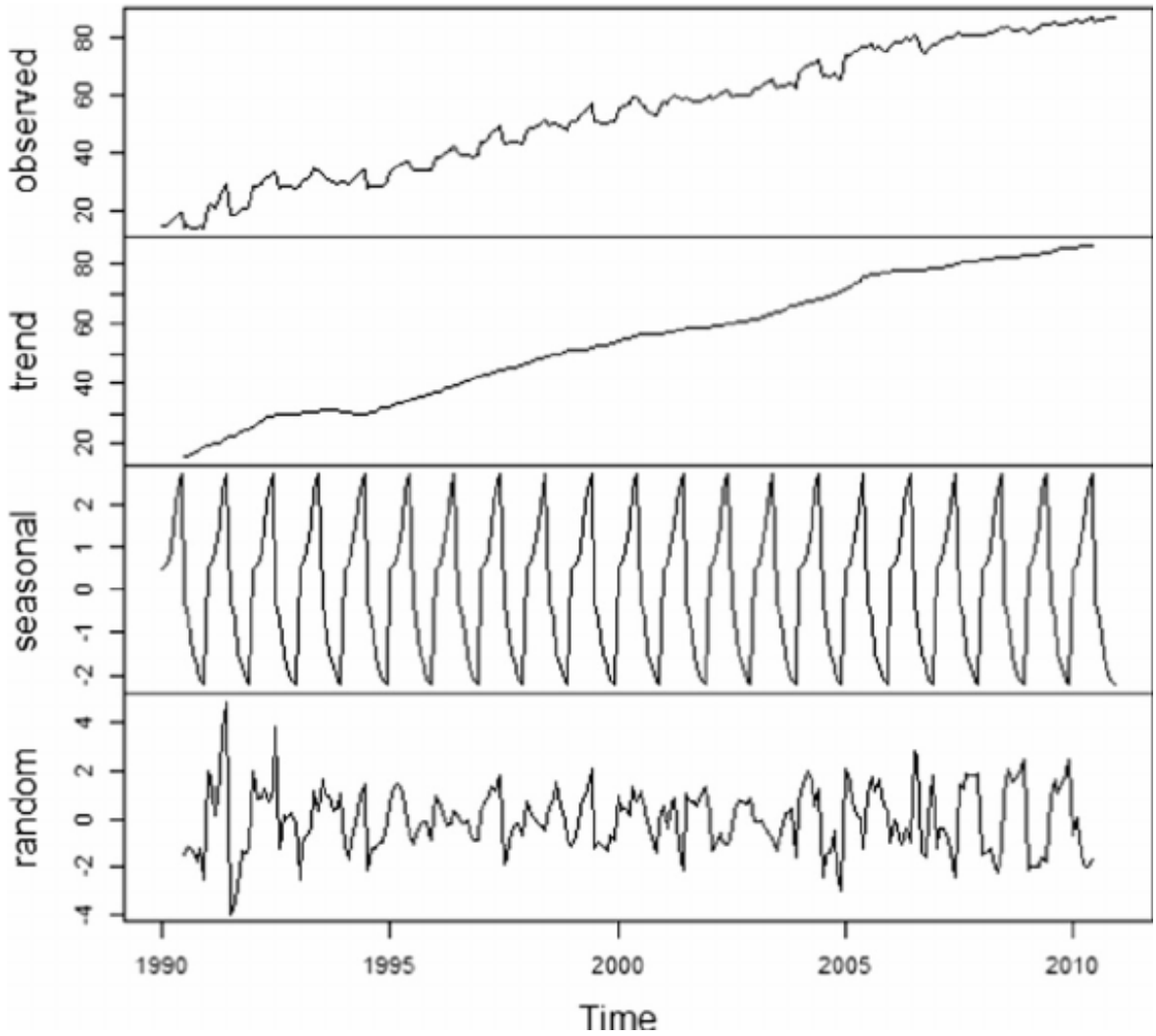
Οι χρονολογικές σειρές παρουσιάζονται πολύ συχνά μέσω γραφημάτων. Η συλλογή των ημερήσιων τιμών κλεισίματος του χρηματιστηρίου Αθηνών για ένα έτος αποτελεί μια χρονοσειρά. Σε μια χρονοσειρά, ο χρόνος είναι συχνά η ανεξάρτητη μεταβλητή και ο στόχος είναι συνήθως να κάνουμε μια πρόβλεψη για το μέλλον. Η στασιμότητα (**stationarity**) είναι ένα σημαντικό χαρακτηριστικό των χρονοσειρών. Μια χρονοσειρά λέγεται ότι είναι στατική εάν οι στατιστικές της ιδιότητες δεν αλλάζουν με την πάροδο του χρόνου. Με άλλα λόγια, έχει σταθερή μέση τιμή και διακύμανση. Εάν οι τιμές αυξάνονται προς το άπειρο, τότε δεν υπάρχει στασιμότητα.

Μερικά ακόμα βασικά χαρακτηριστικά των χρονοσειρών είναι:

- Η τάση (**trend**): Είναι η συστηματική μεταβολή (γραμμική ή μη) των τιμών ενός χαρακτηριστικού στη μονάδα του χρόνου. Δείχνει δηλαδή την διαχρονική κατεύθυνση της χρονοσειράς προς τα πάνω προς τα κάτω ή αν μένει σταθερή.
- Η περιοδικότητα (**periodicity**):Είναι ένα επαναλαμβανόμενο μοτίβο είτε υψηλών είτε χαμηλών τιμών ενός χαρακτηριστικού, το οποίο σχετίζεται με συγκεκριμένες χρονικές στιγμές (π.χ. κάθε φορά σε αργίες αυξάνονται οι τραπεζικές συναλλαγές μέσω διαδικτύου ή κάθε φορά τα Χριστούγεννα αυξάνονται οι πωλήσεις λόγω προφορών).
- Οι ακραίες τιμές (**outliers**):Είναι κάποιες τιμές, οι οποίες διαφέρουν σημαντικά από τη μέση τιμή των υπόλοιπων τιμών του. Αυτές οι ακραίες τιμές συνήθως επηρεάζουν αρνητικά κάποιο μοντέλο πρόβλεψης.

1.3.1 Ανάλυση χρονοσειρών

Η εικόνα παρακάτω βοηθά να γίνει πιο εύκολα αντιληπτή η έννοια των χαρακτηριστικών μιας χρονοσειράς.



Εικόνα 7: Time Series Analysis

Όπως βλέπουμε στην εικόνα η τάση (trend) μιας χρονοσειράς μπορεί να είναι σταθερή, ανοδική, η να μειώνεται. Στην εικόνα απεικονίζεται μια ανοδική τάση γιατί αυξάνεται σε σχέση με το χρόνο. Στα δεδομένα χρονοσειρών, η εποχικότητα είναι η παρουσία παραλλαγών που συμβαίνουν (συνήθως εντός ενός χρόνου η τριμήνου) και μπορεί να προκαλείται από διάφορους παράγοντες και αποτελείται από περιοδικά, επαναλαμβανόμενα και γενικά κανονικά μοτίβα. Στην περίπτωση της εικόνας παρατηρούμε μια εποχικότητα ανά χρόνο. Ένα παράδειγμα εποχικότητας αν μελετούσαμε μια χρονοσειρά με δεδομένα θερμοκρασιών μιας περιοχής σε ένα χρόνο, είναι οι υψηλές θερμοκρασίες που παρατηρούνται περισσότερο κατά την περίοδο του καλοκαιριού, ενώ τις υπόλοιπες περιόδους είναι πιο χαμηλές. Κάθε χρόνο οι θερμοκρασίες ανεβαίνουν το καλοκαίρι, άρα η χρονοσειρά θα έχει πιο υψηλές τιμές κατά την περίοδο εκείνη. Συνδυάζοντας και την τάση στο παραπάνω παράδειγμα θα έλεγε κανείς ότι είναι ανοδική, αφού παρατηρείται ότι όσο περνούν τα χρόνια, λόγω του φαινόμενου του θερμοκηπίου οι θερμοκρασίες του πλανήτη αυξάνονται. Τέλος στην περιοχή random φαίνεται μια χρονοσειρά η οποία είναι εντελώς τυχαία η μεταβολή των τιμών της στο χρόνο. Υπάρχουν διακυμάνσεις ακανόνιστες και δεν ακολουθούν κάποιο μοτίβο.

Σε κάποια προβλήματα ο χρόνος δειγματοληψίας μπορεί να μην είναι σταθερός και τότε χρειάζεται ειδικότερη επεξεργασία της χρονοσειράς για να γίνει η ανάλυση. Για παράδειγμα οι

ημερήσιες τιμές ενός χρηματιστηριακού δείκτη (π.χ. του Χρηματιστηρίου Αθηνών) συνιστούν μια χρονοσειρά με μεταβλητό φυσικό χρόνο δειγματοληψίας, αφού μεσολαβούν Σαββατοκύριακα και γιορτές που είναι κλειστό το χρηματιστήριο. Η πιο απλή προσέγγιση σε αυτήν την περίπτωση είναι να ορίσουμε ως χρόνο αναφοράς όχι το φυσικό χρόνο αλλά τον οικονομικό χρόνο συναλλαγών και να θεωρήσουμε σταθερό χρονικό βήμα μιας (οικονομικής) ημέρας ακόμα και από Παρασκευή σε Δευτέρα. Στη συνέχεια θα θεωρούμε το χρόνο δειγματοληψίας σταθερό που είναι και η τυπική περίπτωση στην ανάλυση μιας χρονοσειράς.

Η πρώτη υπόθεση που θα πρέπει να απορρίψουμε για να έχει νόημα η ανάλυση της χρονοσειράς είναι ότι η μεταβολή των τιμών του μεγέθους που παρατηρούμε είναι εντελώς τυχαία. Αν οι παρατηρήσεις της χρονοσειράς δεν είναι ανεξάρτητες, η πληροφορία που υπάρχει στη χρονοσειρά μπορεί να δίνεται με διαφορετικές μορφές και τα κυριότερα χαρακτηριστικά που θα πρέπει να μελετήσουμε πριν προχωρήσουμε να προσαρμόσουμε κάποιο μοντέλο στη χρονοσειρά είναι :

1.3.1.1 Στασιμότητα (Stationarity)

Αναλύοντας τη **στασιμότητα** (stationarity) μιας χρονοσειράς παρατηρούμε ότι μία μη στάσιμη χρονοσειρά μπορεί να έχει **τάσεις** (trends), δηλαδή αλλαγές στη μέση τιμή της με το χρόνο, π.χ. η τιμή βενζίνης μπορεί να έχει διακυμάνσεις λόγω της διεθνούς αγοράς αλλά και να παρουσιάζει μια αυξητική τάση σε βάθος χρόνου λόγω πληθωρισμού. Μια μη στάσιμη χρονοσειρά μπορεί επίσης να παρουσιάζει **περιοδικότητα** (periodicity), που όταν αναφέρεται σε συγκεκριμένες περιόδους που σχετίζονται με φυσικές εποχές του έτους (μήνα, τρίμηνο, κτλ.) λέγεται και **εποχικότητα** (seasonality) όπως αναφέρθηκε παραπάνω.

1.3.1.2 Αιτιοκρατία (determinism) και στοχαστικότητα (stochasticity)

Όλες οι χρονοσειρές από πραγματικά μεγέθη περιέχουν θόρυβο και με αυτήν την έννοια όλες οι πραγματικές χρονοσειρές είναι στοχαστικές. Αυτό που μετράει στην ανάλυση πραγματικών χρονοσειρών είναι η διερεύνηση και ο εντοπισμός του αιτιοκρατικού μέρους του συστήματος που παράγει τη χρονοσειρά. Αν δεν κυριαρχεί στην εξέλιξη της χρονοσειράς, τότε θεωρούμε πως το σύστημα είναι στοχαστικό.

1.3.1.3 Γραμμικότητα (linearity) και μη-γραμμικότητα (nonlinearity)

Η γραμμικότητα του συστήματος σημαίνει πως οι μεταβλητές του συστήματος αλληλοεπιδρούν γραμμικά, δηλαδή αν θα εκφράζαμε το σύστημα με αναλυτική μορφή όλοι οι όροι θα ήταν γραμμικοί ως προς τις μεταβλητές του συστήματος. Σε αντίθετη περίπτωση το σύστημα είναι μη γραμμικό.

1.4 Ανωμαλίες σε δεδομένα χρονικών σειρών

Σε μια χρονοσειρά και πιο συγκεκριμένα στη γραφική αναπαράσταση μιας χρονοσειράς μια απότομη ακίδα (spike) προς τα πάνω ή ένα απότομο βύθισμα, το οποίο ξεχωρίζει από την υπόλοιπη συμπεριφορά της γραφικής ορίζεται ως **ανώμαλη συμπεριφορά ή ανωμαλία.** Μαθηματικά, αυτό συμβαίνει όταν η τιμή που λαμβάνουν την συγκεκριμένη χρονική στιγμή τα δεδομένα είναι πολύ

μακριά από αυτά που προηγούνται. Υπάρχουν όμως πολλά είδη ανωμαλιών. Μερικά αναφέρονται παρακάτω.

Είδη ανωμαλιών.

Υπάρχουν πολλά είδη ανωμαλιών. Μερικά αναφέρονται παρακάτω:

1. **Point anomalies**. Όταν ένα data instance είναι πολύ μακριά από τα υπόλοιπα. Είτε πάνω είτε κάτω.
2. **Contextual anomalies**. Ανωμαλίες πολύ κοινές σε χρονοσειρές και έχουν να κάνουν με το γενικό πλαίσιο. Για παράδειγμα, είναι λογικό στις διακοπές να ξοδεύω 200 ευρώ την ημέρα, αλλά όχι και τις ημέρες εκτός διακοπών.
3. **Collective anomalies**. Παράδειγμα για την κατανόηση: κάποιος προσπαθεί να κάνει copy από ένα απομακρυσμένο μηχάνημα σε ένα τοπικό host. Αυτό θα φανεί σαν πιθανή κυβερνοεπίθεση και επομένως ανωμαλία.

1.4.1 Ανίχνευση και ανάλυση ανωμαλιών.

Γενικότερα είναι μία τεχνική που χρησιμοποιείται για να αναγνωρίσουμε ασυνήθιστα μοτίβα που δεν συμβαδίζουν με την αναμενόμενη συμπεριφορά. Ονομάζονται outliers, δηλαδή υπερβολικές-ακραίες τιμές.

Έχει πολλές εφαρμογές στην καθημερινότητα μας, όπως ανίχνευση ανωμαλιών στις κινήσεις τραπεζικών λογαριασμών (απάτη πιστωτικών καρτών), ασυνήθιστα μοτίβα στην κίνηση ενός δικτύου που μπορεί να αποτελεί σημάδι επίθεσης από hacker, ή ακόμα και στην ιατρική επιστήμη και συγκεκριμένα η ανάγνωση και η ανίχνευση, ενός όγκου σε μια μαγνητική ή αξονική τομογραφία.

Μερικές φορές όταν πρόκειται για ανάλυση χρονοσειρών και ανακάλυψη ανωμαλιών προκύπτουν κάποιες προκλήσεις, με τις οποίες πρέπει να εργαστούμε ώστε να έχουμε ένα καλό αποτέλεσμα. Για παράδειγμα σε πολλές περιπτώσεις τα όρια μεταξύ ανωμαλιών και φυσιολογικών δεδομένων δεν είναι ακριβή. Σε αυτήν την περίπτωση, οι κανονικές παρατηρήσεις θα μπορούσαν να θεωρηθούν ανωμαλίες και αντιστρόφως. Ακόμα πολλές φορές οι προσεγγίσεις για ανίχνευση ανωμαλιών σε ένα πεδίο πιο συχνά δεν μπορούν να χρησιμοποιηθούν σε ένα άλλο. Δηλαδή είναι πιθανό να μην μπορούμε να ακολουθήσουμε τον ίδιο τρόπο ανάλυσης και ανίχνευσης, όταν τα δεδομένα του ενός τρόπου είναι εντελώς διαφορετικά από τον τρόπο της μεθοδολογίας που ακολουθούμε, ή όταν η μέθοδος που έχουμε σαν πρότυπο είναι διαφορετική από αυτήν που θέλουμε να ακολουθήσουμε. Για παράδειγμα δεν μπορούμε να βασιστούμε σε μια στατιστική μέθοδο ανάλυσης αν επιθυμούμε να εργαστούμε με την μέθοδο της κατηγοριοποίησης, ή όταν έχουμε σκοπό η ανάλυση να γίνει με supervised machine learning δεν θα βοηθηθούμε αν επιλέξουμε να μελετήσουμε ένα πείραμα υλοποιημένο με unsupervised. Πολλά μπορεί να βοηθητικά αλλά δεν θα είναι κανόνας για την υλοποίησή μας.

Τέλος ένα πρόβλημα είναι η διαθεσιμότητα δεδομένων εκπαίδευσης και επικύρωσης για την εκπαίδευση ενός μοντέλου, γιατί μπορεί πλέον να υπάρχει ένας εξαιρετικά μεγάλος αριθμός data set στο διαδίκτυο, αλλά τις περισσότερες φορές θα χρειαστεί κάποια παραμετροποίηση ώστε να ικανοποιεί τις ανάγκες της λειτουργίας του δικού μας μοντέλου.

1.5 Η εργασία: Ανακάλυψη ανωμαλιών σε δεδομένα χρονοσειρών

Η διπλωματική αυτή πραγματεύεται την εύρεση ανωμαλιών σε χρονοσειρές με γνωστές ανωμαλίες με τη χρήση Data Mining αλγορίθμων. Έγινε εκτεταμένη μελέτη για την τεχνητή νοημοσύνη, την μηχανική μάθησης, τις χρονοσειρές , τις ανωμαλίες και την ανίχνευση ανωμαλιών σε χρονοσειρές. Ακόμα, έπειτα από έρευνα επιλέχθηκαν 5 αλγόριθμοι που χρησιμοποιούνται συχνά στην ανίχνευση ανωμαλιών σε χρονοσειρές και έγιναν πειράματα με τρία data set με γνωστές ανωμαλίες, με σκοπό την βελτίωση της λειτουργίας των αλγορίθμων και το καλύτερο δυνατό και πιο γρήγορο αποτέλεσμα. Οι αλγόριθμοι που χρησιμοποιήθηκαν είναι ο one class-support vector machine (OC-SVM), Isolation Forest, K-means, Z-score και η μέθοδος Symbolic Aggregate approximation (S.A.X.).

Στα πλαίσια της ανάλυσης αυτής ήταν και η δημιουργία μιας web εφαρμογής που ενσωματώνει όλους τους αλγόριθμους που επιλέχτηκαν και χρησιμοποιήθηκαν. Πιο συγκεκριμένα η εφαρμογή αναπτύχθηκε σε γλώσσα java (servlet με JSP) η οποία ενσωματώνει πολλά επιμέρους scripts υλοποιημένα σε γλώσσα python με τους αλγόριθμους και τις επιμέρους λειτουργίες τους. Η εφαρμογή υλοποιεί ουσιαστικά μία διεπαφή χρήστη που ενσωματώνει τα 3 βασικά dataset χρονοσειρών με γνωστές ανωμαλίες, ο χρήστης έχει τη δυνατότητα να δει πληροφορίες για την κάθε χρονοσειρά, να επιλέξει να κάνει οπτικοποίηση (visualize), να δει δηλαδή τις τιμές πως μεταβάλλονται σε σχέση με το χρόνο. Ακόμα μπορεί ο χρήστης να διαλέξει τον αλγόριθμο που επιθυμεί (όποιον από τους 5 επιθυμεί) για να δει τις ανωμαλίες που ανιχνεύει σε μορφή διαγράμματος τιμές-χρόνος και να κατεβάσει στον υπολογιστή του ένα αρχείο csv με τις τιμές στις οποίες βρέθηκαν ανωμαλίες, καθώς και στη χρονική στιγμή που βρέθηκαν. Τέλος, ανάλογα με το ποσοστό λάθους του κάθε αλγορίθμου ο χρήστης έχει τη δυνατότητα να ενημερωθεί για το ποιος αλγόριθμος ήταν πιο αποδοτικός και πιο ακριβής στην επιλογή των ανωμαλιών από το σύνολο των δεδομένων.

Στο κομμάτι των πειραμάτων αφιερώθηκαν και οι περισσότερες ώρες. Αφορά έλεγχο της λειτουργίας των αλγορίθμων, tuning και γενικότερη παραμετροποίηση με σκοπό την καλύτερη λειτουργία των αλγορίθμων και της εφαρμογής και πιο εμπεριστατωμένη έρευνα και μελέτη όσον αφορά τα αποτελέσματα και τα συμπεράσματα.

1.6 Αναμενόμενα αποτελέσματα.

Ακολουθώντας προηγούμενες προσεγγίσεις που είναι implemented σε γλώσσα java, δημιουργήθηκε η εφαρμογή που αναφέρεται στην παρούσα εργασία και γράφτηκε σε γλώσσα python σε συνδυασμό με JSP. Τα επιθυμητά αποτελέσματα από τη λειτουργία της εφαρμογής είναι να μπορεί ένας χρήστης να χειριστεί εύκολα και γρήγορα από οποιοδήποτε μηχάνημα, μέσω οποιουδήποτε browser πέντε βασικούς αλγορίθμους για την ανίχνευση ανωμαλιών σε χρονοσειρές και να εξάγει αποτελέσματα προς μελέτη. Στο μεγαλύτερο μέρος τη λειτουργίας της εφαρμογής αυτό ήταν εφικτό, πέρα από ένα μόνο σημείο που δυστυχώς δεν μπόρεσε η προσέγγιση αυτή να φτάσει σε χρόνο την προηγούμενη προσέγγιση.

1.7 Δομή της διπλωματικής

Στο Κεφάλαιο αυτό (1) γίνεται αναφορά στις εισαγωγικές έννοιες της εργασίας, αναλύονται οι βασικοί ορισμοί της χρονοσειράς, των ανωμαλιών, της ανάλυσης δεδομένων και της ανακάλυψης ανωμαλιών σε δεδομένα χρονοσειρών. Ακόμα γίνεται εκτενής αναφορά στην Τεχνητή νοημοσύνη, την μηχανική μάθηση και όλους τους τομείς και υπο-τομείς της. Εργασίες σχετικές με το αντικείμενο της διπλωματικής παρουσιάζονται στο Κεφάλαιο 2, όπου αναλύονται οι αλγόριθμοι και η λειτουργία τους λεπτομερώς. Κάθε αλγόριθμος και κάθε μέθοδος ανακάλυψης ανωμαλιών που χρησιμοποιείται στην εφαρμογή αναλύεται και θεωρητικά και πρακτικά και παραθέτονται διαγράμματα και εικόνες από την λειτουργία των μοντέλων. Στο κεφάλαιο 3 γίνεται περιγραφή της εφαρμογής, των τεχνολογιών που χρησιμοποιήθηκαν και των γλωσσών που προγραμματιστικά γράφτηκε η εφαρμογή. Ακόμα γίνεται παρουσίαση της λειτουργίας της μέσω παραδειγμάτων και εικόνων. Το κεφάλαιο 4 αφορά τα πειράματα που έγιναν για κάθε ένα data set ξεχωριστά και για κάθε έναν από τους πέντε αλγορίθμους. Γίνεται επεξήγηση με φωτογραφίες και εξηγείται γιατί επιλέχτηκαν οι συγκεκριμένες παραμετροποιήσεις σε κάθε αλγόριθμο. Το Κεφάλαιο 5 συζητά Συμπεράσματα και προτάσεις για μελλοντικές έρευνες ή εργασίες. Στο Κεφάλαιο 6 αποτυπώνεται η Βιβλιογραφία, ενώ στο Κεφάλαιο 7 οι πηγές που χρησιμοποιήθηκαν. Ακόμα όπως αναφέρθηκε και στον πρόλογο η εργασία αποτελείται από πέντε βασικά στάδια.

Το πρώτο από αυτά περιλάμβανε τη συλλογή και μελέτη στοιχείων, ερευνών για την ανακάλυψη ανωμαλιών σε δεδομένα χρονοσειρών. Σκοπός του βήματος αυτού ήταν η εξοικείωση με τεχνολογίες που αφορούν τον τομέα, με δεδομένα που μεταβάλλονται στον χρόνο και με την σωστή διαχείρισή τους.

Το δεύτερο περιλάμβανε συλλογή και μελέτη στοιχείων για το ποιοι αλγόριθμοι υπάρχουν για την ανακάλυψη ανωμαλιών σε χρονοσειρές, επιλογή πέντε (5) από τους βασικότερους και ανάλυση των βασικών τους σημείων. -Τι τους κάνει κατάλληλους για αυτό το task;-

Το τρίτο βήμα είχε στόχο την έρευνα και συλλογή, όσο το δυνατό περισσότερο, ολοκληρωμένου προγραμματιστικού κώδικα σχετικού με τους πέντε αλγόριθμους/μεθόδους που επιλέχθηκαν από το προηγούμενο βήμα. Επιπλέον συλλέχθηκε περεταίρω υλικό που βοήθησε στην αρχική σύσταση του προγραμματιστικού μέρους της διπλωματικής εργασίας. Μετά από έρευνα οι αλγόριθμοι που πληρούσαν τις προδιαγραφές και τελικά επιλέχθηκαν ως οι πιο κατάλληλοι για την

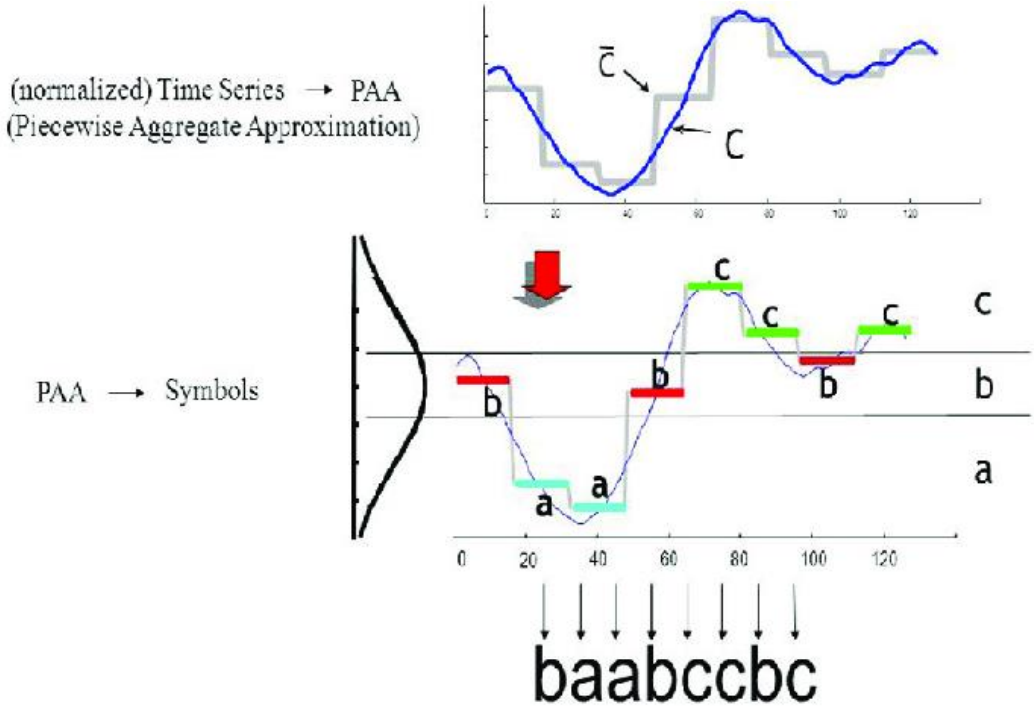
εφαρμογή ήταν οι εξής: **one class-support vector machine (OC-SVM), Isolation Forest, K-means, Z-score και η μέθοδος Symbolic Aggregate approximation (S.A.X.).**

Το τέταρτο βήμα αφορούσε τη δημιουργία μιας εφαρμογής που ενσωματώνει όλους τους αλγόριθμους. Πιο συγκεκριμένα αναπτύχθηκε μια web εφαρμογή που ενσωματώνει τους αλγόριθμους και τις επιμέρους λειτουργίες τους. Η εφαρμογή υλοποιεί ουσιαστικά μία διεπαφή χρήστη που ενσωματώνει 3 βασικά dataset χρονοσειρών με γνωστές ανωμαλίες.

Το πέμπτο και τελευταίο βήμα της διπλωματικής αφορούσε πειράματα με τις χρονοσειρές που έχουν επιλεγεί για την λειτουργία της εφαρμογής με κατάλληλες παραμετροποιήσεις και ελέγχους σε κάθε αλγόριθμο ξεχωριστά για να βγει ένα συμπέρασμα, ως προς το ποιος αλγόριθμος ανταπεξέρχεται καλύτερα με τα δεδομένα που του δίνονται και συνεπώς αν δίνει το καλύτερο δυνατό αποτέλεσμα σε σχέση με την ανίχνευση των ανωμαλιών.

2 *Μέθοδοι ανάλυσης ανωμαλιών σε χρονοσειρές.*

2.1 Symbolic Aggregate approximation (SAX)



Εικόνα 8: Symbolic Aggregate approXimation

Μέθοδος της συμβολικής συγκεντρωτικής προσέγγισης. Οι χρονοσειρές μπορούν να κατατμηθούν σε παράθυρα που αντιπροσωπεύουν τη χρονοσειρά μεταξύ 2 χρονικών δεικτών. Έτσι η μέθοδος **SAX** είναι μία τεχνική που μειώνει ένα παράθυρο μιας χρονοσειράς σε ένα σύμβολο. Τα σύμβολα αντιπροσωπεύουν παράθυρα.

Τι σημαίνει αυτό; Επειδή τα σύμβολα σε έναν πεπερασμένο χώρο συμβόλων έχουν μια πιθανότητα, μπορούμε να σκεφτούμε έτσι την πιθανότητα μιας χρονοσειράς. Επιπλέον τα σύμβολα έχουν το πλεονέκτημα ότι μπορούν να αποθηκεύονται και να χειρίζονται εύκολα, καθώς επίσης και να αναπαρασταθούν ως ακέραιος αριθμός.

Η μέθοδος **SAX** επιτρέπει τη μείωση των διαστάσεων και το indexing, με μία μέτρηση απόστασης (**distance measure**) κατώτατου ορίου (**lower-bounding**).

Τα βασικότερα πλεονεκτήματα της μεθόδου.

- Ευκλείδεια απόσταση χαμηλότερης οριοθέτησης.
- Μείωση των διαστάσεων.
- Μείωση των πολλών αριθμών.

Πηγές: [10] [11] [12] [13] [14] [15]

2.1.1 Βήματα της μεθόδου SAX.

Πριν από όλα τα βήματα πρέπει να γίνει κανονικοποίηση της χρονοσειράς (**Z-Normalization**), ώστε να έχει μέση τιμή **0** -μηδέν- και τυπική απόκλιση **1** -ένα-. **Z-Normalization**. Είναι ένα ουσιαστικό/σημαντικό βήμα της επεξεργασίας που επιτρέπει σε έναν αλγόριθμο να επικεντρωθεί στις ομοιότητες/ανωμαλίες και όχι στο εύρος.

Αφού η χρονοσειρά είναι κανονικοποιημένη ακολουθεί η αναπαράσταση της χρονοσειράς με μερικώς συγκεντρωτική προσέγγιση(**PAA-Piecewise Aggregate Approximation**). Ουσιαστικά το σύνολο των δεδομένων (**data set**) μήκους **N**, διαιρείται σε ίσα διαχωρισμένα τμήματα **W** και υπολογίζεται ο μέσος όρος κάθε τμήματος. Απλούστερα, μειώνεται ο αριθμός των διαστάσεων από **N** σε **W** και κάθε τμήμα **W** αντιπροσωπεύει πολλά διαδοχικά σημεία δεδομένων της χρονοσειράς.

Έπειτα, η μέθοδος λαμβάνει την παράσταση **PAA** ως είσοδο (**input**) και την διακρίνει σε ένα σετ από **k** αλφάβητα, έτσι ώστε τυπικά να ισχύει **k**<<**N**, όπου **N** το αρχικό μέγεθος της χρονοσειράς.

Υποθέτοντας ότι η κανονικοποιημένη χρονοσειρά έχει -φυσιολογικά- μια **Gaussian κατανομή**, στη συνέχεια καθορίζονται τα λεγόμενα σημεία διακοπής (**breakpoints**) που θα παράγουν **k** ισόποσες περιοχές κάτω από την τυπική κανονική καμπύλη.

Διευκρίνηση: Κάτω από το χαμηλότερο **breakpoint** η αναπαράσταση γίνεται με το γράμμα **a**. Οι συντελεστές που είναι μεγαλύτεροι ή ίσοι με το χαμηλότερο **breakpoint** αναπαρίστανται με το γράμμα **b** και συνεχίζει με τον ίδιο τρόπο ο υπολογισμός.

2.1.2 Η μέθοδος μέσα από τη βιβλιοθήκη saxpy.

Το SAX χρησιμοποιείται για να μετασχηματίσει μια ακολουθία λογικών αριθμών (δηλαδή, μια χρονοσειρά) σε μια ακολουθία γραμμάτων (δηλαδή, μια συμβολοσειρά) που είναι (συνήθως αλλά όχι πάντα) πολύ μικρότερη από τη χρονολογική σειρά εισαγωγής. Είναι απλούστερα μια μέθοδος διακριτοποίησης χρονοσειρών, για την καλύτερη κατανόηση των μοτίβων. Δεδομένου ότι η διακριτοποίηση είναι ίσως ο πιο χρησιμοποιούμενος μετασχηματισμός στην εξόρυξη δεδομένων, το SAX έχει χρησιμοποιηθεί ευρέως.

Υπάρχουν 3 πακέτα Python που έχουν υλοποιήσεις SAX και PAA - pyts, saxpy και tslearn. Το saxpy για αυτόν τον σκοπό. φαινόταν λίγο πιο απλό. Εάν έχετε τις τιμές σε μια σειρά τύπου numpy, μπορείτε να χρησιμοποιήσετε αυτήν την προσέγγιση για μετατροπή των χρονοσειρών ως λέξη 4 γραμμάτων χρησιμοποιώντας τα γράμματα a, b και c για παράδειγμα. [32]

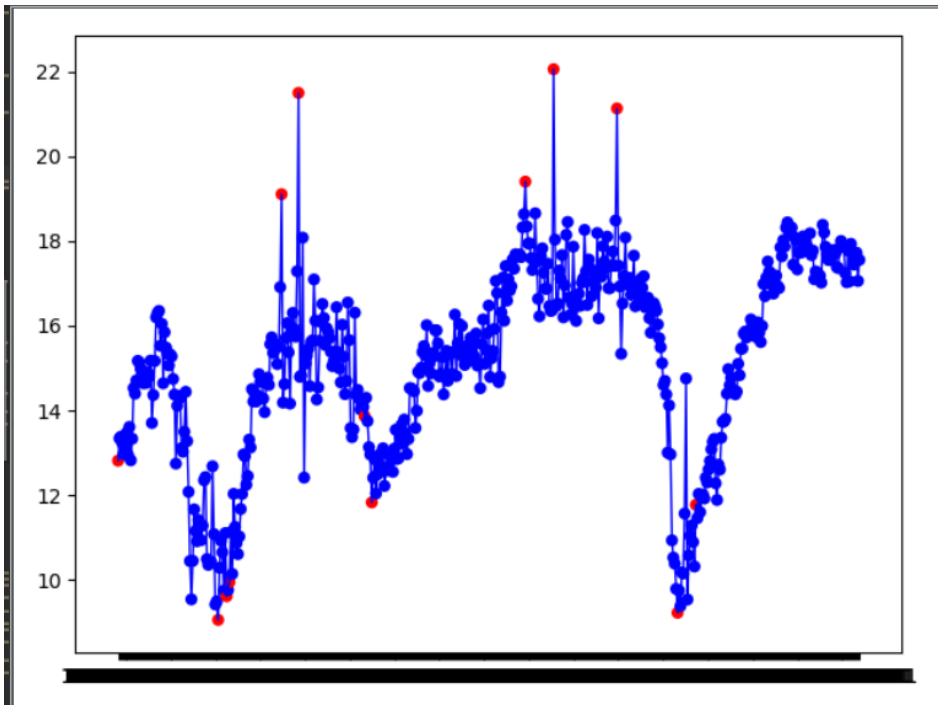
2.1.3 Η μέθοδος S.A.X. στην πράξη.

Η εφαρμογή υλοποιεί μια συνάρτηση που βρίσκει ανωμαλίες με τη μέθοδο S.A.X.

```
def find_discords_brute_force(series, win_size = 1, num_discords=12,
                              z_threshold=0.01):
```

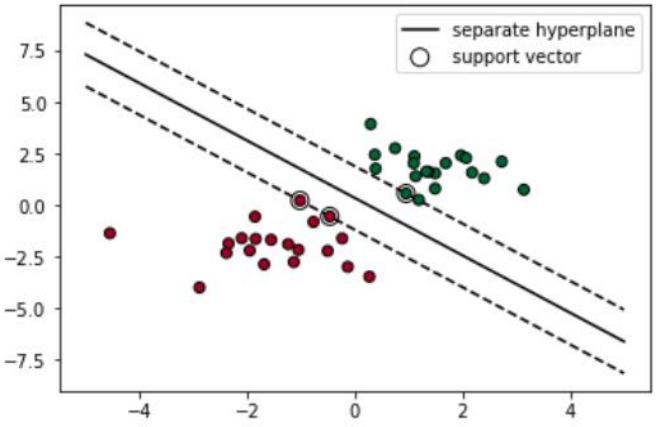
Εικόνα 9: Η συνάρτηση εύρεσης ανωμαλιών της μεθόδου S.A.X.

Εδώ βλέπουμε ότι το μέγεθος παραθύρου είναι 1 (win_size=1). Num_discords=12 που σημαίνει ότι αναμένονται να βρεθούν περίπου 12 ανωμαλίες από το σύνολο του dataset. Τέλος z_threshold=0.01 είναι ένα «κατώφλι» για την κανονικοποίηση. Στη συνέχεια γίνεται οπτικοποίηση των ανωμαλιών.



Εικόνα 10: Γραφική απεικόνιση της χρονοσειράς και των ανωμαλιών (S.A.X.)

2.2 One Class Support Vector Machines (OC-SVM)



Εικόνα 11: One Class SVM

Παρομοιάζοντας το support vector machine ως “road machine”, δηλαδή ως «μηχανή δρόμου», η οποία χωρίζει τα αριστερά από τα δεξιά αυτοκίνητα, τα κτίρια, τους πεζούς και καθιστά την ευρύτερη δυνατή λωρίδα, τότε θα ήταν πολύ πιο εύκολο να το καταλάβουμε. Αυτά τα αυτοκίνητα, τα κτίρια, πολύ κοντά στο δρόμο είναι τα support vectors.

Αν τώρα αντικαταστήσουμε όλα αυτά τα κτήρια, τα αυτοκίνητα και τους πεζούς με κουκίδες, το αποτέλεσμα είναι το εξής. Οι κόκκινες κουκίδες είναι για αντικείμενα στην αριστερή πλευρά του δρόμου. Οι πράσινες κουκκίδες είναι για αυτά στα δεξιά. Ο δρόμος γίνεται διακεκομμένες και συνεχείς γραμμές.

Τα περιθώρια είναι οι (κάθετες) αποστάσεις μεταξύ της γραμμής και των κουκκίδων που είναι πιο κοντά στη γραμμή.

Πηγές: [16] [17] [18] [19] [20]

2.2.1 SVM σε γραμμικές διαχωρίσιμες περιπτώσεις.

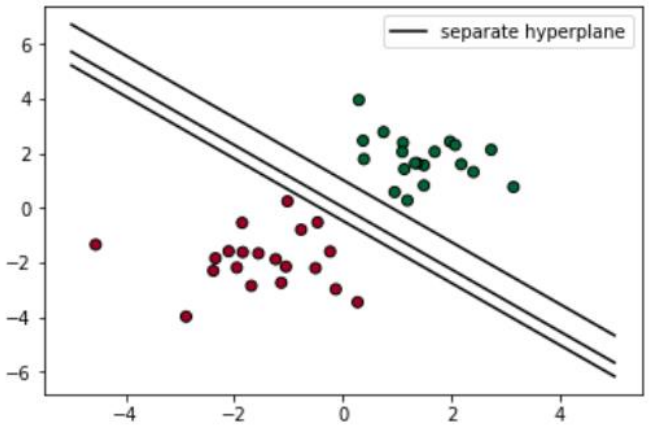
Προφανώς υπάρχουν άπειρες γραμμές για να διαχωριστούν οι κόκκινες και οι πράσινες κουκίδες στο παραπάνω παράδειγμα. Το SVM πρέπει να βρει τη βέλτιστη γραμμή με τον περιορισμό της σωστής ταξινόμησης ανά κλάση:

- Περιορισμός: εξετάζονται μόνο τα ξεχωριστά υπερεπίπεδα (π.χ. γραμμές διαχωρισμού), υπερεπίπεδα που ταξινομούν σωστά τις κλάσεις.
- Βελτιστοποίηση: επιλέγεται εκείνο που μεγιστοποιεί το περιθώριο.

Διευκρινήσεις:

Υπερεπίπεδο (hyperplane) είναι ένας (n μείον 1) -διασταδιακή υποπεριοχή για έναν n-διάστατο χώρο. Για παράδειγμα για ένα χώρο 2 διαστάσεων, το υπερεπίπεδο του θα είναι μία διάσταση, που είναι μόνο μια γραμμή. Για ένα χώρο 3 διαστάσεων, το υπερεπίπεδο του θα είναι 2-διάστατο, το οποίο είναι ένα επίπεδο που τέμνει τον κύβο.

Υπερεπίπεδο διαχωρισμού: Υποθέτοντας ότι η ετικέτα y είναι είτε 1 (για πράσινο) ή -1 (για κόκκινο), όλες αυτές οι τρεις παρακάτω γραμμές διαχωρίζουν τα υπερεπίπεδα. Επειδή όλες μοιράζονται την ίδια ιδιοκτησία (τον ίδιο χώρο) - πάνω από τη γραμμή, είναι πράσινο. κάτω από τη γραμμή, είναι κόκκινο.



Εικόνα 12: SVM separate hyperplane

Περιθώριο (margin): Ας υποθέσουμε ότι έχουμε ένα υπερεπίπεδο (γραμμή X)

Υπολογίζοντας την κατακόρυφη απόσταση από όλες αυτές τις κουκκίδες στη γραμμή X, θα είναι πολλές διαφορετικές αποστάσεις. Από όλες αυτές, η μικρότερη απόσταση. Αυτό είναι το περιθώριο μας.

2.2.2 SVM σε γραμμικές μη διαχωρίσιμες περιπτώσεις.

Εδώ το SVM προσπαθεί να βρει το υπερεπίπεδο που μεγιστοποιεί το περιθώριο, με την προϋπόθεση ότι και οι δύο κλάσεις έχουν ταξινομηθεί σωστά. Αλλά στην πραγματικότητα, τα σύνολα δεδομένων (data sets) πιθανώς δεν είναι ποτέ γραμμικά διαχωρίσιμα, οπότε η 100% σωστά ταξινομημένη κατάσταση από ένα υπερεπίπεδο είναι αδύνατη.

Το SVM απευθύνεται σε μη γραμμικά διαχωρίσιμες περιπτώσεις, εισάγοντας δύο έννοιες: **Soft Margin** και **Kernel Tricks**.

Παράδειγμα. Αν προσθέσω μια κόκκινη κουκίδα στο πράσινο σύμπλεγμα, το σύνολο δεδομένων γίνεται πλέον γραμμικό και δεν είναι διαχωρίσιμο.

Δύο λύσεις σε αυτό το πρόβλημα:

1 **Soft Margin**: προσπαθούμε να βρούμε μια γραμμή για να διαχωρίσουμε, αλλά με ανοχή μίας ή μερικών λανθασμένων ταξινομημένων τελειών.
2 **Kernel Tricks:** προσπαθούμε να βρούμε ένα μη γραμμικό όριο απόφασης.

Διευκρινήσεις:

Soft Margin: Δύο τύποι λανθασμένης ταξινόμησης είναι ανεκτοί από το SVM σε αυτήν την περίπτωση:

1 Η τελεία να είναι στη λάθος πλευρά του ορίου απόφασης, αλλά στη σωστή πλευρά (στο περιθώριο).
2 Η κουκκίδα να είναι στη λανθασμένη πλευρά του ορίου λήψης αποφάσεων και στην λανθασμένη πλευρά του περιθωρίου.

Kernel Tricks: Αυτό που κάνει το Kernel Trick είναι ότι χρησιμοποιεί τα υπάρχοντα χαρακτηριστικά, εφαρμόζει κάποιους μετασχηματισμούς και δημιουργεί νέα χαρακτηριστικά. Αυτά τα νέα χαρακτηριστικά είναι το κλειδί για το SVM να βρει το όριο των μη γραμμικών αποφάσεων.

Ήδη Kernel.

Polynomial kernel: Δημιουργούνται νέα χαρακτηριστικά εφαρμόζοντας τον συνδυασμό πολυωνύμων όλων των υπαρχόντων χαρακτηριστικών.

Radial Basis Function (RBF) kernel: Δημιουργούνται νέα χαρακτηριστικά μετρώντας την απόσταση μεταξύ όλων των άλλων κουκίδων σε συγκεκριμένα κέντρα τελειών.

2.2.3 Sklearn's SVM

Από default με τα attributes που χρησιμοποιούνται πιο συχνά είναι(Τα υπόλοιπα παίρνουν default τιμές): `sklearn.svm.OneClassSVM(kernel='rbf', gamma='scale', nu=0.5)`

Τα **kernels** αναλύθηκαν παραπάνω.

Gamma: Η default τιμή είναι 'scale', που σημαίνει ότι χρησιμοποιεί 1 / (n_features * X.var()). Αν η τιμή είναι 'auto', χρησιμοποιεί 1 / n_features. Πέρα από αυτές τις τιμές μπορεί να δεχτεί και οποιαδήποτε τιμή float.

Nu: Το nu είναι ένα άνω όριο στο κλάσμα των σφαλμάτων εκπαίδευσης (training errors fraction) και ένα κατώτερο όριο του κλάσματος των φορέων υποστήριξης (support vectors fraction). Θα πρέπει να είναι στο διάστημα (0, 1). Από προεπιλογή θα πάρει τιμή 0,5. Ουσιαστικά είναι το ποσοστό που αναμένεται να είναι ανωμαλία.

2.2.4 SVM στην πράξη.

Στην περίπτωση της παρούσας εφαρμογής χρησιμοποιήθηκε RBF Kernel με outlier_fraction=0.05. Δηλαδή δίνουμε στην συνάρτηση να καταλάβει ότι ανωμαλίες αναμένονται να είναι το 5% περίπου του dataset.

```
outliers_fraction = 0.05

data = dataset[['Value']]
scaler = StandardScaler()
np_scaled = scaler.fit_transform(data)
data = pd.DataFrame(np_scaled)
# train oneclassSVM
model = OneClassSVM(nu=outliers_fraction, kernel='rbf', gamma=0.01)

model.fit(data)
```

Εικόνα 13: OneClassSVM function (model)

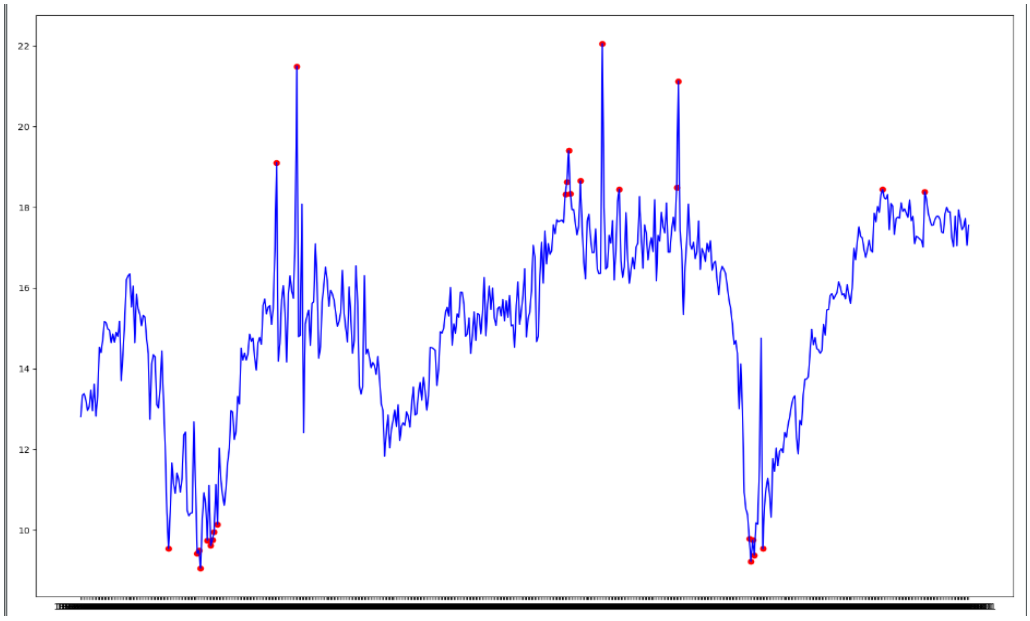
Όπως φαίνεται στην εικόνα περάσουμε τα δεδομένα που θα χρησιμοποιηθούν από Scaler για να επιτύχουμε Standardlization των δεδομένων. Αυτό πρέπει να γίνει γιατί εάν ένα χαρακτηριστικό έχει μια διακύμανση που είναι πολύ μεγαλύτερη από άλλες, μπορεί να κυριαρχήσει στην αντικειμενική συνάρτηση και να κάνει τον εκτιμητή (estimator) ανίκανο να μάθει από άλλα χαρακτηριστικά σωστά όπως αναμενόταν. Είναι απαραίτητο ειδικότερα όταν χρησιμοποιούμε RBF kernels.

Έπειτα αναθέτουμε ένα καινούριο column στο dataset όπου περιέχει τις ανωμαλίες, οι οποίες ανωμαλίες παράγονται σαν -1 από την συνάρτησή μας.

```
a = dataset.loc[dataset['anomaly'] == -1, ['Date', 'Value']] #anomaly
```

Εικόνα 14: anomaly declaration

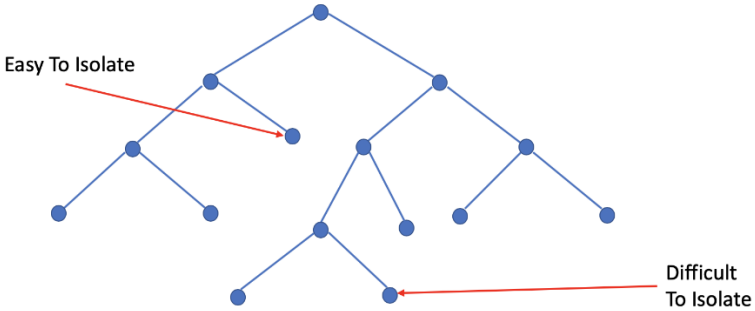
Τέλος κάνουμε visualize το αρχικό μας dataset και προσθέτουμε κόκκινες κουκίδες στα σημεία που βρέθηκε το -1. Το αποτέλεσμα βρίσκεται στην παρακάτω εικόνα.



Εικόνα 15: Γραφική απεικόνιση της χρονοσειράς και των ανωμαλιών (SVM.)

Το αποτέλεσμα μας δείχνει σχηματικά σε ποιο σημείο του dataset διαπιστώθηκε ανωμαλία.

2.3 Isolation Forest

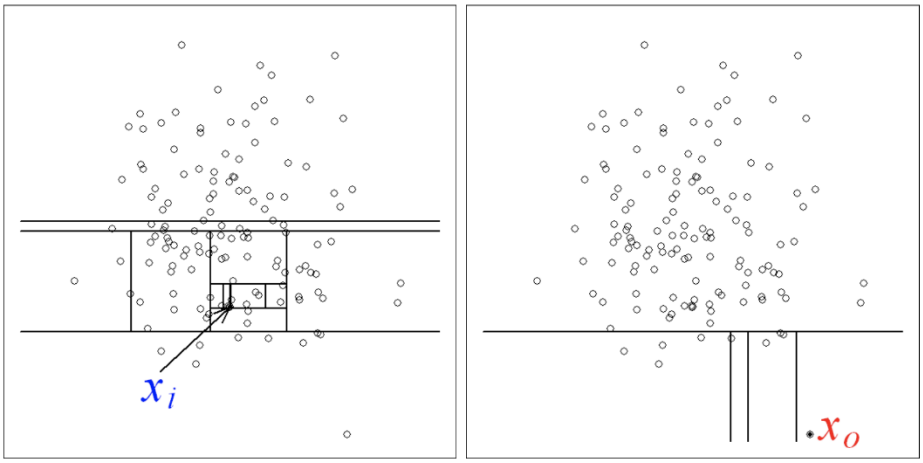


Εικόνα 16: Isolation Forest

Αναγνωρίζει ρητά τις ανωμαλίες αντί να κάνει profiling των κανονικών σημείων δεδομένων (data points). Ο αλγόριθμος αυτός όπως κάθε μέθοδος δένδρων, είναι χτισμένος με βάση τα δέντρα αποφάσεων (decision trees). Σε αυτά τα δέντρα, τα διαμερίσματα-κατατμήσεις (partitions) δημιουργούνται αρχικά επιλέγοντας τυχαία ένα χαρακτηριστικό και στη συνέχεια επιλέγοντας μια τυχαία τιμή διαχωρισμού μεταξύ της ελάχιστης και της μέγιστης τιμής του επιλεγμένου στοιχείου.

Οι αποκλίσεις (outliers) είναι λιγότερο συχνές από τις κανονικές παρατηρήσεις (regular observations) και είναι διαφορετικές από την άποψη των values (βρίσκονται πιο μακριά από τις regular observations στο χώρο των χαρακτηριστικών). Αυτός είναι ο λόγος για τον οποίο με τη χρήση αυτού του τυχαίου διαχωρισμού θα πρέπει να ταυτοποιηθούν πιο κοντά στη ρίζα του δέντρου με όσο το δυνατόν λιγότερους διαχωρισμούς (μικρότερο μέσο μήκος διαδρομής, δηλ., ο αριθμός των ακμών που πρέπει να περάσει μια παρατήρηση στο δέντρο που πηγαίνει από τη ρίζα στον τερματικό κόμβο).

Η ιδέα της αναγνώρισης μιας φυσιολογικής και μιας μη φυσιολογικής παρατήρησης φαίνεται στην εικόνα παρακάτω. Ένα κανονικό σημείο (στα αριστερά) απαιτεί την ταυτοποίηση περισσότερων κατατμήσεων από ένα μη φυσιολογικό σημείο (δεξιά).



Εικόνα 17: Κανονικό και μη κανονικό σημείο (Isolation Forest)

Πηγές: [21] [22] [23]

2.3.1 Anomaly score

Όπως συμβαίνει και με άλλες μεθόδους ανίχνευσης, απαιτείται μια βαθμολογία ανωμαλίας για τη λήψη αποφάσεων. Στην περίπτωση του Isolation Forest, ορίζεται ως:

$$s(x,n) = 2^{-\frac{E(h(x))}{c(n)}}$$

Εικόνα 18: Isolation Forest Anomaly Score

h(x): Είναι το μήκος της διαδρομής της παρατήρησης x.

c(n): Είναι το μέσο μήκος της διαδρομής της ανεπιτυχούς αναζήτησης σε ένα δυαδικό δέντρο αναζήτησης.

n: Είναι ο αριθμός των εξωτερικών κόμβων.

Το μήκος διαδρομής, υπολογιζόμενο κατά μέσον όρο σε ένα δάσος τυχαίων δέντρων, είναι ένα μέτρο της κανονικότητας και η decision function μας.

Σε κάθε παρατήρηση δίνεται ένα score ανωμαλίας και με βάση αυτό μπορεί να ληφθεί μια από τις ακόλουθες αποφάσεις:

- Ένα σκορ κοντά στο 1 δηλώνει ανωμαλίες

- Σκορ πολύ μικρότερο από 0,5 υποδεικνύει κανονικές παρατηρήσεις

- Εάν όλες οι βαθμολογίες είναι κοντά στο 0,5 τότε το σύνολο του δείγματος δεν φαίνεται να έχει σαφώς διακριτές ανωμαλίες.

2.3.2 Sklearn's Isolation Forest.

Από default με τα attributes που χρησιμοποιούνται πιο συχνά είναι(Τα υπόλοιπα παίρνουν default τιμές): IsolationForest(*self, n_estimators=100, max_samples='auto', contamination='auto', random_state=None*)

n_estimators: Ο αριθμός εκτιμητών (base estimators) στο σύνολο (default = 100).

max_samples: Ο αριθμός των δειγμάτων που θα αντληθούν από το X για το train κάθε estimator. Μπορεί να είναι int, float ή auto (default). Εάν τα max_samples είναι μεγαλύτερα από τον αριθμό των δειγμάτων που παρέχονται, όλα τα δείγματα θα χρησιμοποιηθούν για όλα τα δέντρα (χωρίς δειγματοληψία).

contamination: Η ποσότητα της «μόλυνσης» του συνόλου δεδομένων, δηλαδή το ποσοστό των ακραίων τιμών (outliers) στο σύνολο δεδομένων. Χρησιμοποιείται κατά την τοποθέτηση για τον καθορισμό του ορίου των σκορ των δειγμάτων. Εάν είναι "auto", το threshold καθορίζεται όπως στο original paper. Εάν είναι float η μόλυνση πρέπει να είναι στην περιοχή [0, 0.5]. Πρακτικά είναι το ποσοστό των ανωμαλιών που εμείς περιμένουμε ότι θα έχει το data set, το οποίο δεν μπορεί να ξεπεράσει το 50%.

random_state: Μπορεί να είναι int, RandomState instance και None. Αν είναι int, το random_state είναι το seed που χρησιμοποιείται από τη γεννήτρια τυχαίων αριθμών. Εάν είναι RandomState instance, τότε το random_state είναι η γεννήτρια τυχαίων αριθμών. Εάν είναι None η γεννήτρια τυχαίων αριθμών είναι RandomState instance που χρησιμοποιείται από το np.random.

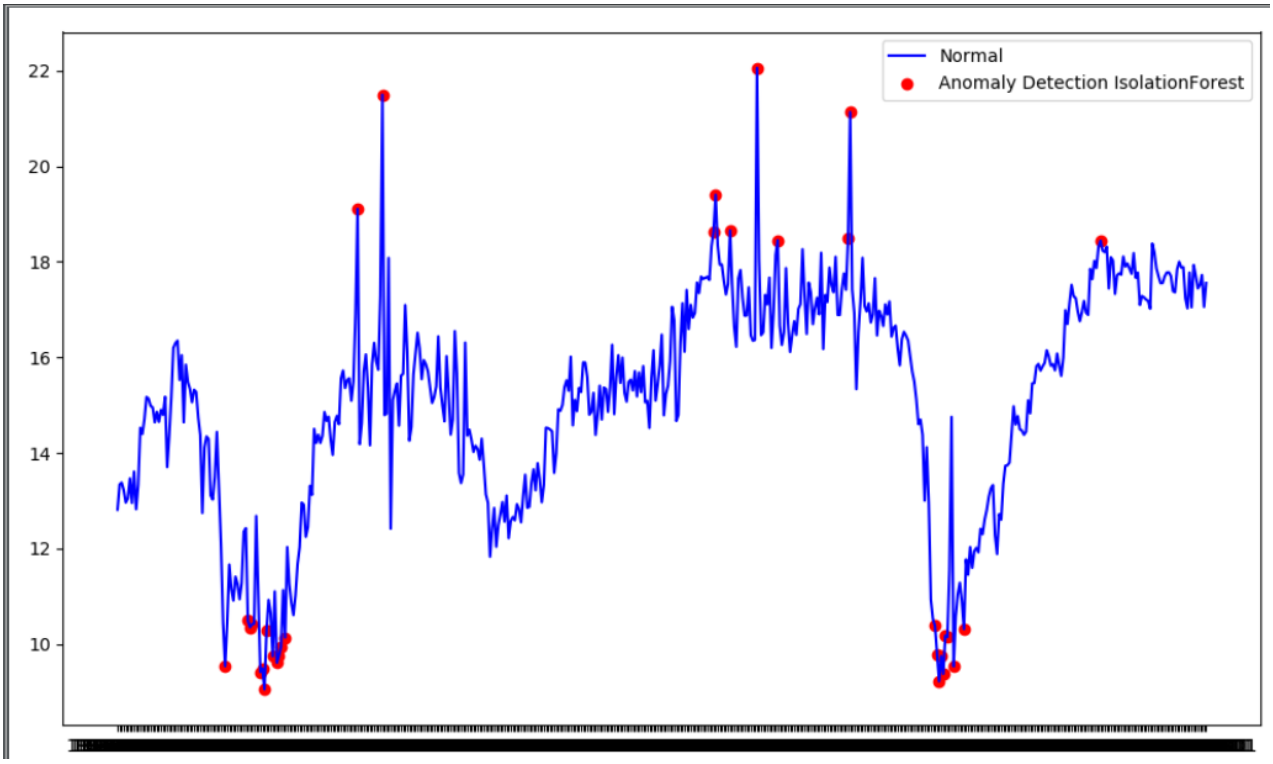
2.3.3 Isolation Forest στην πράξη.

Στην περίπτωση της παρούσας εφαρμογής χρησιμοποιήθηκε Behaviour=’new’, contamination ίσο με outlier_fraction=0.06. Δηλαδή δίνουμε στην συνάρτηση να καταλάβει ότι ανωμαλίες αναμένονται να είναι το 6% περίπου του dataset. Max_samples = 100 και random_state = 1.

```
data = dataset[['Value']]
outliers_fraction = 0.06
scaler = StandardScaler()
np_scaled = scaler.fit_transform(data)
data = pd.DataFrame(np_scaled)
# train isolation forest
model = IsolationForest(behaviour='new', max_samples=100, random_state=1, contamination=outliers_fraction)
model.fit(data)
dataset['anomaly2'] = pd.Series(model.predict(data))
```

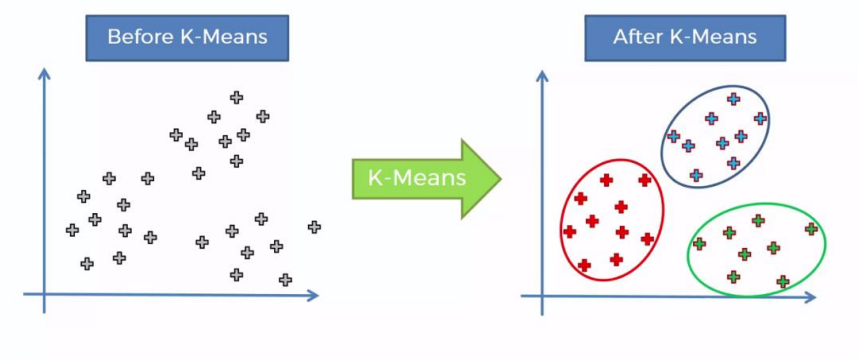
Εικόνα 19: IsolationForest function (model)

Στη συνέχεια αναθέτουμε ένα καινούριο column στο dataset όπου περιέχει τις ανωμαλίες και κάνουμε visualize το αρχικό μας dataset και προσθέτουμε κόκκινες κουκίδες στα σημεία που βρέθηκε ανωμαλία. Το αποτέλεσμα βρίσκεται στην παρακάτω εικόνα.



Εικόνα 20: Γραφική απεικόνιση της χρονοσειράς και των ανωμαλιών (Isolation Forest)

2.4 K-means.



Εικόνα 21: K-Means

Ο K-means είναι ένας από τους πιο απλούς και δημοφιλείς μη επιτηρούμενους (unsupervised) αλγόριθμους μηχανικής μάθησης.

Ο στόχος του αλγορίθμου είναι απλός: η ομαδοποίηση παρόμοιων δεδομένων συνδέει και ανακαλύπτει τα υποκείμενα μοτίβα. Για να επιτευχθεί αυτός ο στόχος, ο K-means αναζητά έναν καθορισμένο αριθμό (k) ομάδων σε ένα σύνολο δεδομένων.

Κάθε σημείο δεδομένων κατανέμεται σε κάθε ομάδα, μειώνοντας το άθροισμα των τετραγώνων. Με άλλα λόγια, ο αλγόριθμος K-means προσδιορίζει τον αριθμό k των κεντροειδών και στη συνέχεια κατανέμει κάθε σημείο δεδομένων στο πλησιέστερο σύμπλεγμα, ενώ διατηρεί τα κεντροειδή όσο το δυνατόν μικρότερα.

Με λίγα λόγια η λειτουργία του αλγόριθμου K Means είναι να ελαχιστοποιήσει την εσωτερική διακύμανση συμπλέγματος και να μεγιστοποιήσει τη μετατόπιση μεταξύ των ομάδων. Το K σημαίνει ότι η ομαδοποίηση αντιστοιχεί στον αριθμό των απαιτούμενων ομάδων. Το «μέσο» (means) αναφέρεται στο μέσο όρο των δεδομένων. δηλαδή, βρίσκοντας το κέντρο.

Πηγές: [24] [25] [26] [27]

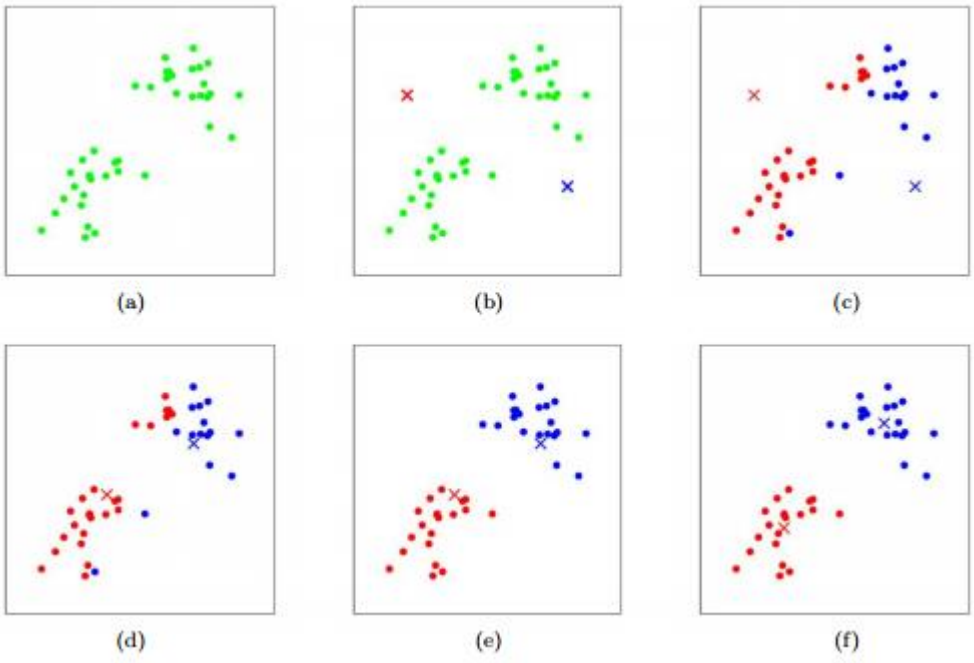
2.4.1 Πώς λειτουργεί ο K-means.

- **Βήμα 1:** Από τα σημεία που πρέπει να συγκεντρωθούν, ορίζονται τυχαία **n** σημεία ως το κέντρο του συμπλέγματος όπου **n** είναι ο αριθμός των απαιτούμενων ομάδων.
- **Βήμα 2:** Βρίσκεται η απόσταση μεταξύ όλων των σημείων με τα σημεία που επιλέγονται ως τυχαία κέντρα.
- **Βήμα 3:** Ορίζονται τα σημεία στο σύμπλεγμα στο οποίο είναι πιο κοντά.

Αυτό συμπληρώνει την 1η επανάληψη, τώρα έχουμε **n** συμπλέγματα (clusters) με τυχαία εκχωρημένα κέντρα.

- **Βήμα 4:** Επανατοποθέτηση των κέντρων. Αφού έχουμε ομάδες n μετά την 1η επανάληψη, πρέπει να υπολογίσουμε εκ νέου τα κέντρα και να εφαρμόσουμε ξανά τον αλγόριθμο.

2.4.1.1 Επεξήγηση.



Εικόνα 22: Step by step K-Means

Τα παραδείγματα εκπαίδευσης εμφανίζονται ως τελείες και τα κεντροειδή του συμπλέγματος εμφανίζονται ως σταυροί.

(a) Πρωτότυπο σύνολο δεδομένων.

(b) Τυχαία αρχικά κεντρομόρια συμπλέγματος.

(c-f) Εικονογράφηση της εκτέλεσης δύο επαναλήψεων του k-means.

Σε κάθε επανάληψη, εκχωρούμε κάθε παράδειγμα εκπαίδευσης στο πλησιέστερο κεντροειδές συμπλέγματος (που δείχνεται με το "ζωγραφίζοντας" τα παραδείγματα εκπαίδευσης το ίδιο χρώμα με το κεντροειδές του συμπλέγματος στο οποίο αποδίδεται). Τότε μετακινούμε κάθε κέντρο συγκέντρωσης στο μέσο όρο των σημείων που του έχουν δοθεί.

<u>Χ συντεταγμένη του κέντρου για ένα cluster</u> =

(Σύνολο χ συντεταγμένων σημείων στο σύμπλεγμα) / Αριθμός σημείων στο σύμπλεγμα

<u>Υ συντεταγμένη του κέντρου για ένα cluster</u> =

(Άθροισμα της συντεταγμένης y των σημείων στο σύμπλεγμα) / Αριθμός σημείων στο σύμπλεγμα

Χρησιμοποιώντας τα νέα σημεία που βρέθηκαν ως κεντρικά επαναλάβετε το **βήμα 2** και το **βήμα 3.** Ο αλγόριθμος σταματάει όταν τα σημεία στο σύμπλεγμα δεν αλλάξουν.

Τα νέα clusters που βρέθηκαν είναι τότε τα τελικά συμπλέγματα από τον αλγόριθμό.

2.4.2 Sklearn's K-means.

Από default με τα attributes που χρησιμοποιούνται πιο συχνά είναι (Τα υπόλοιπα παίρνουν default τιμές):

KMeans(*n_clusters=8, n_init=10, max_iter=300, precompute_distances='auto', random_state=None, algorithm='auto'*)

n_clusters: Ο αριθμός των ομάδων που σχηματίζονται καθώς και ο αριθμός των κεντροειδών που παράγουν. Από default είναι 8.

n_init: Οι φορές που ο αλγόριθμος θα τρέξει με διαφορετικούς σπόρους κεντροειδούς (centroid seeds). Default είναι 10.

max_iter: Μέγιστος αριθμός επαναλήψεων του αλγορίθμου για μία μόνο διαδρομή. Default είναι 300.

precompute_distances: Προϋπολογίζει τις αποστάσεις (πιο γρήγορο, αλλά χρειάζεται περισσότερη μνήμη). Μπορεί να πάρει :

auto: Μην κάνεις precompute τις αποστάσεις αν n_samples * n_clusters> 12 εκατομμύρια. Αυτό αντιστοιχεί σε περίπου 100MB overhead ανά εργασία με διπλή ακρίβεια.

True: Πάντα προϋπολογίζει τις αποστάσεις.

False: Ποτέ δεν προϋπολογίζει τις αποστάσεις.

Default είναι auto.

Random_state: Καθορίζει την παραγωγή τυχαίων αριθμών για την αρχικοποίηση του centroid. Default είναι None.

Algorithm: Επιλέγεται ο Αλγόριθμος K-means για χρήση. Ο κλασικός αλγόριθμος είναι "full". Η παραλλαγή "elkan: είναι πιο αποδοτική χρησιμοποιώντας την τριγωνική ανισότητα, αλλά προς το παρόν δεν υποστηρίζει αραιά δεδομένα. Η επιλογή "Auto" επιλέγει "elkan" για πυκνά δεδομένα και "full" για αραιά δεδομένα. Προφανώς η default τιμή είναι "auto"

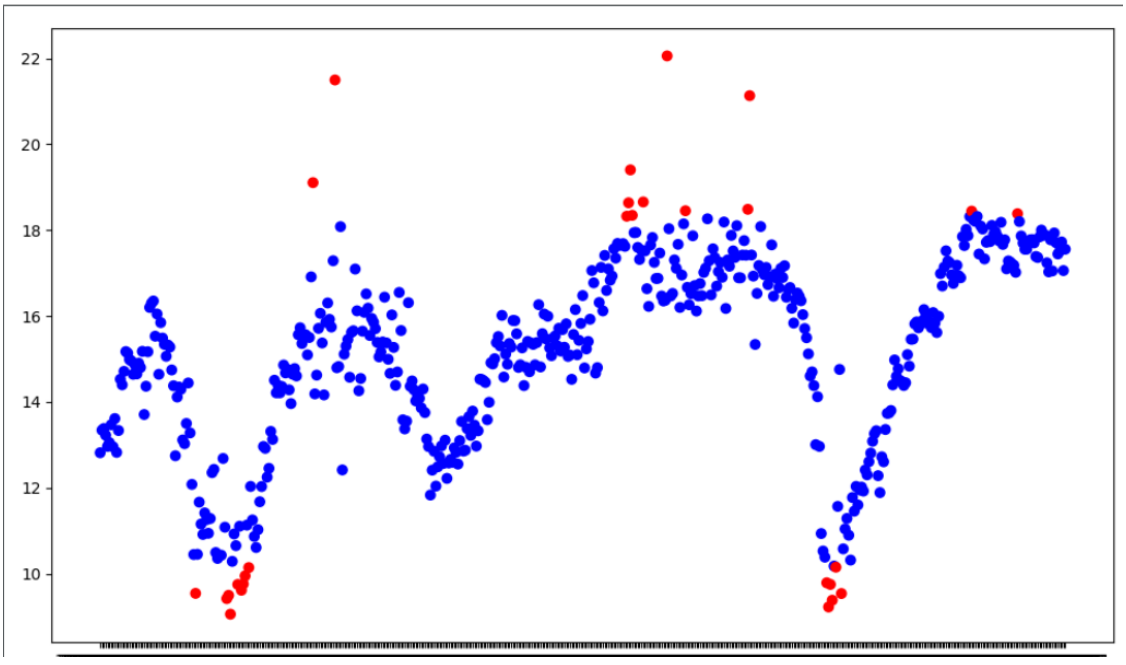
2.4.3 K-Means στην πράξη.

Στην περίπτωση της παρούσας εφαρμογής χρησιμοποιήθηκε αριθμός clusters n_clusters=10, και outlier_fraction=0.055. Δηλαδή δίνουμε στην συνάρτηση να καταλάβει ότι ανωμαλίες αναμένονται να είναι το 5.5% περίπου του dataset. Όλα τα υπόλοιπα είναι default.

```
data = df[['Value']]
n_cluster = range(1, 20)
kmeans = [KMeans(n_clusters=i).fit(data) for i in n_cluster]
scores = [kmeans[i].score(data) for i in range(len(kmeans))]
X = df[['Value']]
X = X.reset_index(drop=True)
km = KMeans(n_clusters=10)
km.fit(X)
km.predict(X)
labels = km.labels_

def getDistanceByPoint(data, model):
    distance = pd.Series()
    for i in range(0,len(data)):
        Xa = np.array(data.loc[i])
        Xb = model.cluster_centers_[model.labels_[i]-1]
        distance.set_value(i, np.linalg.norm(Xa-Xb))
    return distance

outliers_fraction = 0.055
```

Εικόνα 23: K-Means model (getDistnceByPoint function and KMeans function)

Έπειτα υπολογίζεται η απόσταση από το κάθε point ώστε να βρεθούν τα πολύ απομακρυσμένα στοιχεία. Τα οποία και πάλι κάνουμε κόκκινα στο αντίστοιχο διάγραμμα, όπως φαίνεται παρακάτω.



Εικόνα 24: Γραφική απεικόνιση της χρονοσειράς και των ανωμαλιών (KMeans)

2.5 Z-score.

Ένα Z-score μας δίνει μια ιδέα για το πόσο μακριά από το μέσο (mean) είναι ένα σημείο δεδομένων. Πιο τεχνικά, είναι ένα μέτρο για το πόσες τυπικές αποκλίσεις κάτω ή πάνω από το μέσο του «πληθυσμού» (Population) είναι ένα raw σκορ.

Ένα Z-score μπορεί να τοποθετηθεί σε μια καμπύλη κανονικής κατανομής. Κυμαίνονται από -3 τυπικές αποκλίσεις (οι οποίες θα βρεθούν στο άκρο της αριστερής πλευράς της καμπύλης κανονικής κατανομής) έως +3 τυπικές αποκλίσεις (οι οποίες θα βρεθούν στο άκρο της δεξιά πλευράς της καμπύλης κανονικής κατανομής). Για να χρησιμοποιήσουμε ένα z-σκορ, πρέπει να γνωρίζουμε τη μέση (μ) και επίσης την τυπική απόκλιση (σ) του πληθυσμού.

Πηγές: [28] [29] [30] [31]

Εικόνα 25: Z-Score

<u>Παράδειγμα:</u> Είναι ένας τρόπος σύγκρισης των αποτελεσμάτων με έναν «κανονικό» πληθυσμό. Τα αποτελέσματα από δοκιμές ή έρευνες έχουν χιλιάδες πιθανά αποτελέσματα και μονάδες. Αυτά τα αποτελέσματα συχνά φαίνονται άνευ σημασίας. Για παράδειγμα, γνωρίζοντας ότι το βάρος ενός ατόμου είναι 120 κιλά μπορεί να είναι καλή πληροφορία, αλλά αν θέλετε να το συγκρίνετε με το βάρος του «μέσου» ατόμου, κοιτάζοντας έναν τεράστιο πίνακα δεδομένων μπορεί να είναι υπερβολική (ειδικά αν κάποια βάρη καταγράφονται σε χιλιόγραμμα) . Ένα z-σκορ μπορεί να πει πού το βάρος του ατόμου αυτού συγκρίνεται με το μέσο βάρος του μέσου πληθυσμού.

2.5.1 Z-score formula: One sample.

Ο βασικός τύπος για ένα δείγμα είναι: $z = (x - \mu) / \sigma$

Ας υποθέσουμε ότι έχουμε test score 190. Το test έχει μέση τιμή (μ) 150 και τυπική απόκλιση (σ) 25. Εάν υποθέσουμε μια κανονική κατανομή, το Z-score θα είναι: **(190 – 150) / 25 = 1.6**

Αυτό σημαίνει 1,6 τυπικές αποκλίσεις πάνω από το μέσο όρο.

2.5.1.1 Επεξήγηση.

- Ένα Z-score με τιμή 1 είναι μία (1) τυπική απόκλιση πάνω από τον μέσο όρο.
- Ένα Z-score με τιμή 2 είναι δύο (2) τυπικές αποκλίσεις πάνω από το μέσο όρο.
- Ένα Z-score με τιμή -1,8 είναι -1,8 τυπικές αποκλίσεις κάτω από το μέσο όρο.
- Ένα μηδενικό (0) Z-score λέει ότι οι τιμές είναι ακριβώς οι μέσες, ενώ μια βαθμολογία +3 σας λέει ότι η τιμή είναι πολύ υψηλότερη από τον μέσο όρο.

Ένα z-score σας λέει πού βρίσκεται η βαθμολογία σε μια κανονική καμπύλη κατανομής.

2.5.2 Z-score formula: Standard error of the mean.

Όταν έχουμε πολλαπλά δείγματα και θέλουμε να περιγράψουμε την τυπική απόκλιση αυτών των μέσων δειγματοληψίας (**standard error**), Χρησιμοποιούμε τον τύπο: $z = (x - \mu) / (\sigma / \sqrt{n})$

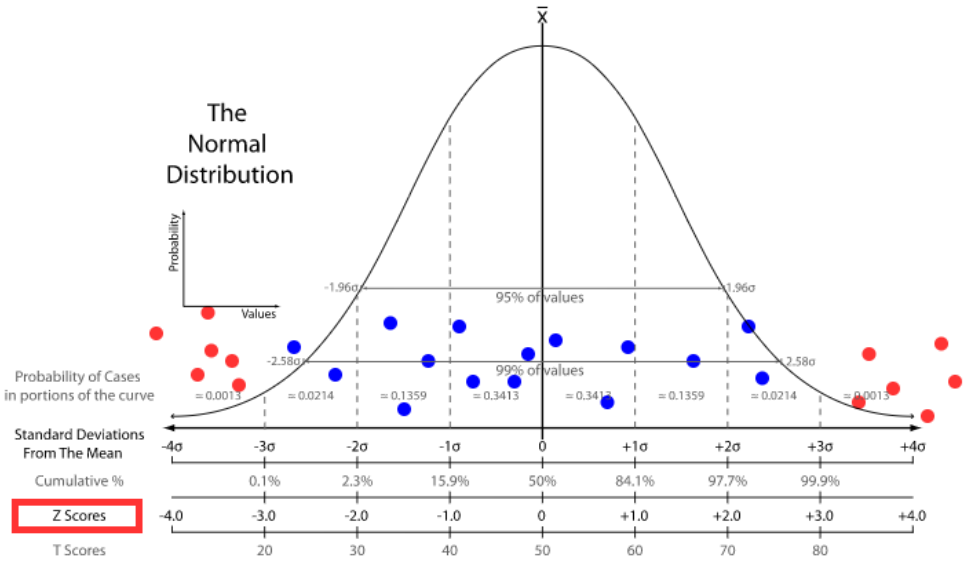
Αυτό το z-score βρίσκει πόσα τυπικά σφάλματα υπάρχουν μεταξύ του μέσου όρου δείγματος και του μέσου όρου του πληθυσμού.

Όταν έχουμε να κάνουμε με μια δειγματοληπτική κατανομή των μέσων, πρέπει να συμπεριλάβουμε το τυπικό σφάλμα στον τύπο.

2.5.3 Z-score for anomaly detection

Η υπόθεση της μεθόδου z-score στην ανίχνευση ανωμαλιών είναι ότι η τιμή των δεδομένων είναι σε Gaussian κατανομή και οι ανωμαλίες είναι τα σημεία δεδομένων μακριά από το μέσο όρο του πληθυσμού.

2.5.3.1 Παράδειγμα με σχήμα.



Εικόνα 26: Standard Deviations from the Mean explained

Οι μπλε κουκκίδες αντιπροσωπεύουν inliers, ενώ οι κόκκινες κουκίδες είναι οι ακραίες τιμές (outliers). Όσο μεγαλύτερος είναι ο αριθμός των τυπικών αποκλίσεων μακριά από τον μέσο όρο, τόσο περισσότερο είναι πιθανό να υπάρχει μια ανωμαλία σε αυτό το σημείο δεδομένων. Με άλλα λόγια, όσο πιο μακριά από το κέντρο, μεγαλύτερη η πιθανότητα να είναι ανωμαλία (outlier).

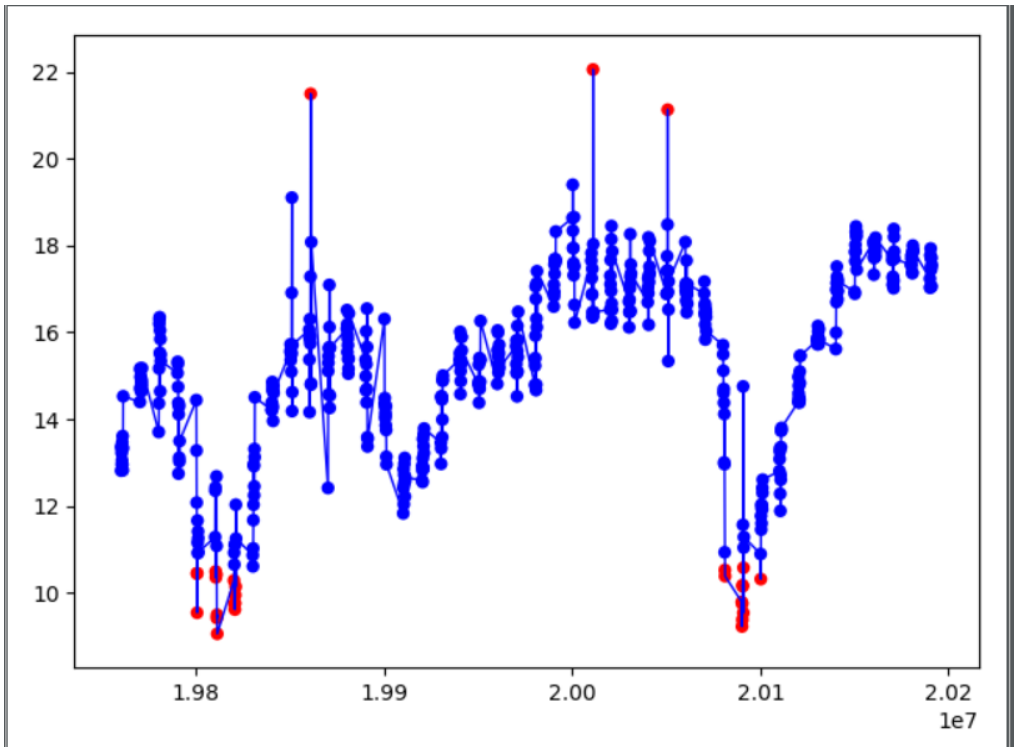
2.5.4 Z-score στην πράξη.

Στην εφαρμογή μας η συνάρτηση που αναζητά τις ανωμαλίες φαίνεται στην εικόνα παρακάτω και ακολουθεί τον τύπο που αναφέρεται παραπάνω, (test_score-μέση τιμή)/τυπική απόκλιση.

```
series = np.array(df.Value)
outliers=[]
df.Date = df.Date.apply(clean)
def detect_outlier(data_1):

    threshold=2
    mean_1 = np.mean(data_1)
    std_1 =np.std(data_1)


    for y in data_1:
        z_score= (y - mean_1)/std_1
        if np.abs(z_score) > threshold:
            outliers.append(y)

    return outliers
# print(series)
outlier_datapoints = detect_outlier(series)
```

Εικόνα 27: Z-Score outlier detection function

Έπειτα βάζω στο dataset μου κόκκινες κουκίδες στα σημεία που βρέθηκαν τα outliers (είναι αποθηκευμένα στον πίνακα outlier_datapoints). Το αποτέλεσμα της διαδικασίας οπτικοποίησης των ανωμαλιών που προκύπτει, φαίνεται στην εικόνα παρακάτω.



Εικόνα 28: Γραφική απεικόνιση της χρονοσειράς και των ανωμαλιών (Z-score)

3 *Η εφαρμογή*

Η εφαρμογή που υλοποιήθηκε στα πλαίσια της παρούσας διπλωματικής εργασίας εστιάζει στην αλληλεπίδραση του χρήστη με πέντε (5) διαφορετικούς αλγορίθμους, για την ανίχνευση ανωμαλιών σε δεδομένα χρονικών σειρών. Όπως αναφέρθηκε και σε προηγούμενο κεφάλαιο, η λειτουργία της εφαρμογής είναι αποτέλεσμα υλοποιημένου κώδικα σε γλώσσες Java και python 3.6 και ο server που φιλοξενεί localhost την εφαρμογή είναι ο Apache Tomcat 9.0.

3.1 Γιατί JSP

Πιο συγκεκριμένα το κομμάτι που είναι υπεύθυνο για την εμφάνιση του web app στον web browser είναι υλοποιημένο σε JSP (Java Server Pages), που σημαίνει ότι έχει γίνει και εκτενής χρήση HTML για κουμπιά, πίνακες, παραγράφους κτλ. Η χρήση JSP είναι μια καλή επιλογή, καθώς αντί να ενσωματώνει HTML στον κώδικα προγραμματισμού, επιτρέπει να ενσωματώσουμε εξειδικευμένο κώδικα (scripting code) σε σελίδες HTML. Η Java είναι η προεπιλεγμένη γλώσσα ενεργειών του JSP, αλλά επιτρέπει και άλλες γλώσσες, όπως JavaScript και Perl. Είναι σημαντικό να αναφερθεί ότι μια σελίδα JSP γίνεται compile πάντα πριν γίνει processed από τον server. Αυτό σημαίνει πως κάθε σελίδα JSP γίνεται compile σε εκτελέσιμο αρχείο την πρώτη φορά που αυτό θα ζητηθεί και κάνει invoke κατευθείαν το αποτέλεσμα σε επόμενα ζητήματα. Το αποτέλεσμα προφανώς είναι η ταχύτητα. Ακόμα μπορείς πολύ εύκολα να κρατάς κάποια entities ή attributes στο session για χρήση σε όλες τις σελίδες της εφαρμογής, πράγμα που στη συγκεκριμένη εφαρμογή είναι απαραίτητο.

Επιπροσθέτως, έχει γίνει χρήση Java για λειτουργίες όπως την διαδικασία όπου ο χρήστης μπορεί να κατεβάσει ένα αρχείο δεδομένων .csv από το server, και κάθε φορά που χρειάζεται να τρέξει ένα python script δημιουργείται μια process και καλείται η `getRuntime()` από το πακέτο `java.lang.Runtime` που κάνει retrieve το τρέχον Java Runtime Environment. Για να γίνει αντιληπτή η λειτουργία της μεθόδου αυτής, συχνά οι προγραμματιστές καλούν αυτή τη μέθοδο για να κάνουν launch έναν browser για την εμφάνιση μιας “help page” σε HTML. Πιο εκλαϊκευμένα, θα μπορούσε να πει κανείς ότι με αυτόν τον τρόπο τρέχω python μέσα σε java.

3.2 Γιατί python

Όσον αφορά το προγραμματιστικό κομμάτι με την γλώσσα python, πρέπει να γίνει αντιληπτό ότι επιλέχθηκε σαν γλώσσα και χρησιμοποιήθηκε λόγω των εργαλείων και των δυνατοτήτων που προσφέρει για ανάλυση. Προσφέρει τεράστια υπολογιστική γκάμα και δυνατότητα οπτικοποίησης όλων των εργασιών σε διαγράμματα. Η άλλη επιλογή ήταν να χρησιμοποιηθεί η γλώσσα R όμως για τους λόγους που θα αριθμήσω παρακάτω η python ήταν η καλύτερη δυνατή επιλογή.

Αρχικά έχει σχεδιαστεί για να είναι εύκολη και κατανοητή και ταυτόχρονα διασκεδαστική στη χρήση. Αφού μπορεί κάποιος να δημιουργήσει προγράμματα με πρωτότυπα εργαλεία και σχετικά γρήγορα, θεωρούμε ότι η κωδικοποίηση στην Python είναι μια πολύ καλή λύση. Ένας άλλος λόγος που η γλώσσα python ήταν η πλέον κατάλληλη γλώσσα για την υλοποίηση των πάρα πολλών υπολογισμών και διεργασιών που έπρεπε να γίνουν, είναι εύκολη στην κατανόηση. Για παράδειγμα, μπορεί να πει κάποιος ότι διαβάζεται σαν την αγγλική γλώσσα, πράγμα που για υψηλού επιπέδου γλώσσα προγραμματισμού είναι πάρα πολύ θετικό. Χειρίζεται δηλαδή πολλή πολυπλοκότητα για λογαριασμό του προγραμματιστή, οπότε είναι πολύ φιλική προς τους αρχάριους. Επιπλέον, η Python «συγχωρεί» τα λάθη, επομένως εξακολουθεί ο προγραμματιστής να είναι σε θέση να μεταγλωττίσει και να εκτελέσει το πρόγραμμά του έως ότου χτυπήσει το προβληματικό μέρος του κώδικα.

Ωστόσο η Python είναι μια δυναμική γλώσσα ως προς το γράψιμο, δηλαδή το ίδιο πράγμα μπορεί εύκολα να σημαίνει κάτι διαφορετικό ανάλογα με το γενικότερο πλαίσιο. Άρα τα σφάλματα είναι δύσκολο να εντοπιστούν και να διορθωθούν χωρίς εμπειρία. Τέλος, όπως είπαμε είναι μια dynamically typed γλώσσα, που σημαίνει ότι είναι αργή ακριβώς επειδή είναι ευέλικτη και χρειάζεται να κάνει πολλούς υπολογισμούς προκειμένου να ορίσει σωστά τον ορισμό για κάτι. Αυτός είναι και ο κύριος λόγος που η εφαρμογή αυτή δεν έπιασε σε χρόνο τις προηγούμενες προσεγγίσεις που έγιναν σε java.

3.2.1 Η χρήση της python στην εφαρμογή

Η χρήση της γλώσσας python στην εφαρμογή αυτή έγινε με σκοπό την υλοποίηση υπολογισμών και αναλύσεων καθώς και τη δημιουργία διαγραμμάτων. Η εφαρμογή απαρτίζεται από 37 διαφορετικά script γραμμένα σε python. Το κοινό χαρακτηριστικό στα περισσότερα από αυτά είναι ότι χρησιμοποιούνται οι βιβλιοθήκες NumPy και pandas που θα αναλυθούν παρακάτω. Πέρα από την χρήση όμως αυτών των πακέτων που χρησιμεύουν για array και data manipulation ξεχωριστά αναφέρονται ονομαστικά οι βιβλιοθήκες που χρησιμοποιήθηκαν κατά περίπτωση.

- ✓ Για τη χρήση του αλγορίθμου One Class SVM και των δυνατοτήτων του, έγινε import το πακέτο δεδομένων **OneClassSVM** από το γενικότερο πακέτο svm της βιβλιοθήκης sklearn. (Sklearn.svm)
- ✓ Για τη χρήση του αλγορίθμου Isolation Forest και των δυνατοτήτων του, έγινε import το πακέτο **IsolationForest** από το γενικότερο πακέτο ensemble της βιβλιοθήκης Sklearn. (sklearn.ensemble)
- ✓ Για τη χρήση του αλγορίθμου KMeans και των δυνατοτήτων του, έγινε import το πακέτο δεδομένων **KMeans** από το γενικότερο πακέτο cluster της βιβλιοθήκης Sklearn. (sklearn.cluster)

- ✓ Για τη λειτουργία της μεθόδου S.A.X. και των δυνατοτήτων της, έγιναν import τα πακέτα δεδομένων **VisitRegistry, early_abandoned_dist, znorm** από τα γενικότερα πακέτα visit_registry, distance, znorm της βιβλιοθήκης saxpy. (saxpy.visit_registry, saxpy.distance, saxpy.znorm)

Τέλος για την λειτουργία του mean squared logarithmic error, έγινε import το πακέτο **mean_squared_log_error** από το γενικότερο πακέτο metrics της βιβλιοθήκης Sklearn. (sklearn.metrics)

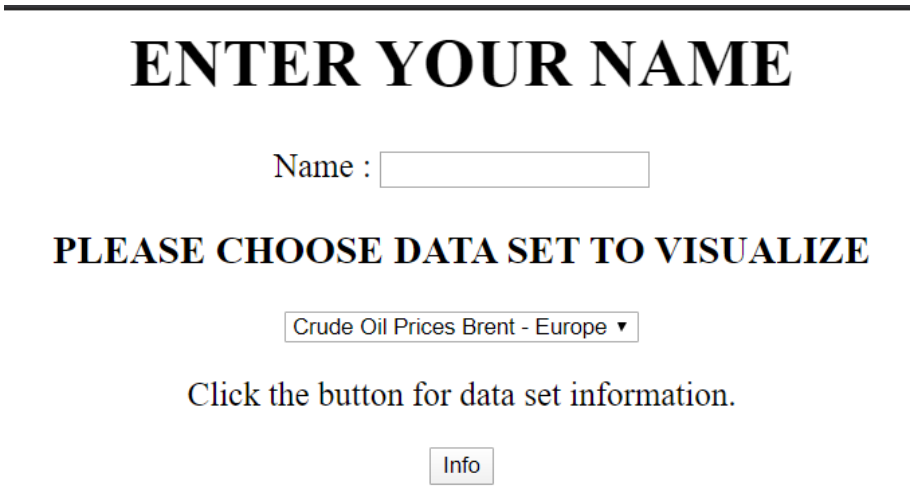
3.2.1.1 NumPy package

Η βιβλιοθήκη NumPy (Numerical Python) είναι το βασικό επιστημονικό πακέτο για την Python (scientific computing) και περιλαμβάνει μεταξύ άλλων, ένα ισχυρό array object N-διαστάσεων, εργαλεία για την ενσωμάτωση κώδικα C / C ++ και Fortran, χρήσιμες γραμμικές άλγεβρες, μετασχηματισμός Fourier και δυνατότητες τυχαίων αριθμών. Σύμφωνα με το επίσημο documentation, εκτός από τις προφανείς επιστημονικές του χρήσεις, το NumPy μπορεί επίσης να χρησιμοποιηθεί ως ένα αποτελεσματικό πολυδιάστατο container γενικών δεδομένων. Μπορούν να οριστούν αυθαίρετοι τύποι δεδομένων. Αυτό επιτρέπει στο NumPy να ενσωματώνεται απρόσκοπτα και γρήγορα με μια μεγάλη ποικιλία βάσεων δεδομένων. Ακόμα Υπάρχουν διάφοροι τρόποι δημιουργίας arrays στο NumPy. Για παράδειγμα, μπορούμε να δημιουργήσουμε έναν πίνακα από μια κανονική λίστα Python ή πλειάδα (tuple) χρησιμοποιώντας την array function. Ο τύπος του πίνακα που προκύπτει συνάγεται από τον τύπο των στοιχείων στις ακολουθίες.

3.2.1.2 Pandas package

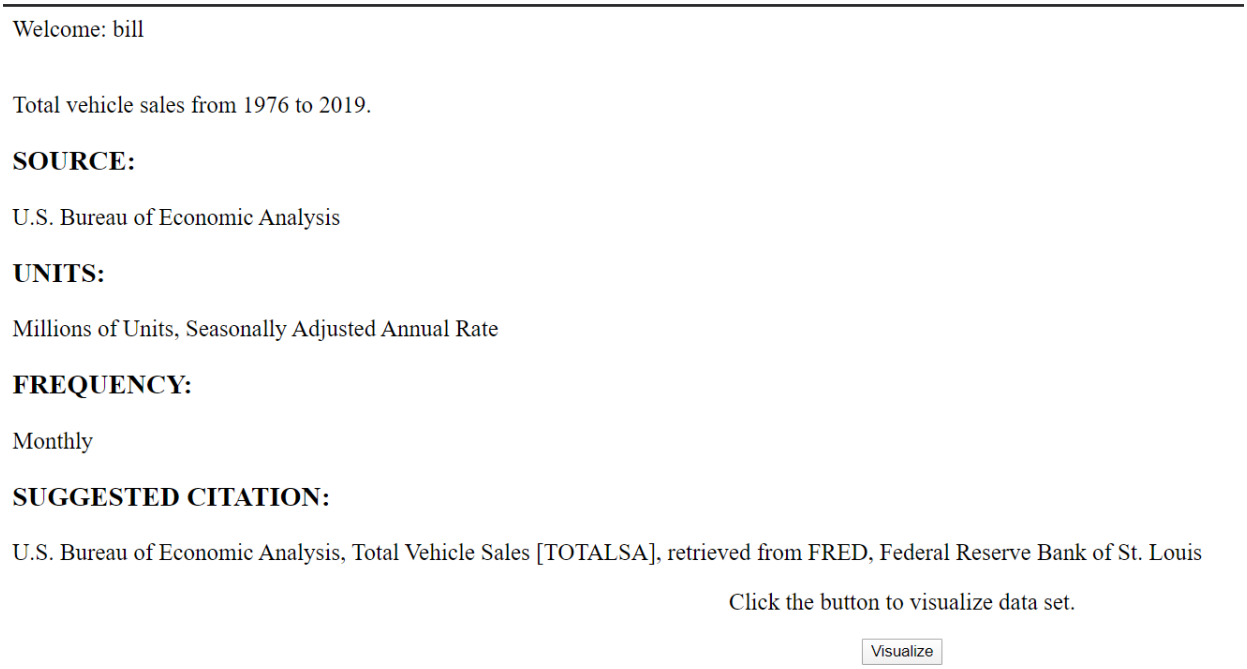
Pandas (προέρχεται από “panel data” σύμφωνα με την Wikipedia) είναι μια βιβλιοθήκη λογισμικού γραμμένη για τη γλώσσα προγραμματισμού Python για χειρισμό και ανάλυση δεδομένων. Συγκεκριμένα, προσφέρει δομές δεδομένων και λειτουργίες για χειρισμό αριθμητικών πινάκων και χρονοσειρών. Panel data είναι ένας όρος οικονομετρίας για σύνολα δεδομένων που περιλαμβάνει παρατηρήσεις σε πολλαπλές χρονικές περιόδους. Όταν πρόκειται για την ανάλυση δεδομένων με την Python είναι ένα από τα πιο προτιμώμενα και ευρέως χρησιμοποιούμενα εργαλεία για τη διαμόρφωση δεδομένων. Παίρνει δεδομένα (όπως ένα αρχείο CSV ή TSV ή μια βάση δεδομένων SQL) και δημιουργεί ένα αντικείμενο Python με σειρές και στήλες που ονομάζονται πλαίσιο δεδομένων (data frame) που μοιάζει πολύ με τον πίνακα σε ένα στατιστικό λογισμικό όπως το Excel για παράδειγμα. Αυτό σημαίνει ότι είναι πολύ πιο εύκολο να δουλέψουμε σε σύγκριση με την εργασία με λίστες (lists) ή λεξικά (dictionaries) μέσω βρόχων ή άλλων προγραμματιστικών μεθόδων.

3.3 Παράδειγμα χρήσης της εφαρμογής.



Εικόνα 29: First page of web app

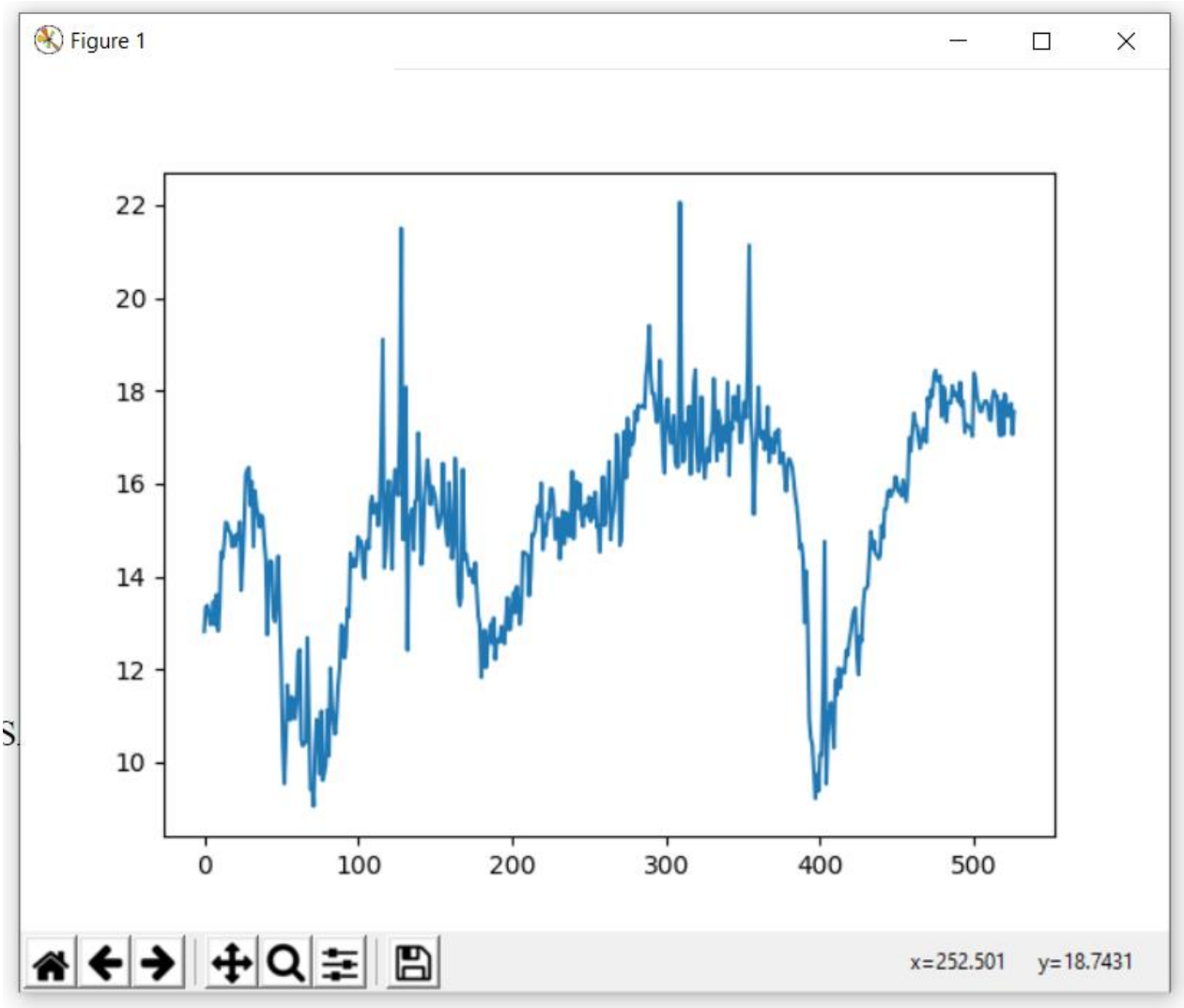
Εισάγουμε το όνομα bill στην πρώτη οθόνη της εφαρμογής και επιλέγουμε το δεύτερο data set με τίτλο "Total Vehicle sales" από το drop down list με τα τρία (3) βασικά data set που υλοποιεί η εφαρμογή. Το συγκεκριμένο data set περιέχει πωλήσεις οχημάτων από το 1967 έως το 2019. Μπορούμε να δούμε τις γενικότερες πληροφορίες του, καθώς και από που προέρχεται πατώντας το κουμπί "info".



Εικόνα 30: Web app Info page

Το συγκεκριμένο data set -όπως και τα άλλα δύο- προέρχονται από την ομοσπονδιακή τράπεζα αποθεματικών του St. Louis.

Όπως βλέπουμε στη σελίδα αυτή υπάρχει και ένα κουμπί visualize. Το συγκεκριμένο κουμπί κάνει execute μία process η οποία καλεί το υπεύθυνο για την οπτικοποίηση της χρονοσειράς python script και το τρέχει μέσα στο αρχείο JSP.



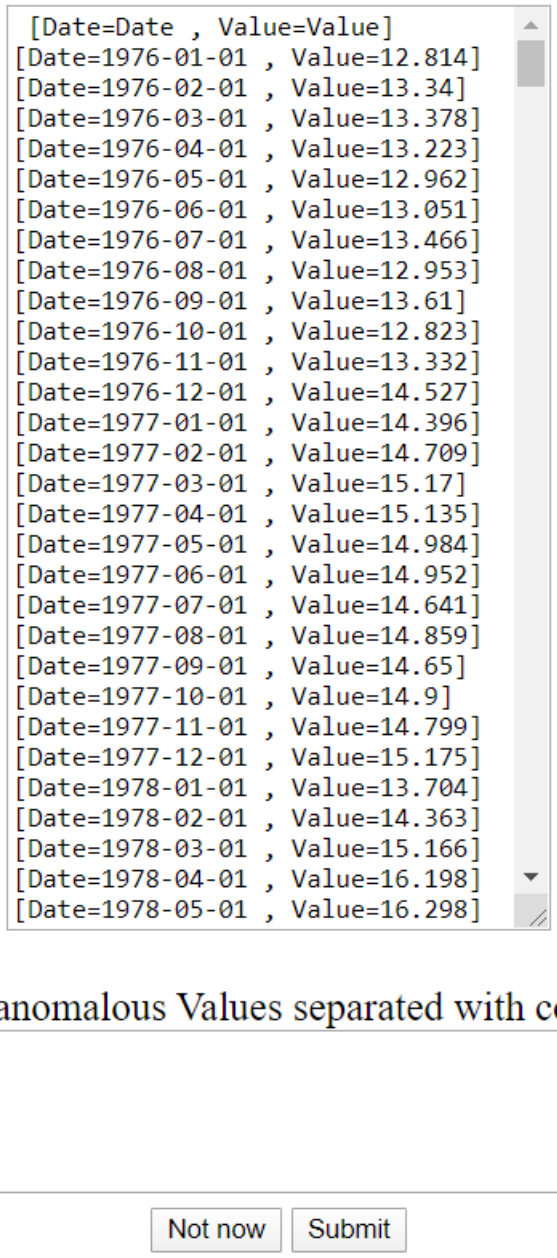
Εικόνα 31: Total Vehicle Sales data set visualization in web app

Η συγκεκριμένη χρονοσειρά έχει δεκαδικές (float) τιμές από 9 μέχρι 22 και το σύνολο των δεδομένων είναι 529.

Η επόμενη σελίδα που μεταφέρεται ο χρήστης, αφού κλείσει την χρονοσειρά που έγινε visualize, περιέχει μία λίστα (ουσιαστικά και προγραμματιστικά είναι ένα text area) με το κάθε ένα index (Date - Value) της χρονοσειράς. Αυτό αποσκοπεί, στο να το μελετήσει ο χρήστης και με βάσει το τι πιστεύει να γράψει στο άδειο Text Field τα values που αυτός θεωρεί anomalous. Ανάμεσα στα values που γράφει ο χρήστης είναι υποχρεωτικό να υπάρχει κόμμα (,) για να το κάνει πιο εύκολο από τον αλγόριθμο να χωρίσει τα δεδομένα κατά την επεξεργασία. Η διαδικασία χρειάζεται για να μπορέσει να εφαρμοστεί η λειτουργία της εφαρμογής για εύρεση του πιο ακριβή -όσον αφορά την

ανακάλυψη ανωμαλιών- αλγορίθμου από τους πέντε που χρησιμοποιούνται. Ουσιαστικά μπορεί να πει κανείς ότι τα data μας γίνονται labeled για να βρεθεί το accuracy, ενώ δεν ήταν.

Ο χρήστης έχει τη δυνατότητα εάν το επιθυμεί να επιλέξει το κουμπί not now που όμως δεν θα του παρέχει τη δυνατότητα αργότερα να συγκρίνει τους αλγορίθμους ως προς την ακρίβειά τους.

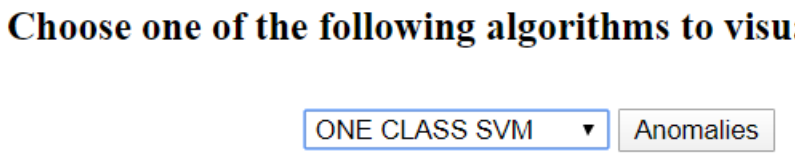


Εικόνα 32: web app page where user can write anomalous values as input

Όποιο κουμπί και να επιλέξει ο χρήστης η επόμενη σελίδα που θα φορτώσει από τον server είναι αυτή της επιλογής του αλγορίθμου που θα ήθελε να κάνει το visualize των ανωμαλιών.

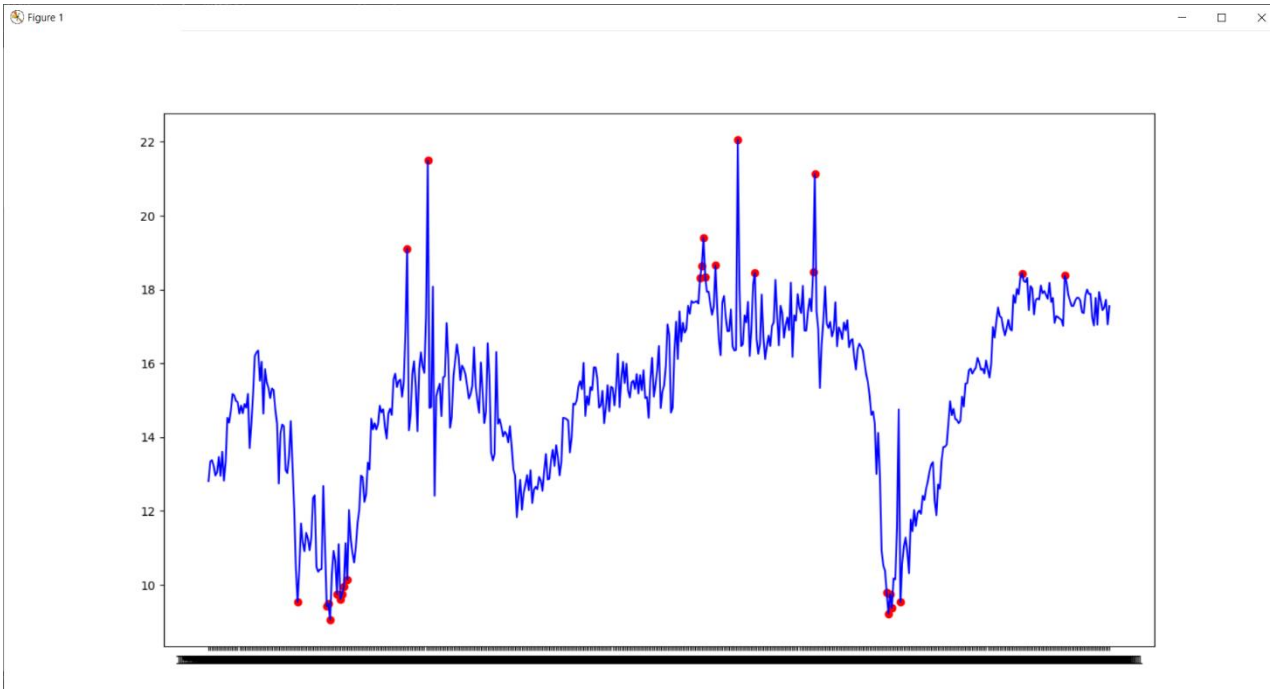
Όπως βλέπουμε στην παρακάτω εικόνα υπάρχει ένα drop down list που εμπεριέχει τους 5 αλγορίθμους, οι οποίοι ονομαστικά είναι: One Class SVM, Isolation Forest, K-Means, Z-Score, Symbolic Aggregate approximation (S.A.X.). Ο χρήστης επιλέγει τον έναν της αρεσκείας του και

πατάει το κουμπί anomalies που θέτει σε λειτουργία το script που βρίσκει και οπτικοποιεί τις ανωμαλίες του data set που έχει επιλεγεί από την αρχική σελίδα της εφαρμογής. Τυχαία επιλέγεται ο αλγόριθμός One Class SVM.



Εικόνα 33: Web app page for choosing one from a list of five algorithms for anomaly detection on data set

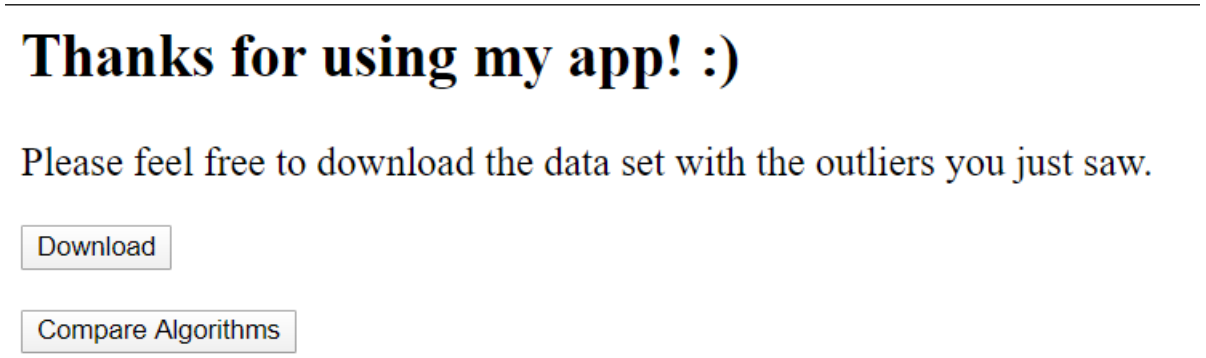
Μετά από λίγα δευτερόλεπτα πάλι με την ίδια διαδικασία της process που καλεί το κατάλληλο python script εμφανίζεται το παρακάτω γράφημα.



Εικόνα 34: Γραφική απεικόνιση της χρονοσειράς και των ανωμαλιών (με την επιλογή του SVM.)

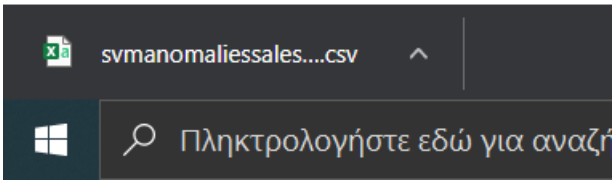
Όπως βλέπουμε πάνω στην οπτικοποιημένη χρονοσειρά (με μπλε χρώμα) οι ανωμαλίες που αναγνωρίζει ο αλγόριθμος One Class SVM τοποθετούνται σαν κουκίδες (με κόκκινο χρώμα) πάνω στην τιμή στην οποία βρέθηκαν.

Όταν ο χρήστης τελειώσει με την παρατήρηση της χρονοσειράς με τις ανωμαλίες και κλείσει το γράφημα, αυτόματα μεταφέρεται σε μία από τις τελευταίες σελίδες της εφαρμογής όπου έχει δύο τελευταία κουμπιά. Το κουμπί download και το κουμπί compare algorithms.



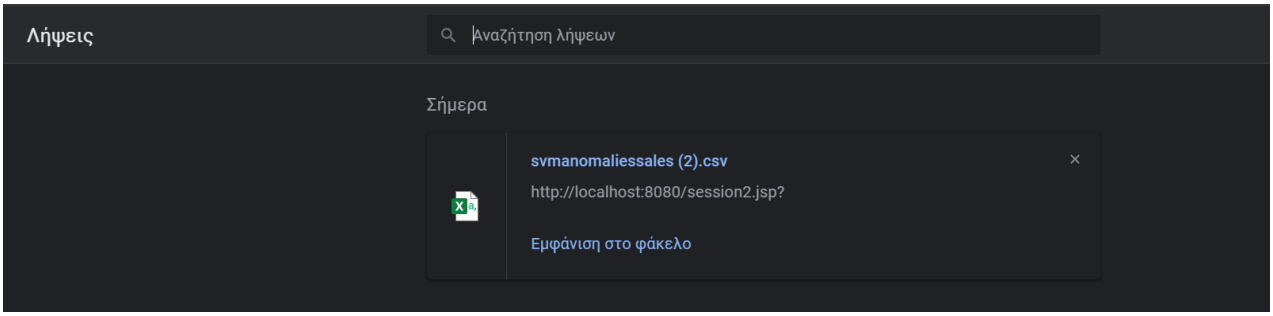
Εικόνα 35: web app last page with options to download csv or compare algorithms

Το κουμπί download επιτρέπει στον χρήστη να κατεβάσει από το server στον υπολογιστή του, ένα αρχείο .csv που περιέχει πληροφορίες για τις ανωμαλίες στο data set που έγινε visualize. Περιέχει δηλαδή τις τιμές που ο αλγόριθμος βρήκε ως ανωμαλίες καθώς και την χρονολογική τιμή που αυτά είναι καταγεγραμμένα. Συνεπώς Date και Value μόνο για τις ανωμαλίες. Για παράδειγμα στο συγκεκριμένο παράδειγμα το csv αρχείο που θα κατέβει αν ο χρήστης επιλέξει το κουμπί download θα περιέχει 28 «εγγραφές» (οι ανωμαλίες που βρέθηκαν), έναντι των 529 που ήταν το αρχικό data set.



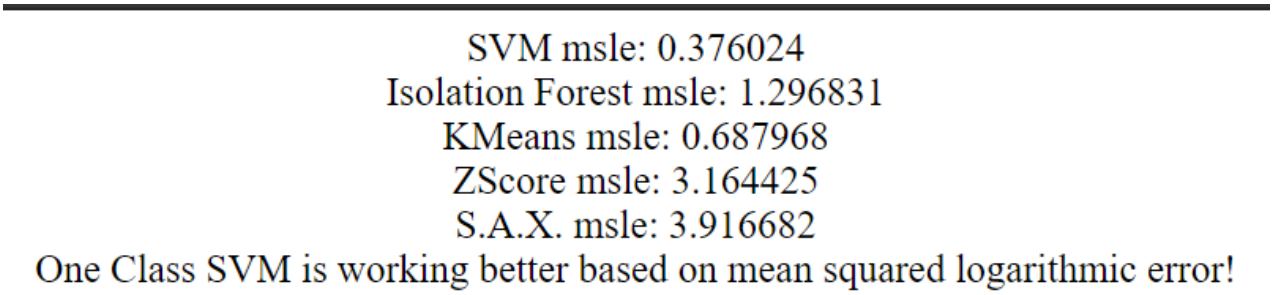
Εικόνα 36: downloaded file example no.1

Για του λόγου το αληθές η παρακάτω εικόνα δείχνει τις λήψεις, που πραγματικά επιβεβαιώνουν ότι το αρχείο κατέβηκε.



Εικόνα 37: downloaded file example no.2

Το δεύτερο κουμπί (compare algorithms) θέτει σε λειτουργία το script εκείνο, που επιτρέπει σε όλους τους αλγόριθμους να τρέξουν και ανάλογα με τα δεδομένα που κατέγραψε ο χρήστης σαν ανωμαλίες στην τρίτη σελίδα της εφαρμογής να εξάγουν ένα αποτέλεσμα ακρίβειας, σύμφωνα με το ποσοστό λάθους. Το ποσοστό λάθος που χρησιμοποιείται στο συγκεκριμένο script είναι το mean squared logarithmic error (msle). Ουσιαστικά όσο πιο κοντά στο μηδέν (0) τόσο πιο ακριβής είναι ο αλγόριθμος (δηλαδή 0% λάθος). Για παράδειγμα το 0.37 του SVM σημαίνει ότι το ποσοστό λάθους είναι 37%. Να τονιστεί εδώ ότι είναι πλασματικά νούμερα γιατί οι υπολογισμοί βασίζονται στις -κατά το χρήστη- αναμενόμενες τιμές ανωμαλιών που ο ίδιος έβαλε χωρισμένες με κόμμα σε προηγούμενη σελίδα. Αλλά αυτό που μας ενδιαφέρει είναι από τους 5 αλγορίθμους ποιος έχει το μικρότερο msle, που σημαίνει ότι είναι και αυτός με το μικρότερο ποσοστό λάθους και μεγαλύτερη ακρίβεια όσον αφορά την ανακάλυψη των ανωμαλιών.



Εικόνα 38: compare algorithms by msle page

Τέλος υπολογίζοντας το script τον μικρότερο αριθμό από τους πέντε που προκύπτουν εμφανίζει μήνυμα με τον αλγόριθμο που συμπεριφέρθηκε καλύτερα.

3.3.1 Γιατί MSLE

Το μέσο τετραγωνικό λογαριθμικό σφάλμα (MSLE) μπορεί να ερμηνευθεί ως μέτρο της αναλογίας μεταξύ των πραγματικών και των προβλεπόμενων τιμών. Η εισαγωγή του λογάριθμου κάνει το MSLE να ενδιαφέρεται μόνο για τη σχετική διαφορά μεταξύ της πραγματικής και της προβλεπόμενης τιμής, ή με άλλα λόγια, ενδιαφέρεται μόνο για την ποσοστιαία διαφορά μεταξύ τους. Αυτό σημαίνει ότι το MSLE θα αντιμετωπίσει μικρές διαφορές μεταξύ μικρών πραγματικών και προβλεπόμενων τιμών περίπου τις ίδιες με τις μεγάλες διαφορές μεταξύ μεγάλων πραγματικών και προβλεπόμενων τιμών. Η εικόνα παρακάτω επεξηγεί ακριβώς αυτό που προαναφέρθηκε.

True value	Predicted value	MSE loss	MSLE loss
30	20	100	0.02861
30000	20000	100 000 000	0.03100
	Comment	big difference	small difference

Εικόνα 39: mean squared logarithmic error (msle)

Επιλέγουμε λοιπόν να χρησιμοποιούμε το MSLE όταν πιστεύουμε ότι ο στόχος μας, που εξαρτάται από την είσοδο, είναι κανονικά κατανεμημένος και δεν θέλουμε τα μεγάλα σφάλματα να τιμωρούνται σημαντικά περισσότερο από τα μικρά, σε περιπτώσεις όπου το εύρος της τιμής στόχου είναι μεγάλο. Συνήθως χρησιμοποιείται πιο συχνά σε μεθόδους όπως Regression.

3.3.2 MSLE και μαθηματικά

Η απώλεια είναι ο μέσος όρος των παρατηρηθέντων δεδομένων των τετραγώνων των διαφορών μεταξύ των πραγματικών και των προβλεπόμενων τιμών που έχουν «λογαριθμιθεί». Ο μαθηματικό τύπος είναι:

$$L(y, \hat{y}) = \frac{1}{N}\sum_{i=0}^{N}(\log(y_i + 1) - \log(\hat{y}_i + 1))^2$$

Εικόνα 40: msle math

όπου ŷ είναι η προβλεπόμενη τιμή.

Αυτή η απώλεια μπορεί να ερμηνευθεί ως μέτρο της αναλογίας μεταξύ των πραγματικών και των προβλεπόμενων τιμών, δεδομένου ότι:

$$\log(y_i + 1) - \log(\hat{y}_i + 1) = log\left(\frac{y_i + 1}{\hat{y}_i + 1}\right)$$

Εικόνα 41: msle math no.2

Ο λόγος που το «1» προστίθεται και στο y και στο ŷ είναι για μαθηματική ευκολία, καθώς το log(0) δεν έχει οριστεί, αλλά και τα δύο y ή ŷ μπορεί να είναι 0.

4 *Πειράματα*

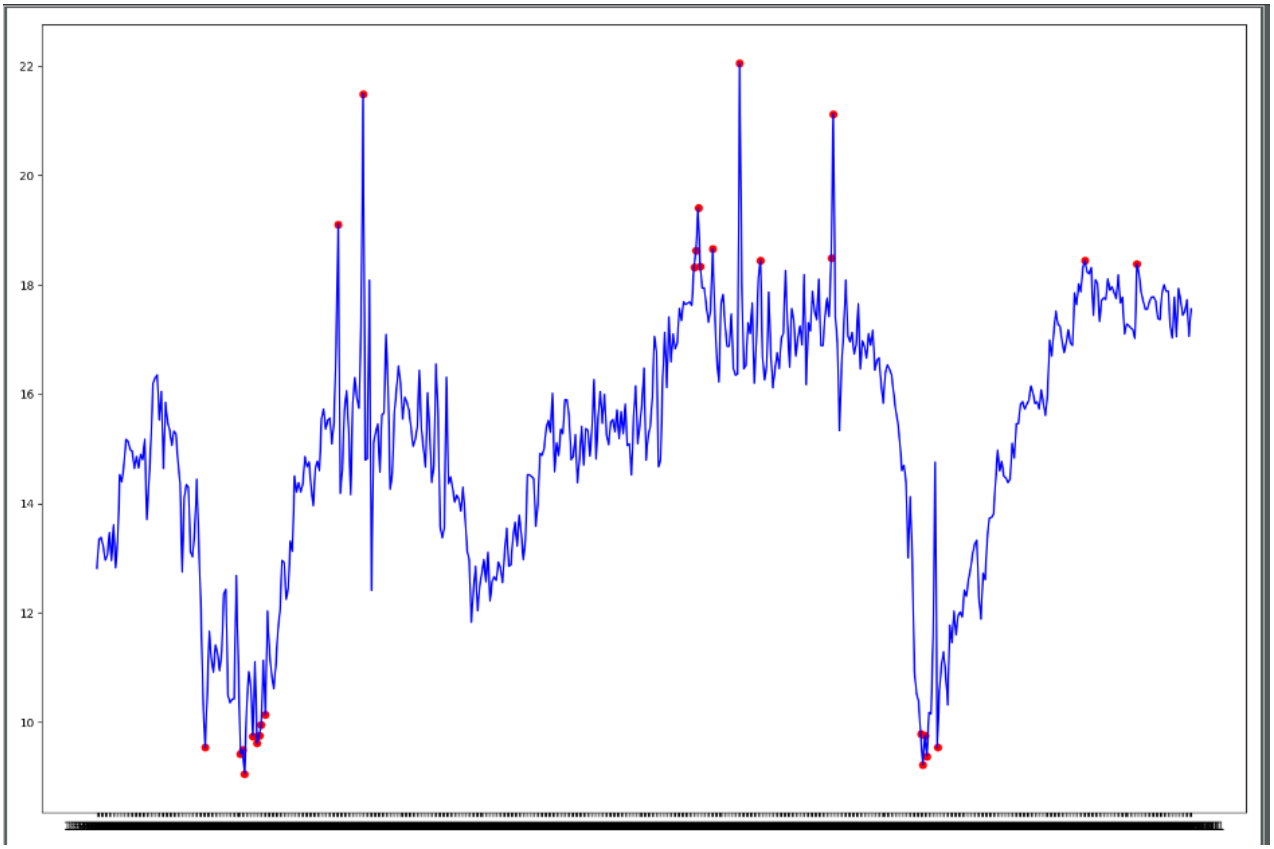
Οι τελικές παραμετροποιήσεις των αλγορίθμων προέκυψαν μέσα από πειράματα. Και στα τρία data set, για κάθε έναν αλγόριθμο ξεχωριστά (εκτός από τη μέθοδο S.A.X. που έχει διαφορετική προσέγγιση) οι δοκιμές εστιάστηκαν κυρίως στο outlier fraction, που πρακτικά είναι το ποσοστό εκείνο που αναμένεται να είναι ανωμαλίες σε σχέση με το σύνολο των δεδομένων. Ουσιαστικά τις περισσότερες φορές έχει να κάνει με το τι πιστεύει ο προγραμματιστής. Όσον αφορά τις τιμές που μπορεί να πάρει η μεταβλητή αυτή, ορίστηκαν κάποιες υποθέσεις / περιορισμοί πριν ξεκινήσουν οι δοκιμές και όσο εργαζόμουν με τις παραμετροποιήσεις προέκυπταν κι άλλες. Αναφέρονται σύντομα:

1. Το ποσοστό των ανωμαλιών στο data set δεν μπορεί να ξεπερνάει το 50% -σε μερικούς αλγορίθμους αυτό ίσχυε από default.
2. Επίσης για ευκολία πράξεων κυρίως αλλά και επειδή δεν υπάρχει τεράστια διαφορά στα αποτελέσματα τα ποσοστά που δοκιμάστηκαν δεν ήταν ανά μονάδα αλλά ανά πέντε (5). Δηλαδή αν δοκίμασα το 20% σαν ποσοστό μπορούσα μόνο μετά να δοκιμάσω ή το 25 ή το 15, και όχι το 19, το 21, το 22 κτλ. (εξαίρεση αυτού του κανόνα αποτέλεσε ο Isolation Forest και θα εξηγηθεί παρακάτω ο λόγος)
3. Οι έλεγχοι έγιναν φθίνοντας από το 50% που ήταν το ταβάνι των υπολογισμών.
4. Καθώς έφθινε το ποσοστό παρατηρήθηκε ότι από το 50% έως το 35% η διαφορά των αποτελεσμάτων ήταν μικρή και αμελητέα σε σχέση με τα ποσοστά που βρίσκονταν από 30% και κάτω.
5. Τέλος τα data set έχουν μικρό ποσοστό ανωμαλιών, οπότε, όπως ήταν αναμενόμενο οι τελικές μετρήσεις και τα περισσότερα πειράματα έγιναν μεταξύ 5% και 10%, τα οποία αν σκεφτεί κανείς ότι τα μεγέθη των instances που είχαν τα data set (το Total Vehicle Sales έχει 529, το Crude Oil prices Brent 394 και το Unemployment Rate 866) είναι απόλυτα λογικό.

Στην συνέχεια αναφέρονται με τη σειρά τα αποτελέσματα των πειραμάτων και η ακριβής παραμετροποίηση των μοντέλων.

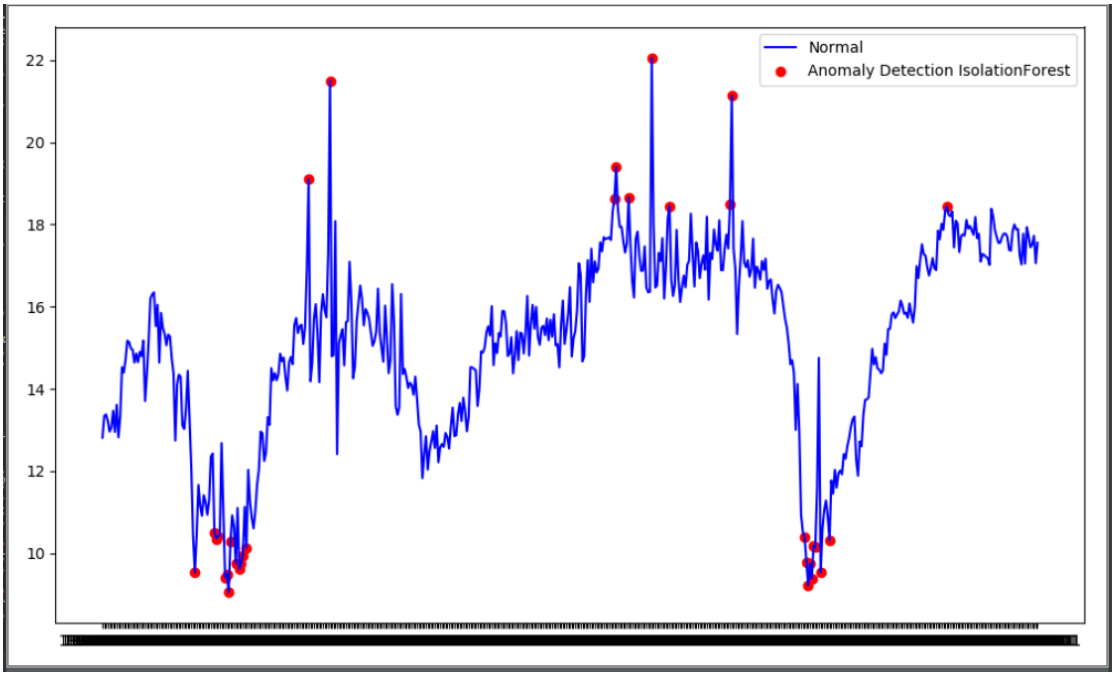
4.1 Total Vehicle Sales data set.

Το συγκεκριμένο data set όπως προαναφέρθηκε, αποτελείται από 529 τιμές. Το outlier fraction που χρησιμοποιήθηκε για τον αλγόριθμο One Class SVM είναι 0.05. Αυτό σημαίνει πως το μοντέλο αναμένεται να αναγνωρίσει το 5% του συνόλου των δεδομένων ως ανωμαλίες. Πράγματι οι ανωμαλίες που προκύπτουν είναι 27 και φαίνονται στο παρακάτω διάγραμμα με κόκκινες κουκίδες.



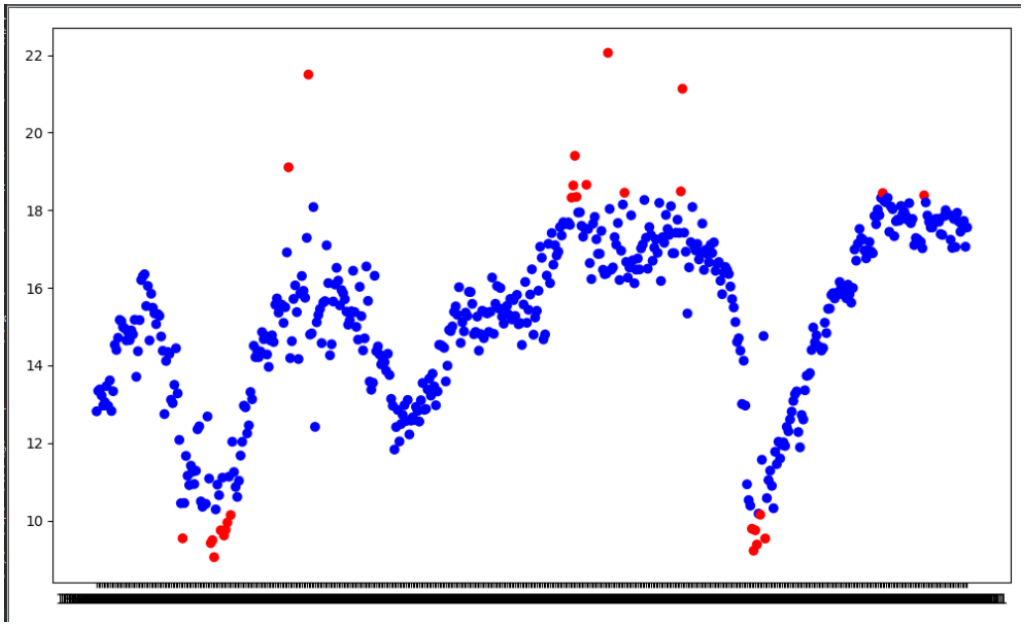
Εικόνα 42: Πείραμα 1. anomaly detection on Total Vehicle Sales data set using SVM

Όσον αφορά τον αλγόριθμο Isolation Forest, παρατηρήθηκε ότι στην λειτουργία του έχει ουσιαστική διαφορά ακόμα και μία ποσοστιαία μονάδα. Έτσι, σε δοκιμές ανα μία μονάδα από το 10% και κάτω, παρατηρείται ότι οι συνθήκες που ικανοποιούν την εύρεση ανωμαλιών είναι για outlier fraction ίσο με 0.06. Δηλαδή 6% του συνολικού μεγέθους του data set. Πράγματι οι ανωμαλίες που προκύπτουν είναι 31 και φαίνονται στο παρακάτω διάγραμμα με κόκκινες κουκίδες.



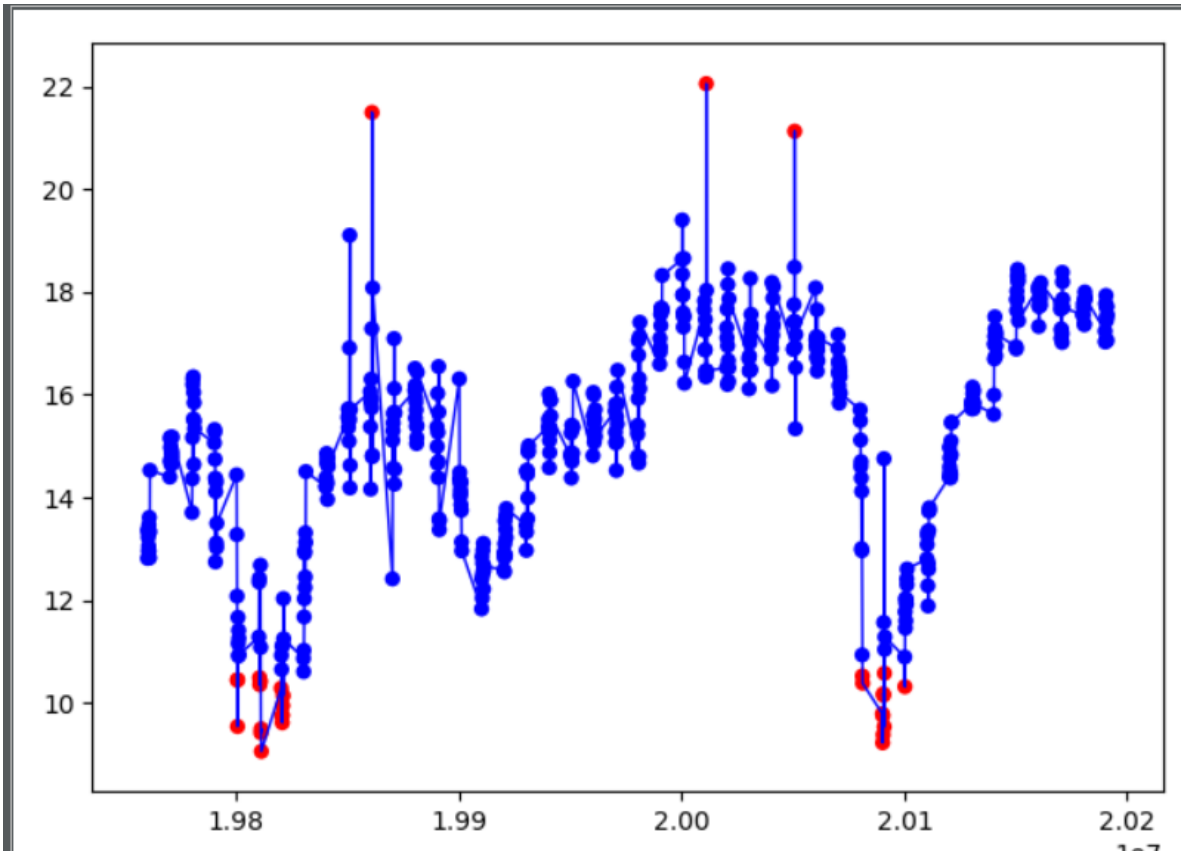
Εικόνα 43: Πείραμα 2. anomaly detection on Total Vehicle Sales data set using Isolation Forest

Στον αλγόριθμο K-Means επειδή έχουμε να κάνουμε με clusters και ομαδοποίηση, οι υπολογισμοί πρέπει να είναι όσο πιο ακριβής γίνεται. Έτσι μετά από ελέγχους κοντά στο πεδίο που ορίστηκε για τους υπόλοιπους αλγορίθμους το outlier fraction για τον K-Means ορίστηκε σε 0.055. Ίσο δηλαδή με το 5.5% του συνολικού data set. Πράγματι από το σύνολο των δεδομένων που εμφανίζονται, οι ανωμαλίες είναι 28 και φαίνονται στο παρακάτω διάγραμμα με κόκκινες κουκίδες.



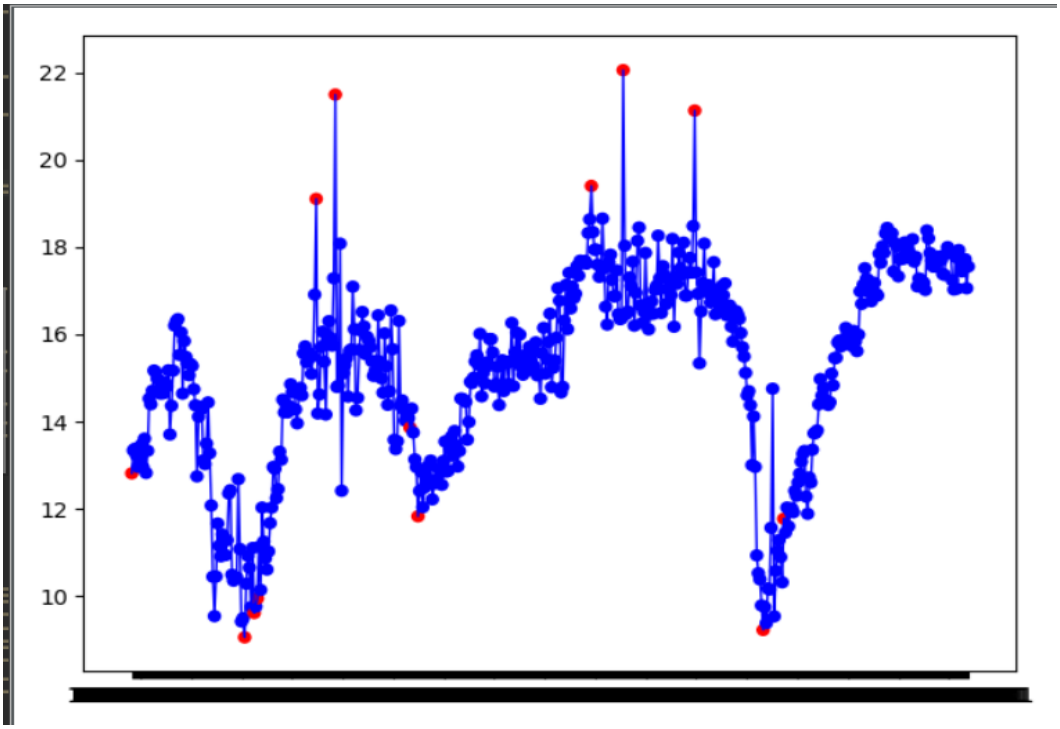
Εικόνα 44: Πείραμα 3. anomaly detection on Total Vehicle Sales data set using K-Means

Στην περίπτωση του Z-Score ο τρόπος με τον οποίο υπολογίζεται αν ένα data point είναι ανωμαλία ή όχι, είναι λίγο διαφορετικός. Όσο μεγαλύτερος είναι ο αριθμός των τυπικών αποκλίσεων μακριά από τον μέσο όρο, τόσο περισσότερο είναι πιθανό να υπάρχει μια ανωμαλία σε αυτό το σημείο δεδομένων. Με άλλα λόγια, όσο πιο μακριά από το κέντρο, μεγαλύτερη η πιθανότητα να είναι ανωμαλία (outlier). (βλ. Κεφάλαιο 2) Από το σύνολο των δεδομένων που εμφανίζονται, οι ανωμαλίες είναι 30 και φαίνονται στο παρακάτω διάγραμμα με κόκκινες κουκίδες.



Εικόνα 45: Πείραμα 4. anomaly detection on Total Vehicle Sales data set using Z-Score

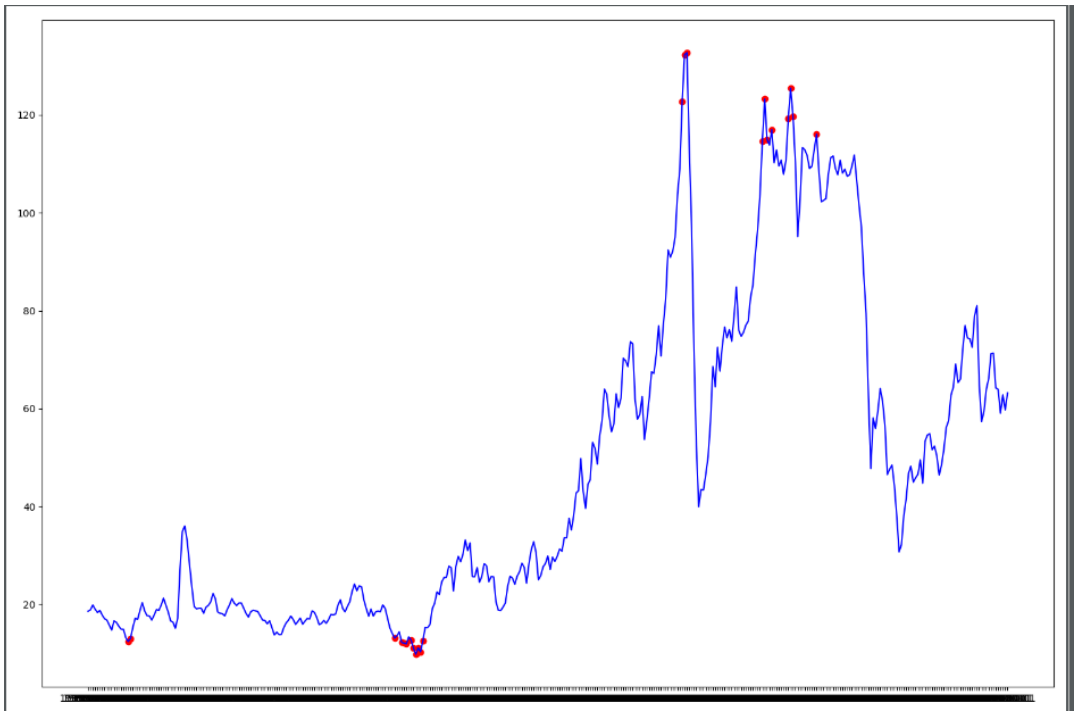
Η μέθοδος S.A.X. ακολουθεί μία τελείως διαφορετική υλοποίηση. Εξηγείται στα προηγούμενα κεφάλαια λεπτομερώς, αλλά αυτό που έχει σημασία σε αυτήν τη μέθοδο είναι το μέγεθος παραθύρου. Συνήθως το μέγεθος παραθύρου πρέπει να είναι μεγαλύτερο ή ίσο από το PAA size. Στην περίπτωση τη δική μας, λόγω της προσέγγισης που ακολουθήθηκε το μέγεθος παραθύρου προσαρμόστηκε στην τιμή 1. Σε αυτήν την τιμή ανταποκρίνεται καλύτερα και στην εύρεση των ανωμαλιών. Οι ανωμαλίες που βρίσκει είναι 13 και φαίνονται παρακάτω στο διάγραμμα με κόκκινες κουκίδες.



Εικόνα 46: Πείραμα 5. anomaly detection on Total Vehicle Sales data set using S.A.X. method

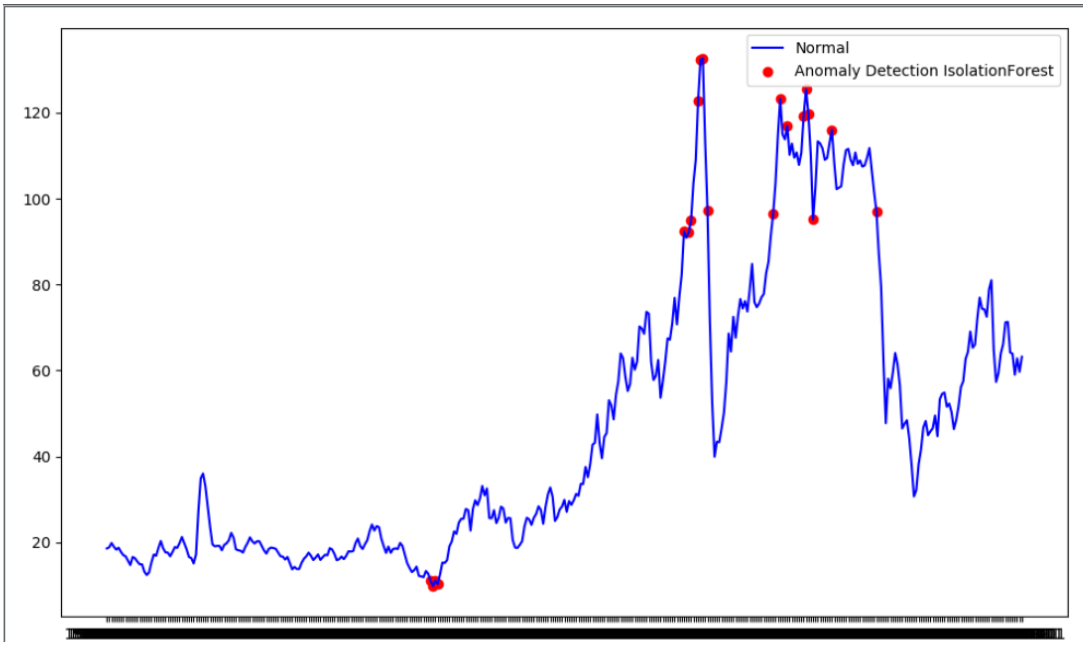
4.2 Crude Oil prices Brent data set

Το συγκεκριμένο data set όπως προαναφέρθηκε, αποτελείται από 394 τιμές. Το outlier fraction που χρησιμοποιήθηκε για τον αλγόριθμο One Class SVM είναι και εδώ 0.05. Δηλαδή 5% όπως προ είπαμε. Οι ανωμαλίες που προκύπτουν είναι 20 και φαίνονται στο παρακάτω διάγραμμα με κόκκινες κουκίδες.



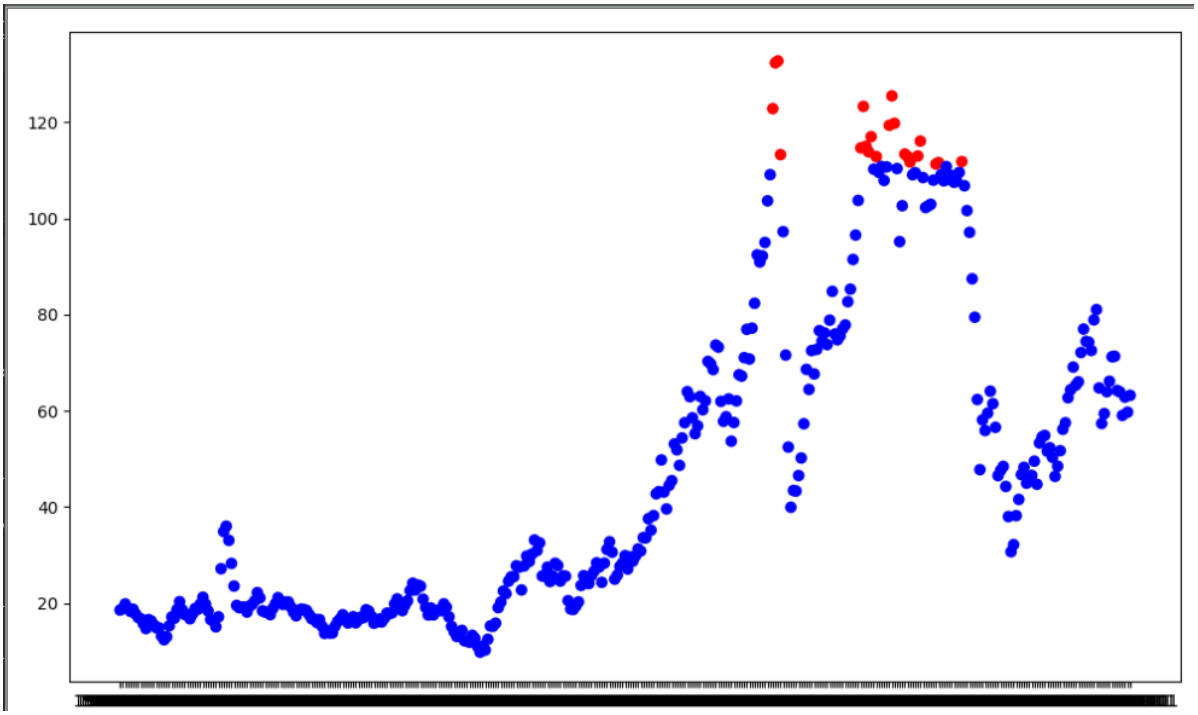
Εικόνα 47. Πείραμα 1. anomaly detection on Crude Oil prices Brent data set using SVM

Για τον αλγόριθμο Isolation Forest, οι συνθήκες που ικανοποιούν εδώ την εύρεση ανωμαλιών είναι για outlier fraction ίσο με 0.05 και όχι 0.06 όπως ήταν η τιμή για τον ίδιο αλγόριθμο στο άλλο data set. Δηλαδή 5% του συνολικού μεγέθους του data set. Πράγματι οι ανωμαλίες που προκύπτουν είναι 20 και φαίνονται στο παρακάτω διάγραμμα με κόκκινες κουκίδες.



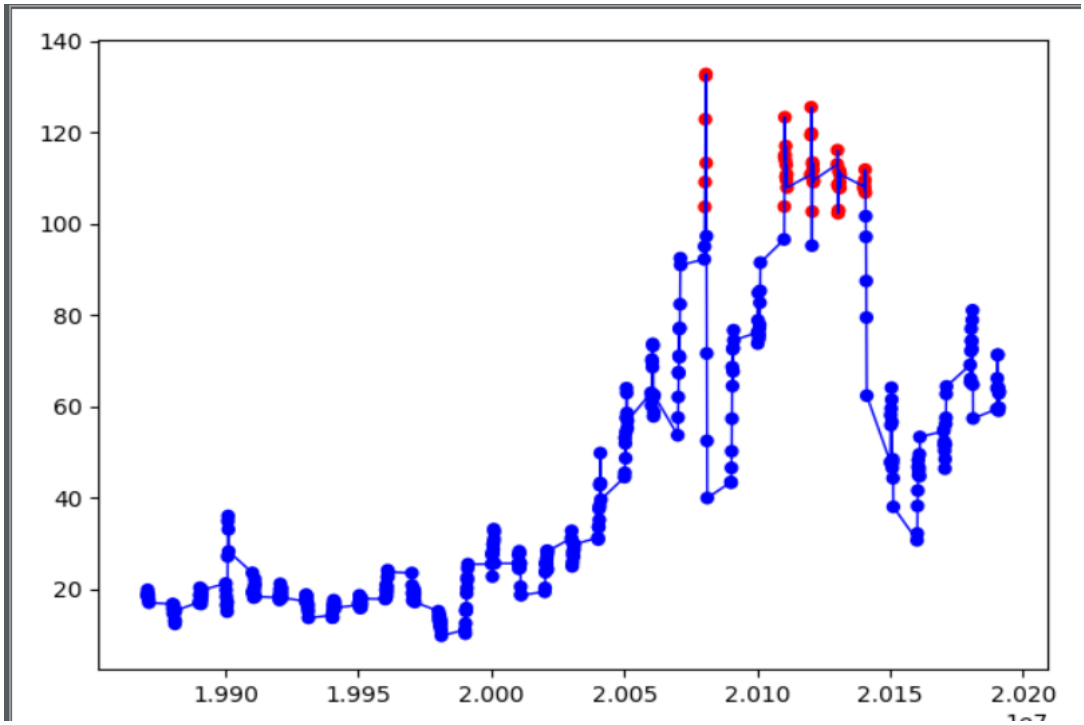
Εικόνα 48: Πείραμα 2. anomaly detection on Crude Oil prices Brent data set using Isolation Forest

Στον αλγόριθμο K-Means το outlier fraction ορίστηκε και εδώ σε 0.055. Ίσο δηλαδή με το 5.5% του συνολικού data set. Πράγματι από το σύνολο των δεδομένων που εμφανίζονται, οι ανωμαλίες είναι 21 και φαίνονται στο παρακάτω διάγραμμα με κόκκινες κουκίδες.



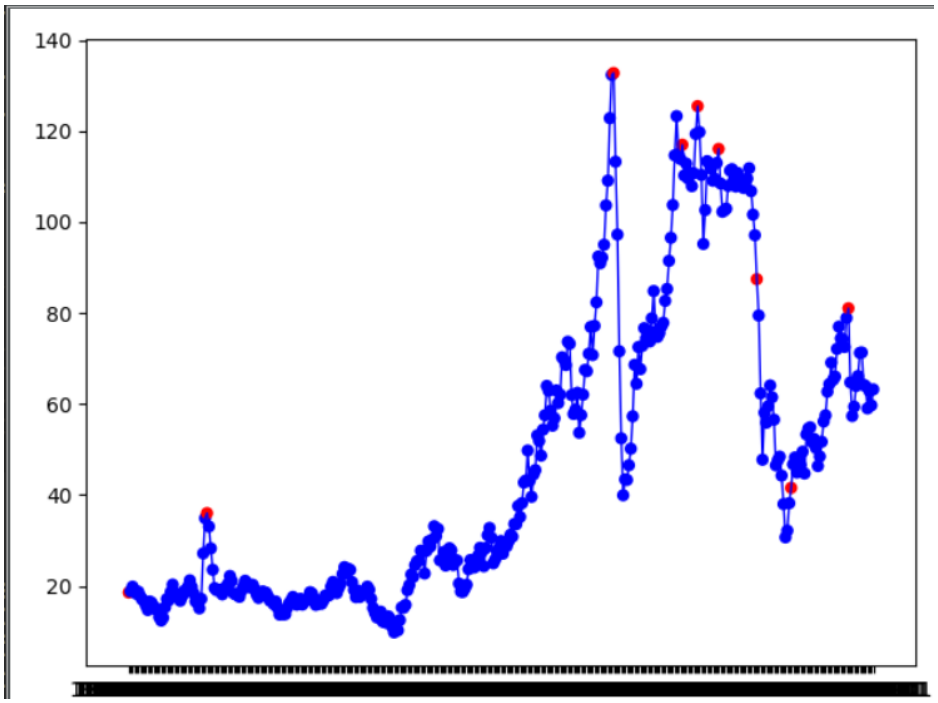
Εικόνα 49. Πείραμα 3. anomaly detection on Crude Oil prices Brent data set using K-Means

Για τον αλγόριθμο Z-Score αυτό που παρατηρείται στο συγκεκριμένο data set είναι ότι από το σύνολο των δεδομένων που εμφανίζονται, οι ανωμαλίες είναι 46 και φαίνονται στο παρακάτω διάγραμμα με κόκκινες κουκίδες.



Εικόνα 50. Πείραμα 4. anomaly detection on Crude Oil prices Brent data set using Z-Score

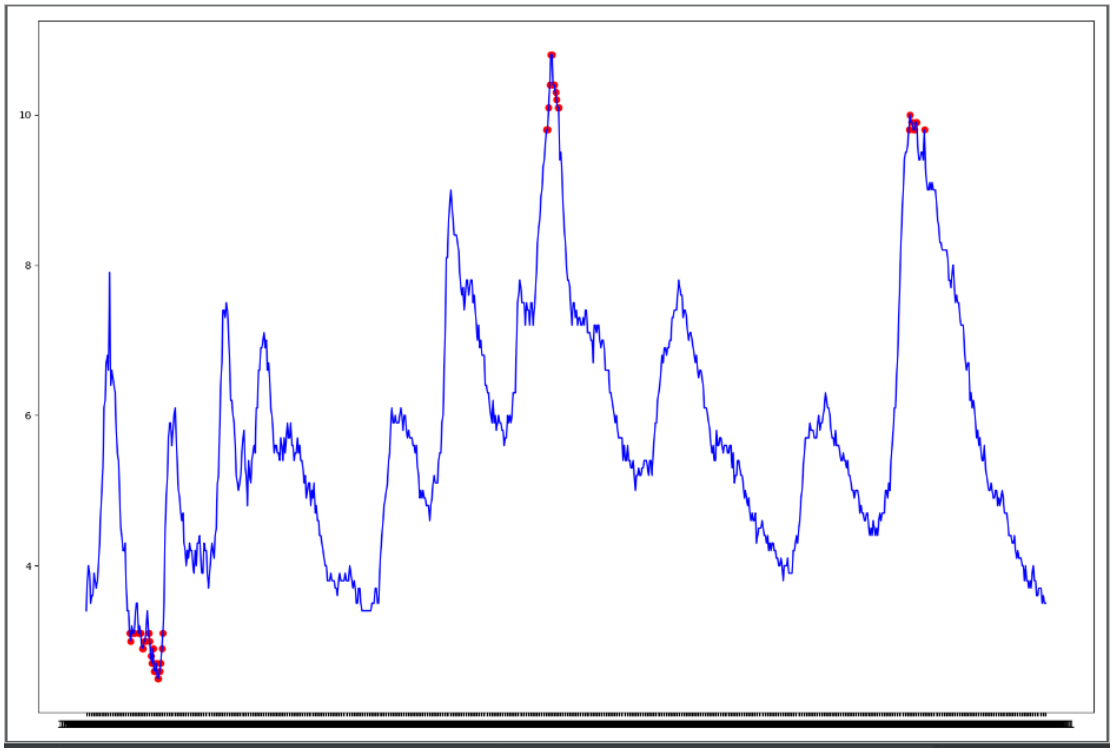
Όσον αφορά την μέθοδο S.A.X και εδώ το μέγεθος παραθύρου παραμένει 1 και οι ανωμαλίες που ο αλγόριθμος βρίσκει εδώ είναι 9 και φαίνονται στο διάγραμμα με κόκκινες κουκίδες.



Εικόνα 51: Πείραμα 5. anomaly detection on Crude Oil prices Brent data set using S.A.X. method

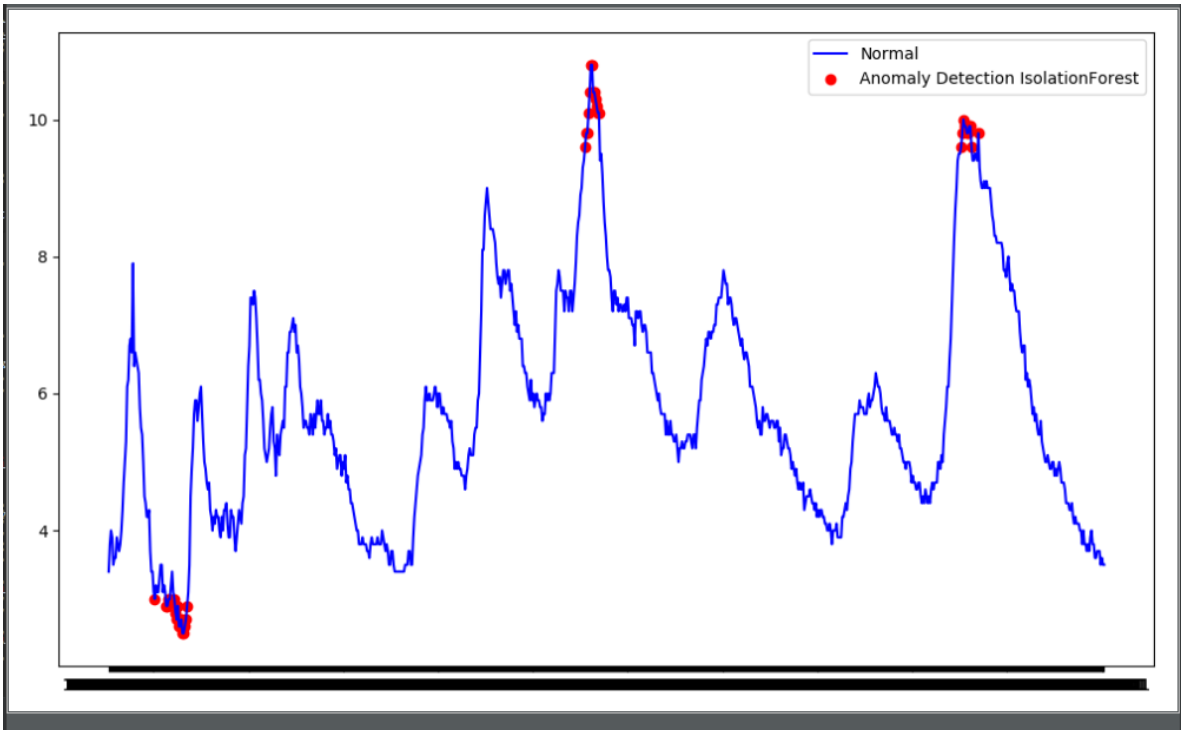
4.3 Unemployment Rate data set

Το συγκεκριμένο data set όπως προαναφέρθηκε, αποτελείται από 863 τιμές. Το outlier fraction που χρησιμοποιήθηκε για τον αλγόριθμο One Class SVM είναι και εδώ 0.05. Οι ανωμαλίες που προκύπτουν είναι 45 και φαίνονται στο παρακάτω διάγραμμα με κόκκινες κουκίδες.



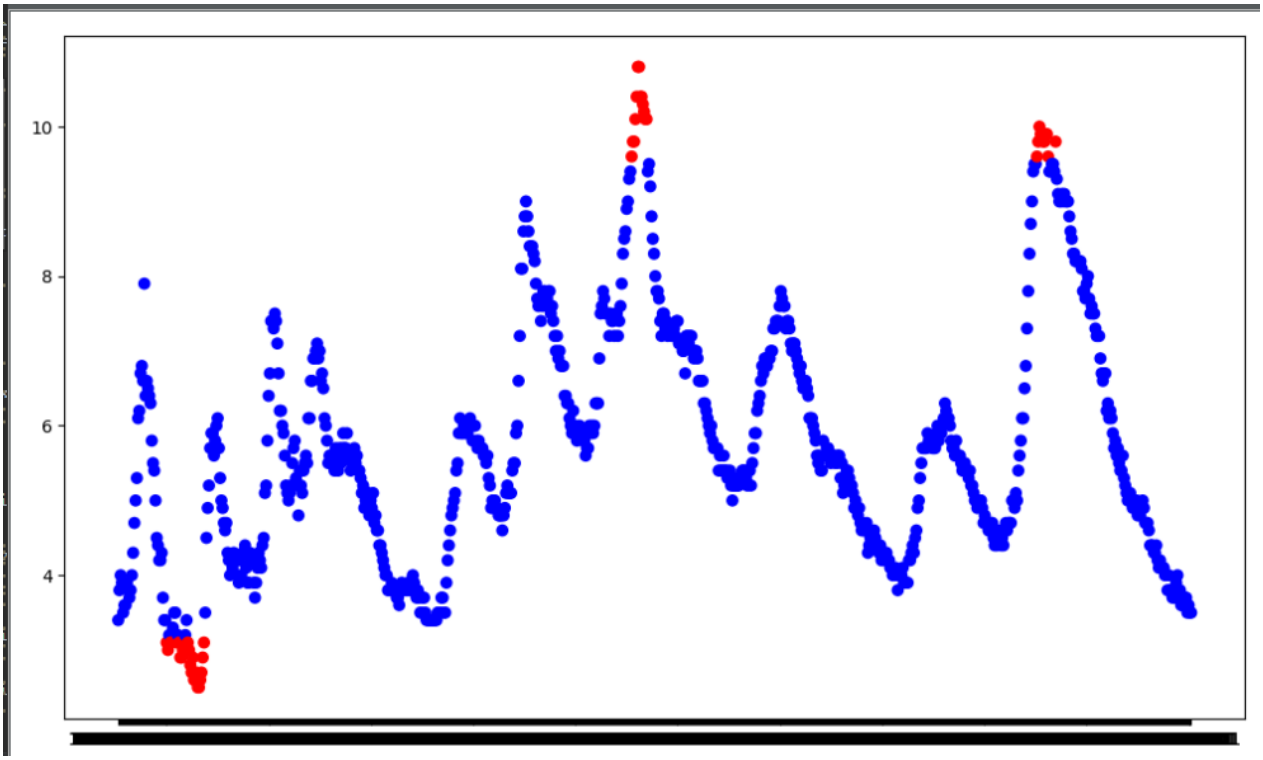
Εικόνα 52. Πείραμα 1. anomaly detection on Unemployment rate data set using SVM

Για τον αλγόριθμο Isolation Forest, οι συνθήκες που ικανοποιούν εδώ την εύρεση ανωμαλιών είναι για outlier fraction ίσο με 0.05 και όχι 0.06 όπως ήταν η τιμή για τον ίδιο αλγόριθμο στο πρώτο data set. Δηλαδή 5% του συνολικού μεγέθους του data set. Πράγματι οι ανωμαλίες που προκύπτουν είναι 41 και φαίνονται στο παρακάτω διάγραμμα με κόκκινες κουκίδες.



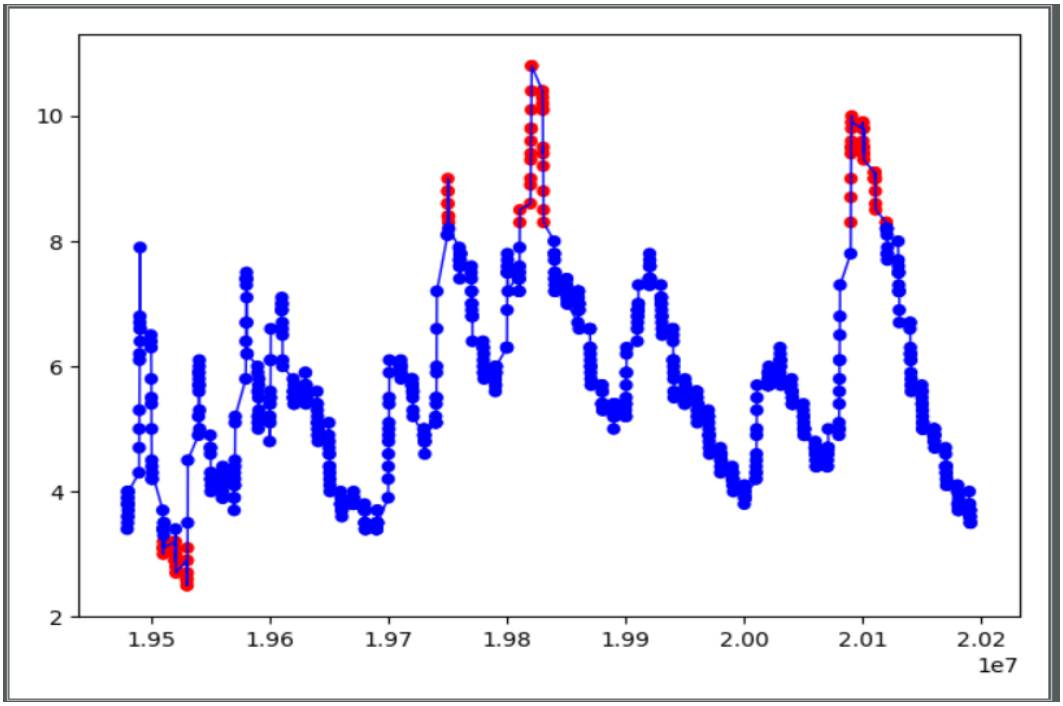
Εικόνα 53: Πείραμα 2. anomaly detection on Unemployment rate data set using Isolation Forest

Τον αλγόριθμο K-Means και εδώ παραμετροποιήσαμε με το 5.5% του συνολικού data set. Πράγματι από το σύνολο των δεδομένων που εμφανίζονται, οι ανωμαλίες είναι 48 και φαίνονται στο παρακάτω διάγραμμα με κόκκινες κουκίδες.



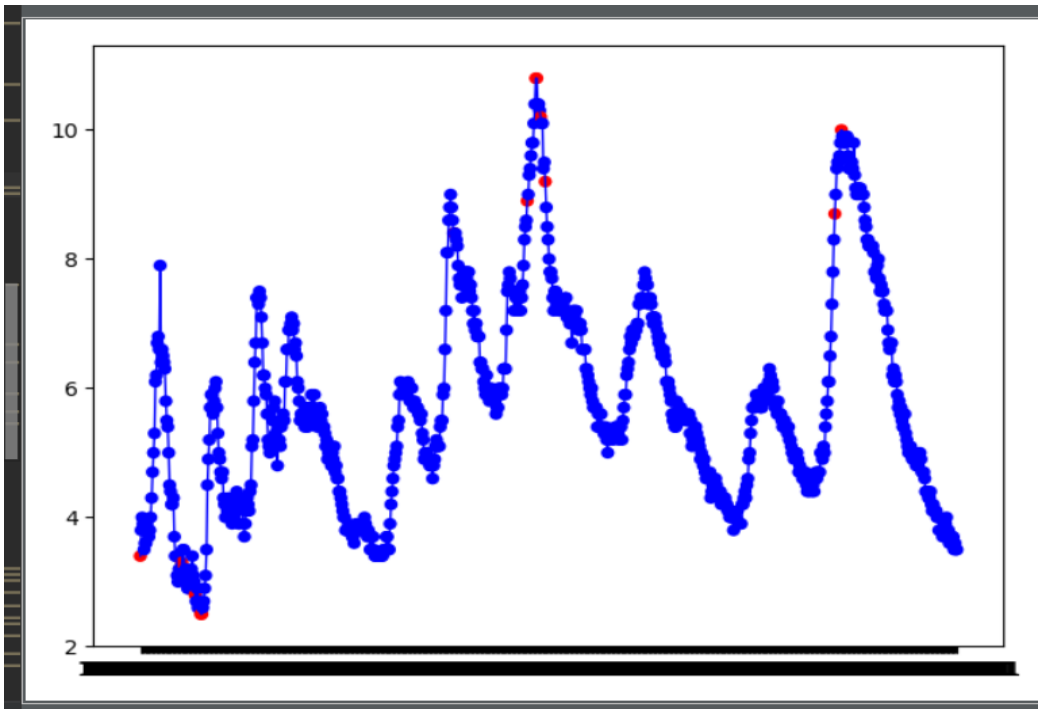
Εικόνα 54. Πείραμα 3. anomaly detection on Unemployment rate data set using K-Means

Για τον αλγόριθμο Z-Score αυτό που παρατηρείται στο συγκεκριμένο data set είναι ότι από το σύνολο των δεδομένων που εμφανίζονται, οι ανωμαλίες είναι 99 και φαίνονται στο παρακάτω διάγραμμα με κόκκινες κουκίδες.



Εικόνα 55: Πείραμα 4. anomaly detection on Unemployment rate data set using Z-Score

Για την μέθοδο S.A.X και εδώ το μέγεθος παραθύρου παραμένει 1 και οι ανωμαλίες που ο αλγόριθμος βρίσκει εδώ είναι 16 και φαίνονται στο διάγραμμα με κόκκινες κουκίδες.



Εικόνα 56: Πείραμα 5. anomaly detection on Unemployment rate data set using S.A.X. method

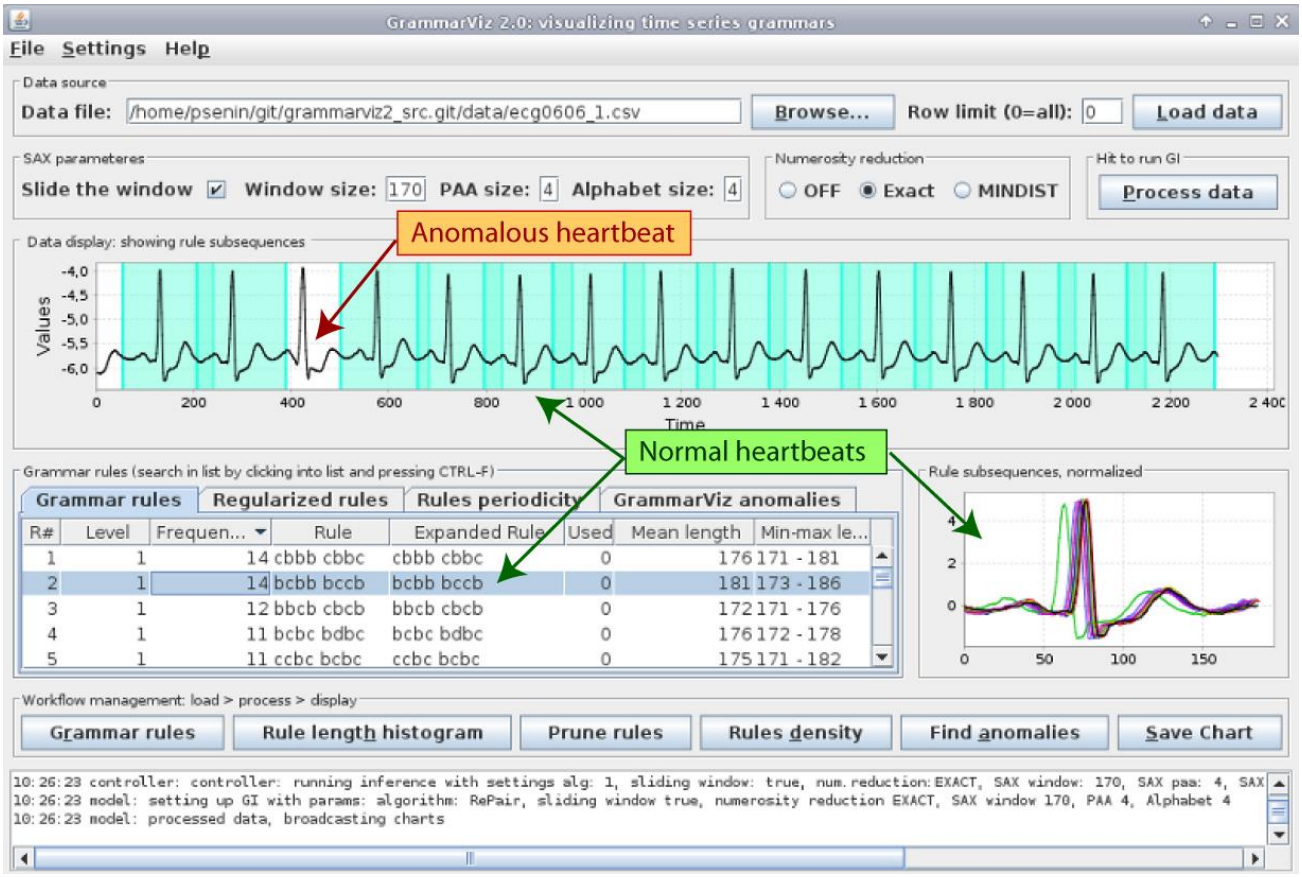
5 *Προηγούμενες προσεγγίσεις*

Το κομμάτι από τη λειτουργία της εφαρμογής που αναφέρεται στην μέθοδο Symbolic Aggregate aproXimation (S.A.X) υλοποιήθηκε βασιζόμενη στον τρόπο που λειτουργεί το GrammarViz και παραθέτω ακριβώς τον ορισμό του δημιουργού.

Είναι ένα λογισμικό για διερευνητική ανάλυση χρονοσειρών με διεπαφές GUI και CLI. Το GUI επιτρέπει τη διαδραστική ροή εργασιών εξερεύνησης χρονοσειρών που επιτρέπει την επαναλαμβανόμενη μεταβλητή διάρκεια και ανώμαλη ανακάλυψη μοτίβων από χρονοσειρές. Είναι implemented σε Java και βασίζεται σε συνεχή signal descretization με S.A.X, Grammatical Inference με Sequitur και Re-Pair και αλγοριθμική πολυπλοκότητα (Kolmogorov).

Μάλιστα η νέα έκδοση GrammarViz 3.0, εισάγει μια προσέγγιση για το κλάδεμα κανόνα γραμματικής και τη διαδικασία αυτόματης επιλογής παραμέτρων διακριτοποίησης (discretization) βάσει του άπληστου κλάσματος κανόνων γραμματικής (greedy grammar rule pruning) και MDL. Πιο συγκεκριμένα με δειγματοληψία ενός πιθανού χώρου παραμέτρων, βρίσκει ένα σύνολο παραμέτρων που παράγει την πιο συνοπτική γραμματική που περιγράφει τις παρατηρούμενες χρονικές σειρές καλύτερα, η οποία συχνά είναι κοντά στο βέλτιστο. Εφαρμόζει επίσης τους αλγορίθμους "Rule Density Curve" και "Rare Rule Anomaly (RRA)" για ανακάλυψη ανωμαλιών χρονοσειρών, που ξεπερνούν σημαντικά τον αλγόριθμο HOT-SAX για την ανακάλυψη ανωμαλιών χρονοσειρών, η οποία είναι η προσέγγιση του GrammarViz 2.0.

Η εικόνα παρακάτω απεικονίζει την γραφική απεικόνιση της εφαρμογής. Όπως φαίνεται η εφαρμογή υλοποιεί πολλές λειτουργίες από αναζήτηση στα αρχεία του υπολογιστή για την επιλογή data set, επιλογή του αριθμού των γραμμών που θα διαβαστούν από το data set, μέχρι παραμετροποίηση του μεγέθους του παραθύρου (sliding window), του μεγέθους του αλφάβητου καθώς και τον αριθμό των αλφαβήτων που θα χρησιμοποιηθούν. Δηλαδή αν τα γράμματα θα είναι από a έως d ή από a έως c και πόσο μήκος θα έχει η γραμματοσειρα. Θα είναι τριών, τεσσάρων, λιγότερων ή περισσότερων γραμμάτων (abc, abcd, adcdf…)



Εικόνα 57: GrammariViz

Ακόμα βλέπουμε ότι με κάποια κουμπιά ο χρήστης έχει τη δυνατότητα να επιλέξει να δει τους κανόνες που χρησιμοποιούνται στο dataset που τρέχει την στιγμή εκείνη, να δει γραφική αναπαράσταση της χρονοσειράς και να επιλέξει να δει τις ανωμαλίες που υπάρχουν στη συγκεκριμένη χρονοσειρά. Τέλος έχει τη δυνατότητα να αποθηκεύσει το chart πατώντας το κουμπί save chart.

6 *Συμπεράσματα – Προτάσεις.*

6.1 Συμπέρασμα

Συγκριτικά με τις προηγούμενες προσεγγίσεις, μπορεί κανείς να παρατηρήσει ότι σημαντικό ρόλο στην εκπαίδευση ενός πολύ καλού μοντέλου τεχνητής νοημοσύνης, παίζει ένα ποιοτικό σώμα δεδομένων και η όσο το δυνατόν καλύτερη παραμετροποίηση. Επίσης για μικρότερη συλλογή δειγμάτων (samples) ο αλγόριθμος SVM συμπεριφέρεται καλύτερα από τους υπόλοιπους. Γενικότερα τα αποτελέσματα που προκύπτουν μέσα από πολλές παραμετροποιήσεις και πειράματα με όλους τους αλγορίθμους και όλα τα δεδομένα, παρατηρείται ότι τις περισσότερες φορές και κατά κύριο λόγο καλύτερα και γρηγορότερα ανταποκρίνονται οι αλγόριθμοι SVM και Isolation Forest, με τις πιο πολλές φορές ίσως να «γέρνουν» προς τον SVM.

Η λύση brute force για το πρόβλημα της ανίχνευσης ανωμαλιών χρονοσειρών ή, πιο συγκεκριμένα, η ανακάλυψη μιας ασυμφωνίας ενός δεδομένου μήκους n (πολλές φορές n=1 σε point anomaly) σε χρονοσειρές T μήκους m, πρέπει να εξετάσει όλες τις πιθανές αποστάσεις μεταξύ κάθε υπο-ακολουθίας C του μήκους n και όλα τα δεδομένα που δεν ταιριάζουν. Αυτή η μέθοδος έχει πολυπλοκότητα O (m2) και είναι πραγματικά χρονοβόρα (αν όχι τις περισσότερες φορές αδύνατη) για μεγάλα σύνολα δεδομένων (data sets).

Οι λύσεις που βρέθηκαν κατά καιρούς για το πρόβλημα αυτό, είναι πολλά υποσχόμενες, όπως η μέθοδος HOT-SAX που προτείνει μια γρήγορη ευρετική τεχνική που είναι ικανή για την πραγματική ανακάλυψη ανωμαλιών (discord) με την αναδιάταξη υπο-ακολουθιών από τον πιθανό βαθμό ασυμφωνίας τους. Ενώ αυτές οι προσεγγίσεις επιτυγχάνουν επιτάχυνση αρκετών τάξεων μεγέθους από τον αλγόριθμο brute-force, το κοινό μειονέκτημά τους είναι ότι όλοι χρειάζονται το μήκος μιας πιθανής ανωμαλίας για να καθοριστεί ως είσοδος και εξάγουν discords σταθερού μήκους. Δηλαδή ο χρήστης που χειρίζεται την μέθοδο αυτή πρέπει να βάλει ως είσοδο το πιθανό μέγεθος των ανωμαλιών που αναμένονται να βρεθούν.

Όσον αφορά τη χρήση της λύσης που χρησιμοποίησα στην εφαρμογή μου για τη μέθοδο S.A.X, μέσω μεγάλου αριθμού πειραμάτων διαπιστώθηκε ότι, είναι πολύ χρονοβόρα η λύση αυτή και σε καμία περίπτωση δεν μπορεί να φτάσει σε ταχύτητα την προσέγγιση που ακολουθεί το

GrammarViz. Επιπλέον καθώς η λειτουργίες για ανίχνευση ανωμαλιών στην εφαρμογή υλοποιήθηκαν σε γλώσσα python δεν επετεύχθη η αποτελεσματικότητα, η ταχύτητα και η ακρίβεια που πέτυχε το GrammarViz, που είναι implemented σε γλώσσα Java. Γιατί όπως ειπώθηκε σε προηγούμενο κεφάλαιο η python είναι αργή επειδή είναι ευέλικτη και χρειάζεται να κάνει πολλούς υπολογισμούς προκειμένου να ορίσει σωστά τον ορισμό για κάτι.

6.2 Προτάσεις για μελλοντική έρευνα

Σύμφωνα με τα παραπάνω οι αλγόριθμοι χρησιμοποιήθηκαν μόνο για visualize και τα 3 βασικά μας data set ήταν σχετικά μικρά σε μέγεθος. Λαμβάνοντας αυτά υπόψιν μια μελλοντική έρευνα πάνω στο κομμάτι αυτό, θα μπορούσε να ενσωματώνει μεγαλύτερα και περισσότερα σώματα δεδομένων, καθώς επίσης και να γινόταν εκτεταμένο training του αλγορίθμου με αυτά τα dataset ώστε τα αποτελέσματα να ήταν ακόμα πιο ακριβή. Ακόμα θα μπορούσαν να χρησιμοποιηθούν Labeled δεδομένα και να εφαρμοστεί προφανώς μόνο classification που αποδεδειγμένα συμπεριφέρεται καλύτερα σε task όπως αυτό.

Το πρόβλημα της μεθόδου Symbolic Aggregate approximation είναι η ταχύτητα. Θα μπορούσε λοιπόν να ενσωματωθεί κάποια διαφορετική προσέγγιση (η HOT-SAX βελτιώνει αρκετά το χρόνο αλλά όχι ικανοποιητικά) για αυτήν την μέθοδο ίσως και να γίνει implemented σε κάποια άλλη γλώσσα που να μην απαιτεί τόσο χρόνο υλοποίησης. Η καταλληλότερη γλώσσα είναι η java για τέτοιου είδους υλοποιήσεις. Υπάρχει αρκετές υλοποιήσεις τα τελευταία χρόνια σε java και όλες ακολουθούν την προσέγγιση του GrammarViz. Βασίζονται δηλαδή στη συμβολική διακριτοποίηση, τη Γραμματική Εξαγωγή και την πολυπλοκότητα Kolmogorv. Διευκολύνει την ανακάλυψη μοτίβων μεταβλητού μήκους γραμμικού χρόνου γρηγορότερα από την ανακάλυψη των ανωμαλιών με χρήση HOT-SAX αλλά η ακρίβεια δεν είναι εγγυημένη.

7 *Βιβλιογραφία*

1. Fuad, Muhammad Marwan Muhammad, and Pierre-François Marteau. "Towards a faster symbolic aggregate approximation method." *arXiv preprint arXiv:1301.5871* (2013).

2. Lin, Jessica, et al. "A symbolic representation of time series, with implications for streaming algorithms." *Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery*. 2003.

3. Lin, Jessica, et al. "Experiencing SAX: a novel symbolic representation of time series." *Data Mining and knowledge discovery* 15.2 (2007): 107-144.

4. Kulahcioglu, Burcu, Serhan Ozdemir, and Bora Kumova. "Application of symbolic Piecewise Aggregate Approximation (PAA) analysis to ECG signals." *17th IASTED International conference on applied simulation and modelling*. Citeseer, 2008.

5. Senin, Pavel, et al. "Grammarviz 2.0: a tool for grammar-based pattern discovery in time series." *Joint European conference on machine learning and knowledge discovery in databases*. Springer, Berlin, Heidelberg, 2014.

6. Chen, Yunqiang, Xiang Sean Zhou, and Thomas S. Huang. "One-class SVM for learning in image retrieval." *Proceedings 2001 International Conference on Image Processing (Cat. No. 01CH37205)*. Vol. 1. IEEE, 2001.

7. Liu, Fei Tony, Kai Ming Ting, and Zhi-Hua Zhou. "Isolation forest." *2008 Eighth IEEE International Conference on Data Mining*. IEEE, 2008.

8. Steinley, Douglas. "K-means clustering: a half-century synthesis." *British Journal of Mathematical and Statistical Psychology* 59.1 (2006): 1-34.

9. Hamerly, Greg, and Charles Elkan. "Learning the k in k-means." *Advances in neural information processing systems*. 2004.

10. Chandola, Varun, Arindam Banerjee, and Vipin Kumar. "Anomaly detection: A survey." *ACM computing surveys (CSUR)* 41.3 (2009): 1-58.

11. Teng, Mingyan. "Anomaly detection on time series." *2010 IEEE International Conference on Progress in Informatics and Computing*. Vol. 1. IEEE, 2010.

12. Senin, P., Lin, J., Wang, X., Oates, T., Gandhi, S., Boedihardjo, A.P., Chen, C., Frankenstein, S., Lerner, M., GrammarViz 2.0: a tool for grammar-based pattern discovery in time series, ECML/PKDD, 2014.

8 *Αναφορές*

[1] «wikipedia,» [Ηλεκτρονικό]. Available: https://el.wikipedia.org/wiki/Τεχνητή_νοημοσύνη.

[2] «www.sas.com,» [Ηλεκτρονικό]. Available: https://www.sas.com/el_gr/insights/analytics/what-is-artificial-intelligence.html.

[3] «www.sas.com,» [Ηλεκτρονικό]. Available: https://www.sas.com/el_gr/insights/analytics/what-is-artificial-intelligence.html.

[4] «wikipedia,» [Ηλεκτρονικό]. Available: https://el.wikipedia.org/wiki/Μηχανική_μάθηση.

[5] «www.sas.com,» [Ηλεκτρονικό]. Available: https://www.sas.com/en_us/insights/analytics/machine-learning.html.

[6] «simplilearn,» [Ηλεκτρονικό]. Available: https://www.simplilearn.com/classification-machine-learning-tutorial.

[7] A. Dave, «medium,» 4 Δεκέμβριος 2018. [Ηλεκτρονικό]. Available: https://medium.com/datadriveninvestor/regression-in-machine-learning-296caae933ec.

[8] «datarobot,» [Ηλεκτρονικό]. Available: https://www.datarobot.com/wiki/prediction/.

[9] «wikipedia,» [Ηλεκτρονικό]. Available: https://en.wikipedia.org/wiki/Cluster_analysis.

[10] «pyts.readthedocs,» [Ηλεκτρονικό]. Available: https://pyts.readthedocs.io/en/latest/auto_examples/approximation/plot_sax.html.

[11] V. Krishnamoorthy, Ιούνιος 2018. [Ηλεκτρονικό]. Available: https://vigne.sh/posts/symbolic-aggregate-approximation/.

[12] Φεβρουάριος 2016. [Ηλεκτρονικό]. Available: https://www.researchgate.net/post/How_does_the_symbolic_aggregate_approximation_SAX_technique_work.

[13] 1 Μαρτης 2016. [Ηλεκτρονικό]. Available: https://www.researchgate.net/post/How_to_choose_the_alphabet_size_paa_size_and_windows_size_for_g

enerating_a_stream_using_SAX.

[14] P.-F. Marteau, Ιανουάριος 2010. [Ηλεκτρονικό]. Available: https://www.researchgate.net/publication/220738301_Towards_a_Faster_Symbolic_Aggregate_Approximation_Method.

[15] 9 Απρίλιος 2015. [Ηλεκτρονικό]. Available: https://www.quora.com/MLconf-2015-Seattle-How-does-the-symbolic-aggregate-approximation-SAX-technique-work.

[16] P. Chen, «Medium,» 11 Απρίλιος 2019. [Ηλεκτρονικό]. Available: https://medium.com/@peijin/using-sax-paa-to-understand-the-s-p500s-yearly-patterns-337750622e49.

[17] L. Chen, «towardsdatascience,» 7 Ιανουάριος 2019. [Ηλεκτρονικό]. Available: https://towardsdatascience.com/support-vector-machine-simply-explained-fee28eba5496.

[18] scikit, «scikit-learn,» [Ηλεκτρονικό]. Available: https://scikit-learn.org/stable/auto_examples/svm/plot_oneclass.html#sphx-glr-auto-examples-svm-plot-oneclass-py.

[19] U. Malik, «stackabuse,» [Ηλεκτρονικό]. Available: https://stackabuse.com/implementing-svm-and-kernel-svm-with-pythons-scikit-learn/.

[20] scikit, «scikit-learn,» [Ηλεκτρονικό]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.svm.OneClassSVM.html.

[21] scikit, «scikit-learn,» [Ηλεκτρονικό]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html.

[22] a. krishnan, «towardsdatascience,» 10 Φεβρουάριος 2019. [Ηλεκτρονικό]. Available: https://towardsdatascience.com/anomaly-detection-with-isolation-forest-visualization-23cd75c281e2.

[23] «scikit-learn,» [Ηλεκτρονικό]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.IsolationForest.html.

[24] E. Lewinson, «towardsdatascience,» [Ηλεκτρονικό]. Available: https://towardsdatascience.com/outlier-detection-with-isolation-forest-3d190448d45e. [Πρόσβαση 2 Ιούλιος 2018].

[25] A. Kalla, «medium,» 6 Μάρτιος 2019. [Ηλεκτρονικό]. Available: https://medium.com/data-science-and-learning/k-means-clustering-4a700d4a4720.

[26] C. Piech, 2013. [Ηλεκτρονικό]. Available: https://stanford.edu/~cpiech/cs221/handouts/kmeans.html.

[27] D. M. J. Garbade, «towardsdatascience,» 13 Σεπτέμβριος 2018. [Ηλεκτρονικό]. Available: https://towardsdatascience.com/understanding-k-means-clustering-in-machine-learning-6a6e67336aa1.

[28] «scikit-learn,» [Ηλεκτρονικό]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html.

[29] A. Zhang, «medium,» 28 Νοέμβριος 2019. [Ηλεκτρονικό]. Available: https://medium.com/swlh/anomaly-detection-with-z-score-pick-the-low-hanging-fruits-ccd5ccccaee9.

[30] S. Santoyo, «towardsdatascience,» 12 Σεμπτέμβριος 2017. [Ηλεκτρονικό]. Available: https://towardsdatascience.com/a-brief-overview-of-outlier-detection-techniques-1e0b2c19e561.

[31] «statisticshowto,» [Ηλεκτρονικό]. Available: https://www.statisticshowto.com/probability-and-statistics/z-

score/.

[32] T. Gushue, Δεκέμβριος 2015. [Ηλεκτρονικό]. Available: http://rstudio-pubs-static.s3.amazonaws.com/138181_b0d6feb591014214a5fed357ed8199f0.html.

[33] «scikit-learn,» [Ηλεκτρονικό]. Available: https://scikit-learn.org/stable/auto_examples/svm/plot_oneclass.html#sphx-glr-auto-examples-svm-plot-oneclass-py.

[34] «scikit-learn,» [Ηλεκτρονικό]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.svm.OneClassSVM.html.

[35] «scikit-learn,» [Ηλεκτρονικό]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html.

Παράρτημα Ι - Δημιουργία μοντέλων και επεξεργασία δεδομένων.

Βιβλιοθήκες python που χρησιμοποιήθηκαν:

- Pandas (διαχείριση των data set)
- Numpy (δημιουργία διανυσμάτων σε πίνακα και γενικότερη χρήση matrix και arrays)
- Sklearn (επεξεργασία δεδομένων, δημιουργία μοντέλων και υπολογισμός των σχετικών metrics)
- Matplotlib (δημιουργία των διαγραμμάτων)
- Saxpy (πακέτο για την μέθοδο Symbolic Aggregate aproXimation)

Δεδομένα που χρησιμοποιήθηκαν:

Τα δεδομένα προέρχονται από Federal Reserve Bank of St. Louis και είναι 3 data set στο σύνολο. Το ένα αναφέρεται σε πωλήσεις οχημάτων από το 1970 μέχρι τέλη του 2019. Το δέυτερο περιέχει τις τιμές σε δολάρια που είχε το βαρέλι για το πετρέλαιο (BRENT) από το 1990 έως το 2019. Το τρίτο data set είναι το ποσοστό των ανέργων από το 1950 περίπου έως το 2019.

Επεξεργασία των δεδομένων:

Όπου κρινόταν αναγκαίο για τους υπολογισμούς οι αριθμοί έγιναν float με την ανάλογη συνάρτηση της python και πολλές φορές, χρειάστηκε να γίνει και απομάκρυνση κάποιων ειδικών χαρακτήρων κυρίως από το Column του κάθε data set που περιέχουν την ημερομηνία (Date).

```
def clean(x):
    x = x.replace("/", "").replace("-", "")
    return float(x)

df.Date = df.Date.apply(clean)
```

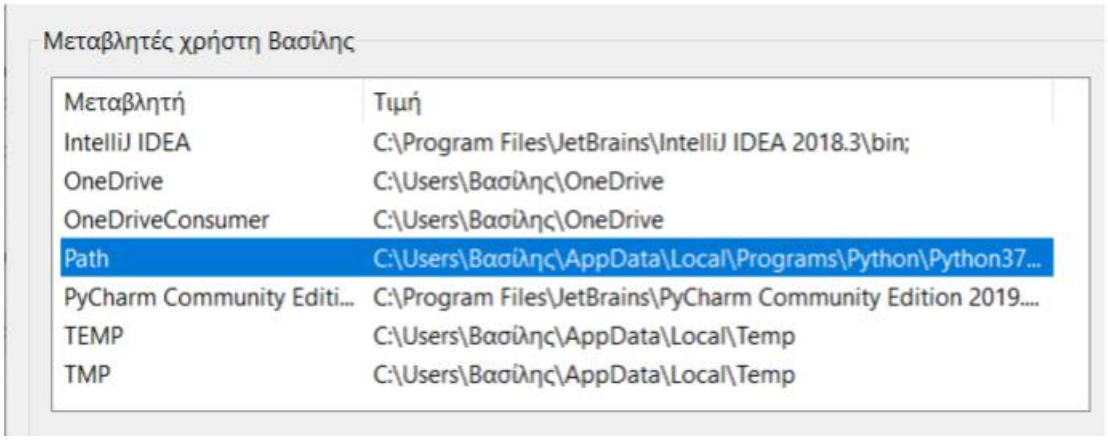
Εικόνα 58: special characters remove

Εγχειρίδιο Χρήσης της εφαρμογής

SYSTEM PREFACE

Πριν από όλα για να λειτουργήσει το project χρειάζεται να έχουμε κατεβάσει το java development kit (jdk1.8.0_151) και αν ο υπολογιστής μας τρέχει windows πρέπει να έχουμε κατεβάσει python (κατά προτίμηση version 3.7).

Προσοχή! Αν ο υπολογιστής δεν αναγνωρίζει την Python χρειάζεται να πάμε στις μεταβλητές περιβάλλοντος συστήματος από τον πίνακα ελέγχου και να προσθέσουμε το path που βρίσκεται η python σαν μεταβλητή.



Εικόνα 59: windows path

GENERAL INFORMATION OF SYSTEM

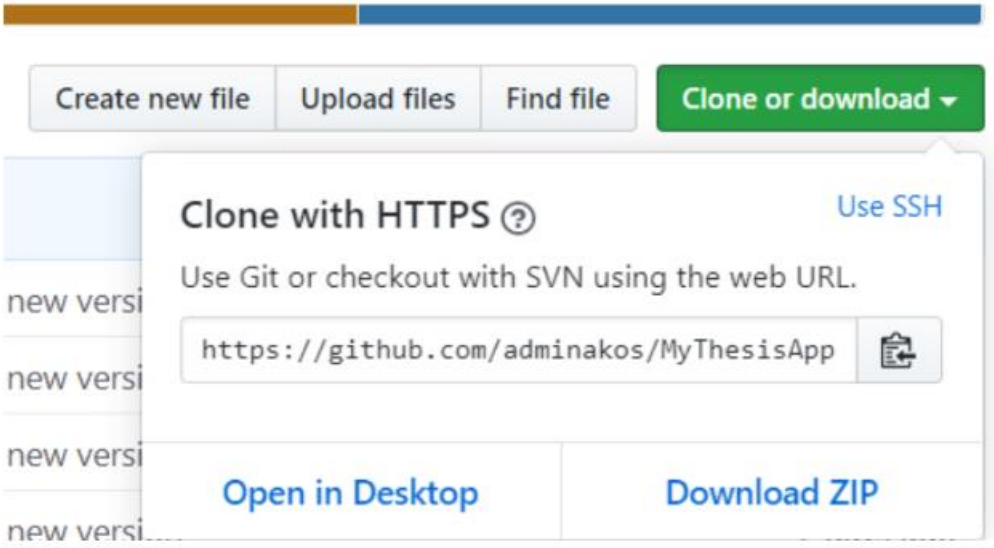
Η εφαρμογή είναι web και χρειάζεται να υπάρχει server για να λειτουργήσει. Στην συγκεκριμένη περίπτωση χρησιμοποιείται ο Apache Tomcat 9.0.30. Επίσης η εφαρμογή δημιουργήθηκε στο IntelliJ IDEA 2018.3 Ultimate Edition.

SOFTWARE INSTALLATION

- Πρέπει να γίνει εγκατάσταση του προγράμματος intelliJ IDEA 2018.3 (προτιμότερο) ή νεότερης έκδοσης. Το βρίσκεις εδώ: https://www.jetbrains.com/idea/download/#section=windows
- Πρέπει να γίνει εγκατάσταση του Apache Tomcat 9.0.30. Το βρίσκεις εδώ: https://tomcat.apache.org/download-90.cgi
- **Προσοχή!** Ο Φάκελος που θα γίνει η εγκατάσταση θα πρέπει να είναι ο ίδιος με αυτόν που αποθηκεύονται τα projects του intelliJ. Δηλαδή ο C:\Users* (όπου * το όνομα χρήστη σας.)

PROJECT

Κάνουμε download ή clone το project από το github. Για την επιλογή Download: Επισκεπτόμαστε την σελίδα https://github.com/adminakos/MyThesisApp και επιλέγουμε download ZIP όπως φαίνεται στη εικόνα.



Εικόνα 60: github

Κάνουμε unzip το project στον φάκελο C:\Users*\IdeaProjects (όπου * το όνομα χρήστη σας.) και έπειτα το ανοίγουμε (**open**) στο intelliJ. Αν το open project δεν λειτουργεί επιλέγουμε την επιλογή **import** διότι συνήθως με αυτόν τον τρόπο το intelliJ διορθώνει πιθανά σφάλματα και warnings από παραμετροποιήσεις που ίσως λείπουν.

Για την επιλογή **Clone**: Θα χρειαστεί να υπάρχει κατεβασμένο στον υπολογιστή το Git. Έπειτα η σειρά που πρέπει να ακολουθήσουμε είναι η εξής.

Πρέπει να είμαστε μέσα στο φάκελο που αποθηκεύονται τα project του intelliJ IDEA. Συνήθως ο φάκελος ονομάζεται IdeaProjects και βρίσκεται στη θέση C:\Users*\IdeaProjects (όπου * το όνομα χρήστη σας.)

- Στη συνέχεια Δεξί κλικ-> Git Bash Here
- Και μόλις ανοίξει το terminal η συνέχεια ειναι η παρακάτω.
- Να κάνουμε το Git global setup με τις παρακάτω εντολές
- git config --global user.name "username"
- git config --global user.email "email"

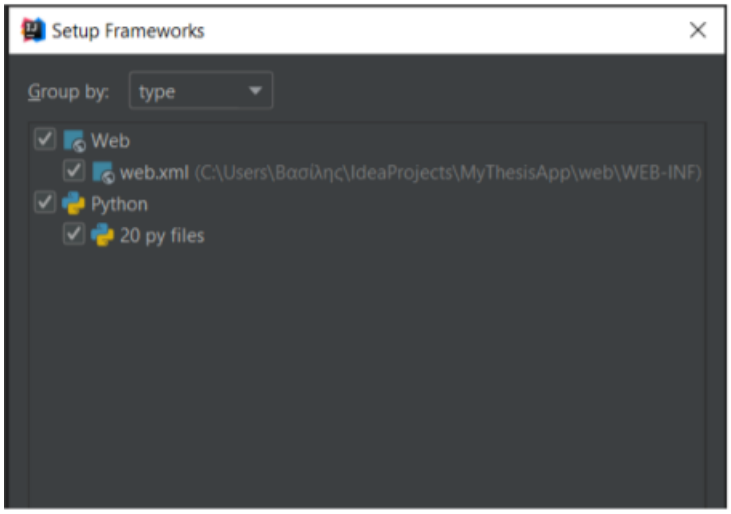
Στην ενότητα username μέσα στα " " θα εισάγεις το username του github λογαριασμού σου
Στην ενότητα email μέσα στα " " θα εισάγεις το email του github λογαριασμού σου.

- Κάνουμε Clone με την εντολή git clone https://github.com/adminakos/MyThesisApp.git

Το clone είναι προτιμότερο να γίνεται όταν θες να συνεισφέρεις στο project ως contributor.

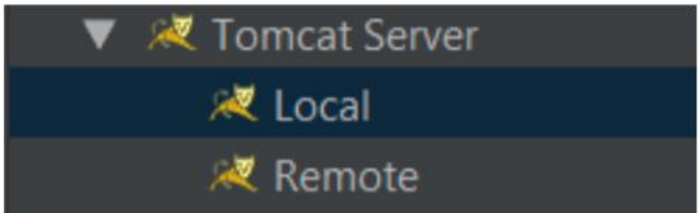
CONFIGURATION

Αφού γίνουν τα παραπάνω και το project ανοίξει κανονικά στο intelliJ, υπάρχει η πιθανότητα επειδή είναι servlet να θέλει τις κατάλληλες παραμετροποιήσεις. (Αναφέρεται ξανά ότι με την επιλογή import αυτό το βήμα ίσως είναι περιτό). Αρχικά επιλέγοντας **File->Project Structure-> SDKs** πρέπει στα SDKs του project να υπάρχουν το jdk1.8 και Python 3.7. θα καταλάβετε ότι λείπει η python όταν τα imports στα python scripts δεν αναγνωρίζονται και υπάρχει ένα error/warning για python interpreter.



Εικόνα 61: intelliJ set up frameworks

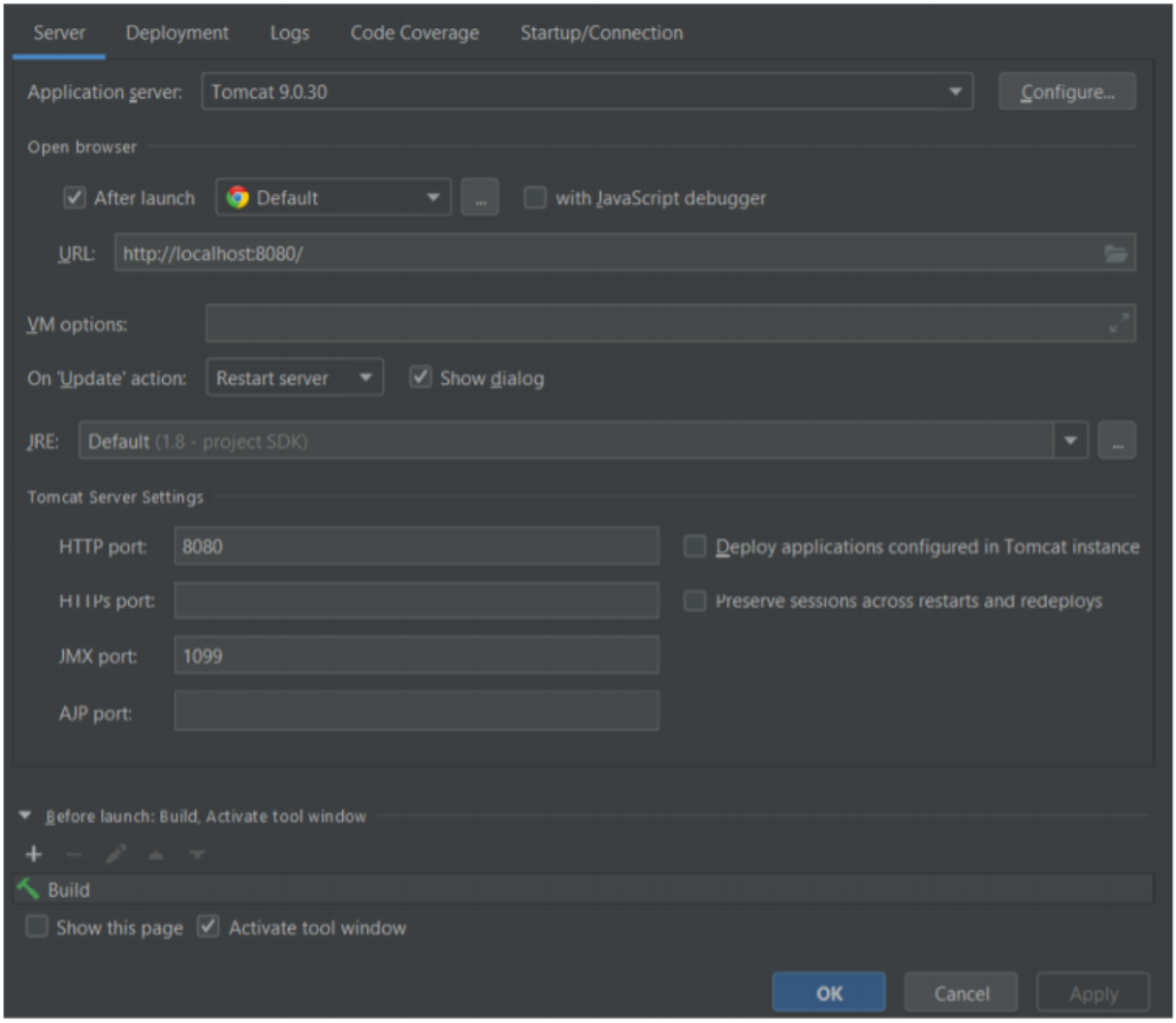
Επίσης πρέπει να παραμετροποιήσουμε το Project να τρέχει στον server (apache tomcat) γι' αυτό επιλέγετε δίπλα από το build πάνω δεξιά **Edit Configurations...->Templates->Tomcat Server->Local**



Εικόνα 62: intelliJ server selection

Παραμετροποιούμε όπως φαίνεται στην εικόνα παρακάτω.

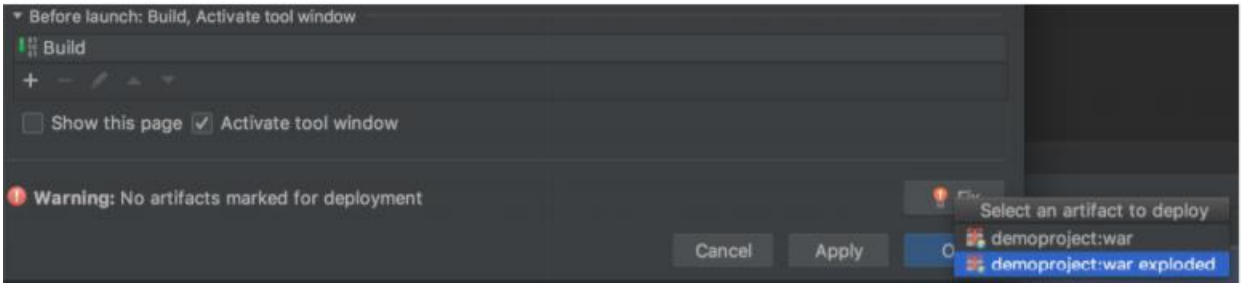
- **Application server:** Tomcat 9.0.30 (ή όποια άλλη version έχετε. Εδώ συνήθως το intelliJ βρίσκει το server μόνο του και είναι από default συμπληρωμένο. Αν δεν είναι ψάχνετε εκεί που αποθηκεύσατε τον Apache Tomcat-οδηγίες στην αρχή του manual)
- **URL:** http://localhost:8080/index.jsp (για να τρέχει η πρώτη μας σελίδα με το "τρέξιμο" του server)
- Στο **Open Browser** επιλέγεται το κουτάκι After Launch και απο το drop down list επιλέγετε Chrome.
- Σιγουρεύεστε ότι στην ενότητα **JRE** έχει το Default (1.8-project SDK).
- **HTTP Port:** 8080
- **JMX Port:** 1099



Εικόνα 63:intelliJ ports and app parameters

OTHER INFORMATION

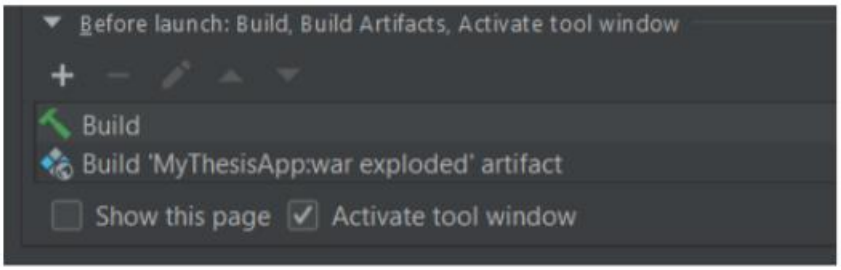
Στο κάτω μέρος του παραθύρου και πάνω περίπου από το Apply ίσως υπάρχει ένα **Warning: No artifacts marked for Deployment.** Και ένα κουμπί **Fix**.



Εικόνα 64: artifacts

Προσοχή! στο πάτημα του Fix έχουμε να επιλέξουμε μεταξύ δύο επιλογών. **MyThesisApp:war** και **MyThesisApp:war exploded**. Επιλέγουμε το δεύτερο (**exploded**).

Γενικότερα ακόμα και αν δεν υπάρχει το warning και το κουμπί Fix πρέπει το πεδίο αυτό να είναι όπως φαίνεται στη εικόνα.



Εικόνα 65: build artifact

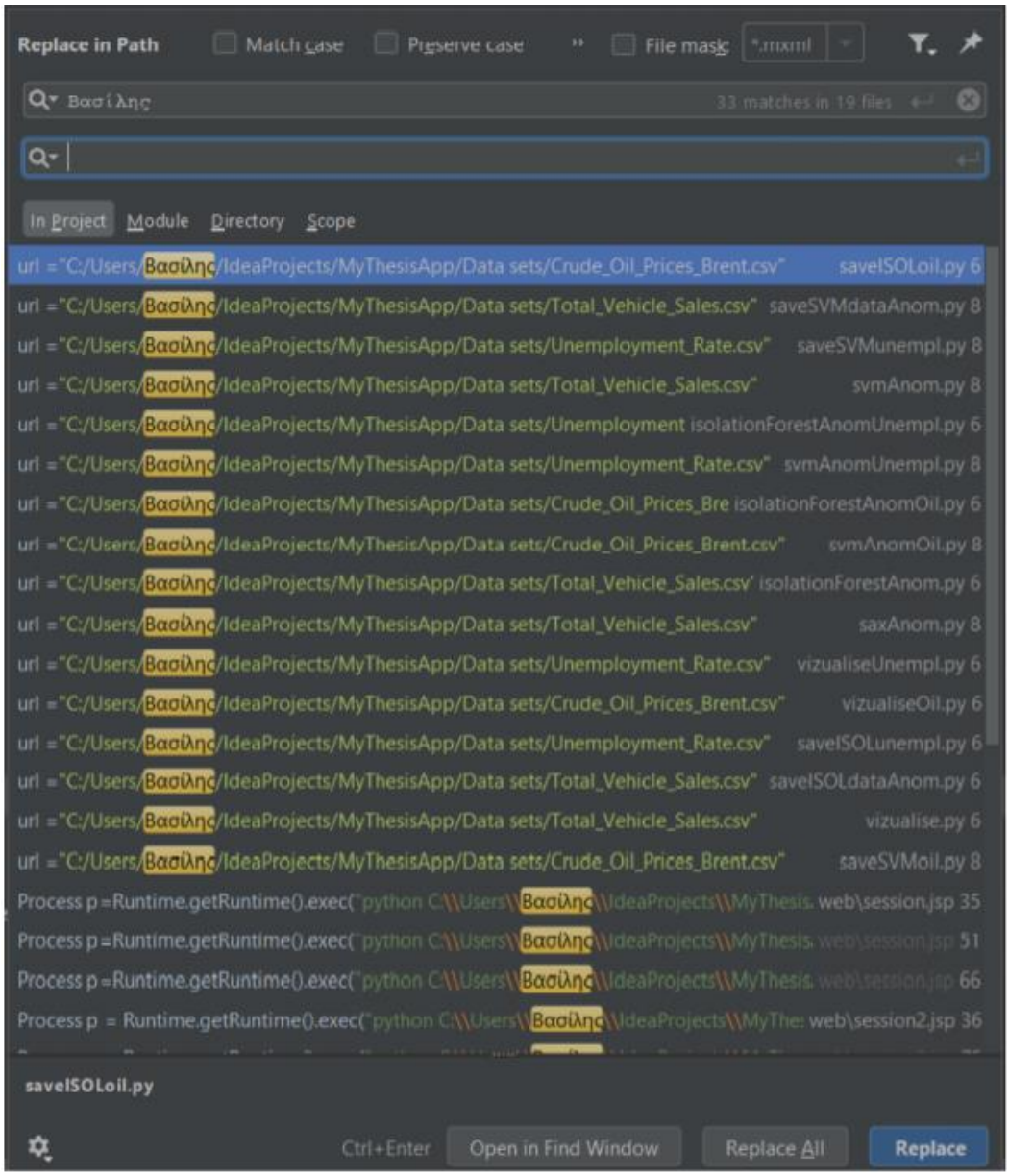
Αν δεν υπάρχει το κουμπί + και το προσθέτουμε από τη λίστα που εμφανίζεται. (εικόνα πάνω)

Τέλος επειδή μέσα στον κώδικα υπάρχουν απαραίτητα paths για τη λειτουργια του roject και επειδή αυτά αλλάζουν απο υπολογιστή σε υπολογιστή. Επιλέγουμε Ctrl + Shift + R και εμφανίζεται αυτό το παράθηρο. Στο πανω text area βάζουμε το ήδη υπάρχον κείμενο και στο κάτω βάζουμε αυτό με το οποίο θέλουμε να το αντικαταστήσουμε. Στη συγκεκριμένη περίπτωση στο πάνω θα γράψετε "Βασίλης" και στο κάτω το δικό σας όνομα χρήστη (του υπολογιστή). Με αυτόν τον τρόπο θα αλλάξουν όλα τα paths.

(Για παράδειγμα το script που χρησιμοποιείται για να κάνει οπτικοποίηση της χρονοσειράς από C:\Users\Βασίλης\IdeaProjects\MyThesisApp\vizualise.py θα μετατραπεί σε C:\Users\"το δικο σας ονομα χρήστη"\IdeaProjects\MyThesisApp\vizualise.py. Ετσι θα είναι εφικτό να τρέξει στον υπολογιστή σας.)

OTHER INFORMATION

Τέλος επειδή μέσα στον κώδικα υπάρχουν απαραίτητα paths για τη λειτουργία του roject και επειδή αυτά αλλάζουν από υπολογιστή σε υπολογιστή. Επιλέγουμε Ctrl + Shift + R και εμφανίζεται το παράθυρο που φαίνεται στην εικόνα.



Εικόνα 66: change path

Στο πάνω text area βάζουμε το ήδη υπάρχον κείμενο και στο κάτω βάζουμε αυτό με το οποίο θέλουμε να το αντικαταστήσουμε. Στη συγκεκριμένη περίπτωση στο πάνω θα γράψετε "Βασίλης" και στο κάτω το δικό σας όνομα χρήστη (του υπολογιστή) και επιλέγετε Replace All. Με αυτόν τον τρόπο θα αλλάξουν όλα τα paths.

***Για παράδειγμα το script που χρησιμοποιείται για να κάνει οπτικοποίηση της χρονοσειράς από C:\Users\Βασίλης\IdeaProjects\MyThesisApp\vizualise.py θα μετατραπεί σε C:\Users\"το δικό σας όνομα χρήστη"\IdeaProjects\MyThesisApp\vizualise.py. Έτσι θα είναι εφικτό να τρέξει στον υπολογιστή σας.**