ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

Αναγνώριση και διαχείριση ανωμαλιών Δεδομένων με χρήση ευφυών τεχνικών

Κακάτσος Δημήτριος

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΥΠΕΥΘΥΝΟΣ
Κολομβάτσος Κωσταντίνος
Επίκουρος Καθηγητής

Λαμία έτος 2023-24

ΠΑΝΕΠΙΣΤΗΜΙΟ
ΘΕΣΣΑΛΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

Αναγνώριση και διαχείριση ανωμαλιών Δεδομένων με χρήση ευφυών τεχνικών

Κακάτσος Δημήτριος

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΥΠΕΥΘΥΝΟΣ

Κολομβάτσος Κωσταντίνος
Επίκουρος Καθηγητής

Λαμία έτος 2023-24

UNIVERSITY OF
THESSALY

SCHOOL OF SCIENCE

DEPARTMENT OF COMPUTER SCIENCE & TELECOMMUNICATIONS

Outlier detection and management through machine learning

Kakatsos Dimitrios

FINAL THESIS

ADVISOR

Kolomvatsos Kostantinos
Assistant Professor

Lamia year 2023-24

«Με ατομική μου ευθύνη και γνωρίζοντας τις κυρώσεις [(1)], που προβλέπονται από της διατάξεις της παρ. 6 του άρθρου 22 του Ν. 1599/1986, δηλώνω ότι:

*1. Δεν παραθέτω κομμάτια βιβλίων ή άρθρων ή εργασιών άλλων αυτολεξεί **χωρίς να τα περικλείω σε εισαγωγικά** και χωρίς να αναφέρω το συγγραφέα, τη χρονολογία, τη σελίδα. Η αυτολεξεί παράθεση χωρίς εισαγωγικά χωρίς αναφορά στην πηγή, είναι λογοκλοπή. Πέραν της αυτολεξεί παράθεσης, λογοκλοπή θεωρείται και η παράφραση εδαφίων από έργα άλλων, συμπεριλαμβανομένων και έργων συμφοιτητών μου, καθώς και η παράθεση στοιχείων που άλλοι συνέλεξαν ή επεξεργάσθηκαν, χωρίς αναφορά στην πηγή. Αναφέρω πάντοτε με πληρότητα την πηγή κάτω από τον πίνακα ή σχέδιο, όπως στα παραθέματα.*

*2. Δέχομαι ότι η αυτολεξεί **παράθεση χωρίς εισαγωγικά**, ακόμα κι αν συνοδεύεται από αναφορά στην πηγή σε κάποιο άλλο σημείο του κειμένου ή στο τέλος του, είναι αντιγραφή. Η αναφορά στην πηγή στο τέλος π.χ. μιας παραγράφου ή μιας σελίδας, δεν δικαιολογεί συρραφή εδαφίων έργου άλλου συγγραφέα, έστω και παραφρασμένων, και παρουσίασή τους ως δική μου εργασία.*

3. Δέχομαι ότι υπάρχει επίσης περιορισμός στο μέγεθος και στη συχνότητα των παραθεμάτων που μπορώ να εντάξω στην εργασία μου εντός εισαγωγικών. Κάθε μεγάλο παράθεμα (π.χ. σε πίνακα ή πλαίσιο, κλπ), προϋποθέτει ειδικές ρυθμίσεις, και όταν δημοσιεύεται προϋποθέτει την άδεια του συγγραφέα ή του εκδότη. Το ίδιο και οι πίνακες και τα σχέδια

4. Δέχομαι όλες τις συνέπειες σε περίπτωση λογοκλοπής ή αντιγραφής.

Ημερομηνία: 05-03-2024

Ο – Η Δηλ.

Κακάτσος Δημήτριος

(1) «Όποιος εν γνώσει του δηλώνει ψευδή γεγονότα ή αρνείται ή αποκρύπτει τα αληθινά με έγγραφη υπεύθυνη δήλωση
του άρθρου 8 παρ. 4 Ν. 1599/1986 τιμωρείται με φυλάκιση τουλάχιστον τριών μηνών. Εάν ο υπαίτιος αυτών των πράξεων
σκόπευε να προσπορίσει στον εαυτόν του ή σε άλλον περιουσιακό όφελος βλάπτοντας τρίτον ή σκόπευε να βλάψει άλλον, τιμωρείται με κάθειρξη μέχρι 10 ετών.»

ΠΕΡΙΛΗΨΗ

Η εργασία επικεντρώνεται στο θέμα της αναγνώρισης και διαχείρισης ανωμαλιών στα δεδομένα με τη χρήση ευφυών τεχνικών. Αναλύει τη σημασία της ανίχνευσης ανωμαλιών (outliers) σε δεδομένα και παρουσιάζει εξελιγμένες τεχνικές ευφυούς αναγνώρισης, όπως η μηχανική μάθηση. Αρχικά, δημιουργούνται τυχαία data points αλλά με συγκεκριμένο πεδίο τιμών και στην συνέχεια χρησιμοποιούνται τεχνικές για την αναγνώριση ανωμαλιών σε κάθε μέγεθος δεδομένων. Ο κύριος αλγόριθμος που χρησιμοποιούμε για την διαχείριση των δεδομένων είναι το SVM (Support Vector Machine) ακολουθούμενο από βοηθητικές συναρτήσεις.
Με αυτά έχουμε ένα μοντέλο για την ανίχνευση ανωμαλιών για κάθε μέγεθος δεδομένων. Τέλος, το πρόγραμμα προβαίνει σε συνοπτική ανάλυση αυτών των τεχνικών καθώς και μια σύγκριση μεταξύ του SVM και του Isolation Forest ένας αλγόριθμος που χρησιμοποιείται για την αναγνώριση και διαχείριση δεδομένων.

ABSTRACT

The work focuses on the issue of identifying and managing anomalies in data using intelligent techniques. It discusses the importance of detecting anomalies (outliers) in data and presents sophisticated intelligent recognition techniques, such as machine learning. First, random data points are generated but with a specific range of values and then techniques are used to identify anomalies in each data size. The main algorithm we use to manage the data is SVM (Support Vector Machine) followed by helper functions. With these we have a model for detecting anomalies for any size of data. Finally, the program provides a brief analysis of these techniques as well as a comparison between SVM and Isolation Forest, an algorithm used for data recognition and management.

Table of Contents

ΠΕΡΙΛΗΨΗ I
ABSTRACT III

ΚΕΦΑΛΑΙΟ 1 ΕΙΣΑΓΩΓΗ 2

ΤΟ ΠΡΟΒΛΗΜΑ 2
ΣΤΟΧΟΣ 2
W MOTIVATION (INLIERS-OUTLIERS) 3
1. ADAPTABILITY TO CHANGING DATA 3
2. EFFICIENT MEMORY USAGE 3
3. REAL-TIME ADJUSTMENT 3
4. THRESHOLD TUNING 4

ΚΕΦΑΛΑΙΟ 2 ΜΗΧΑΝΙΚΗ ΜΑΘΗΣΗ 5

ΟΡΙΣΜΟΣ 5
ΤΥΠΟΙ 6
ΙΣΤΟΡΙΑ 7

ΚΕΦΑΛΑΙΟ 3 ΑΝΑΓΝΩΡΙΣΗ ΑΝΩΜΑΛΙΩΝ ΣΕ ΔΕΔΟΜΕΝΑ 9

ΑΝΩΜΑΛΙΕΣ ΣΕ ΔΕΔΟΜΕΝΑ 9
ΑΙΤΙΕΣ 10
ΑΝΑΓΝΩΡΙΣΗ-ΤΕΧΝΙΚΕΣ 10
ΔΙΑΧΕΙΡΙΣΗ-ΦΙΛΤΡΑΡΙΣΜΑ 12

ΚΕΦΑΛΑΙΟ 4 ΜΟΝΤΕΛΟ 14

ΤΙ ΕΙΝΑΙ ΤΟ ONE CLASS SVM 14
ΓΙΑΤΙ ΧΡΗΣΙΜΟΠΟΙΗΘΗΚΕ 14
ΥΛΟΠΟΙΗΣΗ 15
ΣΥΝΑΡΤΗΣΕΙΣ 17
ΣΥΜΠΕΡΑΣΜΑ 23
ISOLATION FOREST 23
ΓΙΑΤΙ ΧΡΗΣΙΜΟΠΟΙΗΘΗΚΕ 24
ΥΛΟΠΟΙΗΣΗ 24

ΚΕΦΑΛΑΙΟ 5 ΠΕΙΡΑΜΑΤΑ 26

ΕΙΣΑΓΩΓΗ 26
ΠΕΙΡΑΜΑ #1 27
ΠΕΙΡΑΜΑ #2 32

ΠΕΙΡΑΜΑ #3 36
ΠΕΙΡΑΜΑ #4 40
ΠΕΙΡΑΜΑ #5 44
ΠΕΙΡΑΜΑ #6 48

ΚΕΦΑΛΑΙΟ 6 : ΣΥΜΠΕΡΑΣΜΑΤΑ 52

ΑΠΟΤΕΛΕΣΜΑΤΑ & ΤΕΛΙΚΟ ΣΥΜΠΕΡΑΣΜΑ 52
ΜΕΛΛΟΝΤΙΚΕΣ ΕΠΕΚΤΑΣΕΙΣ 54

ΒΙΒΛΙΟΓΡΑΦΙΑ 57

ΚΕΦΑΛΑΙΟ 1 Εισαγωγή

Το πρόβλημα

Η αναγνώριση και διαχείριση ανωμαλιών στα δεδομένα αποτελούν κρίσιμο παράγοντα σε ποικίλους τομείς, από την ασφάλεια δεδομένων έως την υγεία και την οικονομία. Η ανάγκη για αποτελεσματικά εργαλεία που να μπορούν να αντιληφθούν και να αντιμετωπίσουν ανωμαλίες είναι επιτακτική. Σε αυτό το πλαίσιο, η παρούσα έρευνα επικεντρώνεται στη χρήση του αλγορίθμου SVM(Support Vector Machine) ως ένα προηγμένο εργαλείο για την αναγνώριση και διαχείριση ανωμαλιών σε πραγματικό χρόνο. Αναλύουμε το πρόβλημα, θέτουμε τους στόχους μας και εξετάζουμε τη σημασία αυτής της έρευνας για τη βελτίωση της αξιοπιστίας και της αποτελεσματικότητας στην αναγνώριση ανωμαλιών στα δεδομένα.

Στο πλαίσιο αυτό, η εργασία αναλύει το πρόβλημα της ανίχνευσης ανωμαλιών, καθορίζει τους στόχους της έρευνας και εξετάζει τη σημασία της χρήσης του αλγορίθμου SVM για την επίλυση αυτού του προβλήματος.
Ένας καθοριστικός παράγοντας αυτής της έρευνας είναι η δημιουργία των δεδομένων, όπου το πρόγραμμα αναλαμβάνει δυναμικά τη δημιουργία σημείων δεδομένων με βάση low και high intervals. Μέσω της χρήσης αυτοματοποιημένων διαδικασιών δημιουργίας δεδομένων, επιδιώκουμε να αναπτύξουμε ένα προηγμένο σύστημα που θα είναι σε θέση να προσομοιώνει ποικίλες συνθήκες και να δημιουργεί δεδομένα που καλύπτουν διάφορα εύρη τιμών. Αυτή η προσέγγιση δίνει στην έρευνα τη δυνατότητα να δοκιμάσει και να αξιολογήσει την απόδοση του αλγορίθμου σε πραγματικές συνθήκες, παρέχοντας χρήσιμες πληροφορίες και εφαρμογές σε πραγματικά προβλήματα αναγνώρισης ανωμαλιών.

ΣΤΟΧΟΣ

Ο στόχος όλου του προγράμματος είναι να αναπτυχθεί ένας αποτελεσματικός αλγόριθμος για την αναγνώριση ανωμαλιών δεδομένων ικανός στο να ξεχωρίζει δεδομένα και να γίνεται η σωστή διαχείριση των ανωμαλιών αυτών των δεδομένων. Το πρόβλημα της αναγνώρισης ανωμαλιών είναι ένα κρίσιμο ζητούμενο που το οποίο εμφανίζεται σε πολλούς τομείς για παράδειγμα στο cybersecurity, στην οικονομία, στην υγεία και σε βιομηχανικά συστήματα. Επι της ουσίας ο στόχος είναι η δημιουργία ενός ευέλικτου και προσαρμοστικού αλγορίθμου που μπορεί να χειριστεί διαφορά σύνολα δεδομένων με εξελισσόμενα μοτίβα στα δεδομένα.

MOTIVATION (INLIERS-OUTLIERS)

Σε αυτή την ενότητα θα αναφέρουμε μια σημαντική λειτουργία του προγράμματος και το motivation πίσω από αυτό. Αρχικά, στο πρόγραμμα χρησιμοποιείται μια παράμετρο η οποία καθορίζει το μέγεθος των δεδομένων που έχουμε να επιλέξουμε ως outliers. Αυτή η παράμετρος την έχουμε ονομάσει "w" στην ουσία χρησιμοποιείται για το outlier buffering. Το buffer λειτουργεί στην προσωρινή αποθήκευση των ανωμαλιών δεδομένων, επιτρέποντας για δυναμικές προσαρμογές που πρέπει να γινούν στα όρια του φίλτρου (filter intervals) με βάση τα χαρακτηριστικά των ανιχνευόμενων ανωμαλιών. Αυτή η δυναμική προσαρμογή συμβάλλει στην προσαρμοστικότητα του αλγορίθμου αναγνώρισης δεδομένων.

Το motivation με τη χρήση ενός outlier buffer με την παράμετρο w είναι να επιτευχθούν κάποιοι στόχοι. Ο κάθε στόχος έχει την ιδιαιτερότητα του μέσα στο πρόγραμμα γι' αυτό είναι καλό να τους αναλύσουμε.

1. ADAPTABILITY TO CHANGING DATA

Το buffer επιτρέπει στον αλγόριθμο να προσαρμοστεί σε αλλαγές των χαρακτηριστικών των outliers με το πέρασμα του χρόνου. Αυτό γίνεται με το να αποθηκεύονται προσωρινά οι outliers και να ρυθμίζονται τα όρια του φίλτρου ανάλογα, ο αλγόριθμος μπορεί να ανταποκριθεί σε αλλαγές στη διανομή δεδομένων και να διατηρήσει την ακριβή αναγνώριση ανωμαλιών.

2. EFFICIENT MEMORY USAGE

Το μέγεθος του buffer βοηθά στην αποτελεσματική διαχείριση των πόρων μνήμης. Περιορίζοντας τον αριθμό των αποθηκευμένων outliers ο αλγόριθμος εξισορροπεί την ανάγκη για την διατήρηση πληροφοριών σχετικά με ανωμαλίες με περιορισμούς της διαθέσιμης μνήμης.

3. REAL-TIME ADJUSTMENT

Καθώς ανιχνεύονται νέοι outliers και προστίθονται στο buffer τα όρια του φίλτρου προσαρμόζονται δυναμικά με βάση τα χαρακτηριστικά των outliers. Αυτή η προσαρμογή γίνεται σε πραγματικό χρόνο και επιτρέπει στον αλγόριθμο να βελτιώνει συνεχώς την κατανόηση του για τα δεδομένα και να βελτιώνει την ακρίβεια της αναγνώρισης ανωμαλιών.

4. THRESHOLD TUNING

Το buffer σε συνδυασμό με τη ρύθμιση κατωφλίου, εξασφαλίζει ότι ο αλγόριθμος μπορεί να διαφοροποιήσει αποτελεσματικά τα κανονικά δεδομένα με τους outliers. Το threshold προσαρμόζεται δυναμικά με βάση των outliers που είναι αποθηκευμένοι στο buffer, συμβάλλοντας στην ικανότητα του αλγορίθμου να λαμβάνει σωστές αποφάσεις σχετικά με τα data points.

Συνοπτικά, το buffer με την παράμετρο w παίζει σημαντικό ρολό στην ενίσχυση της προσαρμοστικότητας, της αποδοτικότητας της μνήμης και της απόκρισης σε πραγματικό χρόνο του αλγορίθμου αναγνώρισης ανωμαλιών. Επιτρέπει στον αλγόριθμο να προσαρμόζει δυναμικά τις παραμέτρους του με βάση τα χαρακτηριστικά των outliers, συμβάλλοντας στην συνολική αποτελεσματικότητα του στην αναγνώριση και τη διαχείριση ανωμαλιών σε σύνολα δεδομενων.

ΚΕΦΑΛΑΙΟ 2 Μηχανική Μάθηση

Ορισμός

Η μηχανική μάθηση είναι υποπεδίο της επιστήμης των υπολογιστών, που αναπτύχθηκε από τη μελέτη της αναγνώρισης προτύπων και της υπολογιστικής θεωρίας μάθησης στην τεχνητή νοημοσύνη. Το 1959, ο Άρθουρ Σάμουελ ορίζει τη μηχανική μάθηση ως "Πεδίο μελέτης που δίνει στους υπολογιστές την ικανότητα να μαθαίνουν, χωρίς να έχουν ρητά προγραμματιστεί". Η μηχανική μάθηση διερευνά τη μελέτη και την κατασκευή αλγορίθμων που μπορούν να μαθαίνουν από τα δεδομένα και να κάνουν προβλέψεις σχετικά με αυτά. 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 μηχανική μάθηση είναι στενά συνδεδεμένη και συχνά συγχέεται με υπολογιστική στατιστική, ένας κλάδος, που επίσης επικεντρώνεται στην πρόβλεψη μέσω της χρήσης των υπολογιστών. Έχει ισχυρούς δεσμούς με την μαθηματική βελτιστοποίηση, η οποία παρέχει μεθόδους, τη θεωρία και τομείς εφαρμογής. Η Μηχανική μάθηση εφαρμόζεται σε μια σειρά από υπολογιστικές εργασίες, όπου τόσο ο σχεδιασμός όσο και ο ρητός προγραμματισμός των αλγορίθμων είναι ανέφικτος. Παραδείγματα εφαρμογών αποτελούν τα φίλτρα spam (spam filtering), η οπτική αναγνώριση χαρακτήρων (OCR), οι μηχανές αναζήτησης και η υπολογιστική όραση. Η Μηχανική μάθηση μερικές φορές συγχέεται με την εξόρυξη δεδομένων, όπου η τελευταία επικεντρώνεται περισσότερο στην εξερευνητική ανάλυση των δεδομένων, γνωστή και ως μη επιτηρούμενη μάθηση.

Στο πεδίο της ανάλυσης δεδομένων, η μηχανική μάθηση είναι μια μέθοδος που χρησιμοποιείται για την επινόηση πολύπλοκων μοντέλων και αλγορίθμων που οδηγούν στην πρόβλεψη. Τα αναλυτικά μοντέλα επιτρέπουν στους ερευνητές, τους επιστήμονες δεδομένων, τους μηχανικούς και τους αναλυτές να παράγουν αξιόπιστες αποφάσεις και αποτελέσματα και να αναδείξουν αλληλοσυσχετίσεις μέσω της μάθησης από ιστορικές σχέσεις και τάσεις στα δεδομένα.

Ο Tom M. Mitchell πρότεινε έναν πιο επίσημο ορισμό που χρησιμοποιείται ευρέως: «Ένα πρόγραμμα υπολογιστή λέγεται ότι μαθαίνει από εμπειρία E ως προς μια κλάση εργασιών T και ένα μέτρο επίδοσης P, αν η επίδοσή του σε εργασίες της κλάσης T, όπως αποτιμάται από το μέτρο P, βελτιώνεται με την εμπειρία E». Αυτός ο ορισμός είναι σημαντικός για τον

καθορισμό της μηχανικής μάθησης σε βασικό λειτουργικό πλαίσιο παρά με γνωστικούς όρους, ακολουθώντας έτσι την πρόταση του Alan Turing στην εργασία του «Υπολογιστικές μηχανές και Νοημοσύνη», ότι το ερώτημα αν μπορούν οι μηχανές να σκεφτούν, μπορεί να αντικατασταθεί με το ερώτημα αν μπορούν οι μηχανές να κάνουν αυτό που εμείς (ως σκεπτόμενες οντότητες) μπορούμε να κάνουμε.

ΤΥΠΟΙ

Η μηχανική μάθηση έχει διαφορετικούς τύπους μοντέλων μηχανικής μάθησης, χρησιμοποιώντας διάφορες αλγοριθμικές τεχνικές. Ανάλογα με τη φύση των δεδομένων και το αποτέλεσμα που επιζητούμε, μπορεί να χρησιμοποιηθεί ένα από τα τέσσερα μοντέλα μάθησης.

Οι βασικοί τύποι μηχανικής μάθησης περιλαμβάνουν:

Εποπτευόμενη μάθηση (supervised learning): Το μοντέλο εκπαιδεύεται σε ένα σύνολο δεδομένων που περιλαμβάνει είσοδο και την αντίστοιχη επιθυμητή έξοδο. Σκοπός είναι να προβλέπει την έξοδο για νέες εισόδους. Το μοντέλο εκπαιδεύεται σε ένα σύνολο δεδομένων που περιλαμβάνει είσοδο και την αντίστοιχη επιθυμητή έξοδο. Σκοπός είναι να προβλέπει την έξοδο για νέες εισόδους.

Μη εποπτευόμενη μάθηση (unsupervised learning): Το μοντέλο εκπαιδεύεται σε ένα σύνολο δεδομένων χωρίς επιθυμητές εξόδους. Ο στόχος είναι να ανακαλύψει δομές ή πρότυπα στα δεδομένα.

Ενισχυτική μάθηση (reinforcement learning): Το μοντέλο εκπαιδεύεται να λαμβάνει αποφάσεις σε ένα περιβάλλον με σκοπό να μεγιστοποιήσει μια ανταμοιβή. Επιτυγχάνει αυτό μέσω δοκιμών και λαθών.

Ωστόσο, υπάρχει και μια τέταρτη κατηγορία που ορισμένοι συμπεριλαμβάνουν, **Ημι εποπτευόμενη μάθηση (semi supervised learning):**

Σε αυτήν την περίπτωση, το μοντέλο εκπαιδεύεται τόσο σε ετικετασμένα όσο και σε μη ετικετασμένα δεδομένα. Αυτό μπορεί να είναι χρήσιμο όταν τα ετικετασμένα δεδομένα είναι περιορισμένα, ενώ τα μη ετικετασμένα είναι περισσότερα και εύκολο να συλλεχθούν.

Ο τρόπος που κατατάσσονται οι τύποι της μηχανικής μάθησης είναι θέμα συζήτησης και η ταξινόμηση μπορεί να διαφέρει ανάλογα με το ποιον

ειδικότερο τομέα της μηχανικής μάθησης κοιτάζετε. Συνολικά, η μηχανική μάθηση αποτελεί έναν σημαντικό πυλώνα στην εξέλιξη της τεχνολογίας και έχει εφαρμογές σε πολλούς τομείς, όπως η υγεία, ο χρηματοοικονομικός τομέας, η ρομποτική, και πολλοί άλλοι.

ΙΣΤΟΡΙΑ

Ως επιστημονικό εγχείρημα, η μηχανική μάθηση αναπτύχθηκε από την αναζήτηση για την τεχνητή νοημοσύνη. Ήδη από την πρώιμη περίοδο της έρευνας στον τομέα της τεχνητής νοημοσύνης σε ακαδημαϊκό επίπεδο, το ζήτημα της κατασκευής μηχανών που θα μάθαιναν από δεδομένα απασχόλησε τους ερευνητές. Προσπάθησαν να προσεγγίσουν το πρόβλημα με διάφορες συμβολικές μεθόδους, καθώς και με τα λεγόμενα νευρωνικά δίκτυα. Αυτά ήταν κυριώς perceptrons και μοντέλα, που όπως διαπιστώθηκε αργότερα ήταν επανεφευρέσεις των γενικευμένων γραμμικών μοντέλων της στατιστικής. Επίσης χρησιμοποιήθηκε η πιθανοθεωρητική λογική, ιδιαίτερα στην αυτοματοποιημένη ιατρική διάγνωση.

Ωστόσο, μια αυξανόμενη έμφαση σε προσεγγίσεις που βασίζονται στην λογική γνώση προκάλεσε ένα ρήγμα μεταξύ Τεχνητής Νοημοσύνης και μηχανικής μάθησης. Τα πιθανοθεωρητικά συστήματα μαστίζονταν από θεωρητικά και πρακτικά προβλήματα απόκτησης δεδομένων και αναπαράστασής τους. Από το 1980, έμπειρα συστήματα επικράτησαν στο πεδίο της Τεχνητής Νοημοσύνης (ΤΝ), και ο ρόλος της στατιστικής υποχώρησε. Η εργασία σε συμβολική/βασισμένη σε γνώση εκμάθηση συνεχίστηκε εντός της ΤΝ, οδηγώντας στον επαγωγικό λογικό προγραμματισμό, αλλά οι κατευθηντήριες γραμμές της στατιστικής ήταν τώρα έξω από το χώρο της τεχνητής νοημοσύνης, στην αναγνώριση προτύπων και στην ανάκτηση πληροφοριών. Η έρευνα για νευρωνικά δίκτυα εγκαταλείφθηκε από την ΤΝ και την Επιστήμη Υπολογιστών τον ίδιο περίπου καιρό. Η ίδια επίσης κατεύθυνση ακολουθήθηκε πέρα από την ΤΝ και την πληροφορική, από ερευνητές άλλων ειδικοτήτων, συμπεριλαμβανομένων των Hopfield, Rumelhart και Hinton. Η επιτυχία ήρθε στα μέσα της δεκαετίας του 1980 με την επανεφεύρεση της μεθόδου ανάστροφης μετάδοσης (backpropagation).

Η Μηχανική μάθηση, αναδιοργανώθηκε ως ένα ξεχωριστό πεδίο, που άρχισε να ακμάζει κατά τη δεκαετία του 1990. Η προσοχή μετατοπίστηκε από τις συμβολικές προσεγγίσεις που κληρονόμησε από την Τεχνητή Νοημοσύνη, που στόχο είχαν την αντιμετώπιση επιλύσιμων προβλημάτων πρακτικής φύσης, και δόθηκε έμφαση σε μεθόδους και μοντέλα της στατιστικής και της θεωρίας πιθανοτήτων. Επίσης επωφελήθηκε από την διαθεσιμότητα

ψηφιοποιημένων πληροφοριών και της δυνατότητας να διανεμηθούν μέσω του Διαδικτύου.

ΚΕΦΑΛΑΙΟ 3 ΑΝΑΓΝΩΡΙΣΗ ΑΝΩΜΑΛΙΩΝ ΣΕ ΔΕΔΟΜΕΝΑ

ΑΝΩΜΑΛΙΕΣ ΣΕ ΔΕΔΟΜΕΝΑ

Σε αυτή την ενότητα θα μελετήσουμε την έννοια των ανωμαλιών σε δεδομένα ή αλλιώς outliers. Ένας outlier είναι ένα σημείο δεδομένων το οποίο είναι διαφορετικό από τα άλλα δεδομένα. Αυτό μπορεί να οφείλεται σε μεταβλητότητα στη μέτρηση ή μπορεί να είναι αποτέλεσμα πειραματικού σφάλματος. Ένας outlier μπορεί να είναι ένδειξη πιθανότητας, αλλά μπορεί επίσης να προκαλέσει σοβαρά προβλήματα στις στατιστικές αναλύσεις.

Οι outliers μπορούν να εμφανιστούν τυχαία σε οποιαδήποτε κατανομή, αλλά μπορούν να υποδεικνύουν νέα συμπεριφορά ή δομές στο σύνολο δεδομένων, σφάλμα μέτρησης ή ότι ο πληθυσμός έχει μια ``βαριά`` κατανομή(heavy-tailed distribution). Όταν υπάρχει σφάλμα μέτρησης κάποιος μπορεί να τα απορρίψει ή να χρησιμοποιήσει στατιστικά στοιχεία που είναι αξιόπιστα σε outliers, ενώ στην περίπτωση κατανομών ``βαριάς`` ουράς, υποδεικνύεται ότι η κατανομή έχει υψηλότερη κλίση και ότι πρέπει να γίνεται προσεκτικά η χρήση εργαλείων που αναλαμβάνουν μια κανονική κατανομή. Μια συχνή αιτία των outliers είναι ένα μείγμα δυο κατανομών που μπορεί να δηλώνουν «σωστή δοκιμή» έναντι «λάθους μέτρησης». Αυτό διαμορφώνεται από ένα μοντέλο.

Στα περισσότερα σετ δεδομένων, ορισμένα σημεία θα είναι πιο μακριά από τη μέση τιμή του δείγματος από αυτό που κρίνεται λογικό. Αυτό μπορεί να οφείλεται σε σφάλμα του συστήματος ή κάποιο ελάττωμα στην θεωρία της κατανομής και τον τρόπο δημιουργίας των δεδομένων. Ως εκ τούτου τα outliers μπορούν να υποδεικνύουν ελαττώματα στα δεδομένα, λανθασμένες διαδικασίες ή περιοχές όπου η συγκεκριμένη θεωρία να μην είναι έγκυρη. Ωστόσο σε μεγάλα δείγματα δεδομένων είναι αναμενόμενος ένας μικρός αριθμός outliers και όχι λόγω κάποιας ανωμαλίας στα δεδομένα.

Τα outliers λοιπόν μπορεί να περιλαμβάνουν το μέγιστο ή το ελάχιστο δείγμα ή και τα δυο ανάλογα με το αν είναι εξαιρετικά χαμηλά ή υψηλά (low,high). Ωστόσο το μέγιστο και το ελάχιστο δείγμα δεν είναι πάντα outlier, επειδή μπορεί να μην απέχουν ασυνήθιστα από την κατανομή των δεδομένων.

Οι outliers μπορεί να έχουν πολλές αιτίες. Μια φυσική συσκευή για τη λήψη μετρήσεων μπορεί να έχει υποστεί παροδική δυσλειτουργία. Ενδέχεται να υπήρξε σφάλμα στη μετάδοση ή τη μεταγραφή δεδομένων. Οι outliers προκύπτουν λόγω αλλαγών στη συμπεριφορά του συστήματος, δόλιας συμπεριφοράς, ανθρώπινου λάθους, σφάλματος οργάνου ή απλώς λόγω φυσικών αποκλίσεων στους πληθυσμούς. Ένα δείγμα μπορεί να έχει μολυνθεί με στοιχεία εκτός του υπό εξέταση πληθυσμού. Εναλλακτικά, ένας outlier θα μπορούσε να είναι το αποτέλεσμα ενός ελαττώματος στην υπόθεση θεωρίας, που απαιτεί περαιτέρω διερεύνηση από τον ερευνητή. Επιπλέον, η παθολογική εμφάνιση των outliers μιας συγκεκριμένης μορφής εμφανίζεται σε μια ποικιλία συνόλων δεδομένων, υποδεικνύοντας ότι ο αιτιολογικός μηχανισμός για τα δεδομένα μπορεί να διαφέρει στο ακραίο άκρο.

Δεν υπάρχει άκαμπτος μαθηματικός ορισμός του τι συνιστά έναν outlier. Ο προσδιορισμός του εάν μια παρατήρηση είναι outlier ή όχι είναι τελικά μια υποκειμενική άσκηση. Υπάρχουν διάφορες μέθοδοι ανίχνευσης των outliers, μερικές από τις οποίες αντιμετωπίζονται ως συνώνυμες με την ανίχνευση κάτι καινούργιου. Ορισμένες είναι γραφικές, όπως κανονικές γραφικές παραστάσεις πιθανοτήτων. Άλλες βασίζονται σε μοντέλα.

Οι μέθοδοι που βασίζονται σε μοντέλα που χρησιμοποιούνται συνήθως για την αναγνώριση υποθέτουν ότι τα δεδομένα προέρχονται από μια κανονική κατανομή και προσδιορίζουν παρατηρήσεις που θεωρούνται "απίθανες" με βάση τον μέσο όρο και την τυπική απόκλιση.

Εμείς θα αναφερθούμε στην αναγνώριση των δεδομένων και πως από την αναγνώριση θα διαχειριστούμε τα δεδομένα. Πρέπει βέβαια να σημειωθεί ότι η επιλογή του τρόπου αντιμετώπισης των outliers θα πρέπει να εξαρτάται από την αιτία.

Ακόμη και όταν ένα μοντέλο κανονικής κατανομής είναι κατάλληλο για τα δεδομένα που αναλύονται, αναμένονται outliers για μεγάλα μεγέθη δείγματος και δεν θα πρέπει να απορρίπτονται αυτόματα, εάν συμβαίνει αυτό. Αντίθετα, θα πρέπει κανείς να χρησιμοποιήσει μια μέθοδο που είναι εύρωστη για outliers για να μοντελοποιήσει ή να αναλύσει δεδομένα με φυσικά outliers.

Αναγνώριση δεδομένων καλείται η αναγνώριση προτύπων από ένα σύνολο δεδομένων που εμφανίζουν διαφορετική συμπεριφορά από την προσδοκώμενη. Άλλες ονομασίες που συναντιούνται στην βιβλιογραφία για τα πρότυπα είναι ακραίες τιμές, ασύμφωνες παρατηρήσεις, εξαιρέσεις, παρεκκλίσεις, ιδιαιτερότητες και προσμίξεις σε διαφορετικούς τομείς εφαρμογών. Στόχος είναι η υψηλού επιπέδου ανίχνευση πιθανών ανωμαλιών, διατηρώντας όμως χαμηλά ποσοστά λανθασμένης προειδοποίησης. Ως παράδειγμα μπορούμε να αναφέρουμε τον προσδιορισμό απειλής στην έγκριση δανείων ή πιστωτικών καρτών από μια τράπεζα.

Υπάρχουν τρεις κατηγορίες τεχνικών αναγνώρισης ανωμαλιών. Η **εποπτευόμενη αναγνώριση ανωμαλιών**, όπου οι τεχνικές που έχουν εφαρμοστεί σε αυτήν την κατηγορία, λαμβάνουν την διαθεσιμότητα τόσο ενός συνόλου δεδομένων εκπαίδευσης που έχει χαρακτηριστεί ως κανονικό όσο και ενός χαρακτηρισμένου ως ακραίων τιμών. Σε τέτοιες περιπτώσεις, συνήθως, εφαρμόζονται μοντέλα πρόβλεψης για κανονικές έναντι ανώμαλων τάξεων. Η **ημι-εποπτευομενη αναγνώριση ανωμαλιών**, όπου οι τεχνικές σε αυτή την περίπτωση θεωρούν ότι ένα σύνολο δεδομένων εκπαίδευσης έχει χαρακτηριστεί μόνο κανονικό. Το θετικό αυτής της κατηγορίας είναι ότι εφαρμόζεται σε μεγαλύτερη κλίμακα από την προηγούμενη, για το λόγο ότι δεν απαιτείται επισήμανση για την τάξη ακραίων τιμών. Η συνήθης προσέγγιση σε αυτή την κατηγορία είναι η κατασκευή ενός μοντέλου για το σύνολο ώστε να ανταποκρίνεται στην κανονική συμπεριφορά, και ύστερα η εφαρμογή του για τον προσδιορισμό των ανωμαλιών στα δοκιμαστικά δεδομένα. Τέλος, η **μη-εποπτευόμενη αναγνώριση ανωμαλιών**, όπου δεν απαιτούνται δεδομένα εκπαίδευσης, και έτσι είναι οι πιο ευρέως εφαρμόσιμες τεχνικές. Οι τεχνικές σε αυτή τη περίπτωση υποθέτουν ότι τα κανονικά στιγμιότυπα είναι περισσότερα από τις ακραίες τιμές στα δοκιμαστικά δεδομένα. Αν αυτός ο συλλογισμός δεν ισχύει, τότε οι τεχνικές έχουν μεγάλο ποσοστό λανθασμένης εκτίμησης.

Αρκετές τεχνικές αναγνώρισης ανωμαλιών έχουν προταθεί στην βιβλιογραφία. Από αυτές οι πιο δημοφιλείς είναι:

• Μέθοδοι που βασίζονται στην απόσταση (Distance based methods)

• Πρώτης-κλάσης Σύστημα διανυσμάτων υποστήριξης (One Class Support Vector Machines)

(η διαφορά του One Class Support Vector Machines σε σχέση με τα συνηθισμένα είναι ότι τα δεδομένα εκπαίδευσης δεν είναι πανομοιότυπα κατανεμημένα με αυτά του ελέγχου.)

• Επαναλαμβανόμενα Νευρωνικά Δίκτυα (Replicator Neural Networks)

• Αναγνώριση ακραίων τιμών βασισμένη στην συσταδοποίηση

• Αναγνώριση ανωμαλιών υπό όρους (Conditional Anomaly Detection)

--ΝΑ ΒΑΛΩ ΚΑΠΟΙΕΣ ΧΑΡΑΚΤΗΡΙΣΤΙΚΕΣ ΔΟΥΛΕΙΕΣ ΓΙΑ ΑΝΑΓΝΩΡΙΣΗ ΑΝΩΜΑΛΙΩΝ. ΑΠΟ 2 ΜΟΝΤΕΛΑ ΣΕ ΚΑΘΕ ΚΑΤΗΓΟΡΙΑ.--

Πολλές από τις τεχνικές που αναφέρθηκαν παραπάνω παρέχουν μόνο μια πρόβλεψη βαθμολογίας ανωμαλίας, η οποία συχνά μπορεί να εξηγηθεί στους χρήστες καθώς το σημείο βρίσκεται σε μια περιοχή χαμηλής πυκνότητας δεδομένων.

ΔΙΑΧΕΙΡΙΣΗ-ΦΙΛΤΡΑΡΙΣΜΑ

Ο χειρισμός των ακραίων τιμών στην ανίχνευση ανωμαλιών συνήθως περιλαμβάνει έναν συνδυασμό αλγοριθμικών προσεγγίσεων και ευρετικών για τον εντοπισμό, το φιλτράρισμα ή τη διαχείριση σημείων δεδομένων που αποκλίνουν σημαντικά από την αναμενόμενη κανονική συμπεριφορά. Σε αυτήν την ενότητα θα εστιάσουμε στις τεχνικές φιλτραρίσματος. Οι μέθοδοι φιλτραρίσματος περιλαμβάνουν τη χρήση συγκεκριμένων κριτηρίων ή φίλτρων για τον εντοπισμό και τον χειρισμό των ακραίων τιμών.

Ακολουθούν μερικές κοινές μέθοδοι φιλτραρίσματος:

1. Threshold filtering:

Ορίζουμε ένα προκαθορισμένο όριο πέραν του οποίου τα σημεία δεδομένων θεωρούνται outliers. Οποιοδήποτε σημείο δεδομένων υπερβαίνει αυτό το όριο είτε αφαιρείται είτε προσαρμόζεται. Για παράδειγμα, μπορούμε να ορίσουμε ένα όριο με βάση ένα συγκεκριμένο εκατοστημόριο ή μια σταθερή τιμή.

2. Range filtering:

Καθορίζουμε ένα εύρος εντός του οποίου αναμένεται να εμπίπτουν τα περισσότερα από τα δεδομένα σας. Τα σημεία δεδομένων εκτός αυτού του εύρους θεωρούνται ακραία και φιλτράρονται.

Με μια ευρεία έννοια, ο όρος "μέθοδοι φιλτραρίσματος" συνδέεται συχνά με τεχνικές που περιλαμβάνουν τον καθορισμό συγκεκριμένων κριτηρίων ή ορίων για τον εντοπισμό και τον χειρισμό των outliers ή των outliers σε ένα

σύνολο δεδομένων. Οι αλγόριθμοι μηχανικής μάθησης, συμπεριλαμβανομένων ορισμένων αλγορίθμων ανίχνευσης outlier, μπορούν πράγματι να θεωρηθούν ως μια μορφή μεθόδου φιλτραρίσματος.

Ωστόσο, είναι σημαντικό να σημειωθεί ότι όταν οι άνθρωποι αναφέρονται σε μεθόδους φιλτραρίσματος, μπορεί επίσης να μιλούν για απλούστερες τεχνικές όπως ο καθορισμός αριθμητικών ορίων ή η χρήση στατιστικών μέτρων για το φιλτράρισμα των ακραίων τιμών. Αντίθετα, οι αλγόριθμοι μηχανικής μάθησης, όπως π.χ ο one-class SVM ή ο Isolation Forest, είναι πιο προηγμένες τεχνικές που μπορούν να μάθουν αυτόματα μοτίβα και να εντοπίσουν ανωμαλίες στα δεδομένα.

Έτσι, ενώ τόσο οι παραδοσιακές μέθοδοι φιλτραρίσματος όσο και οι αλγόριθμοι μηχανικής μάθησης μπορούν να χρησιμοποιηθούν για αναγνώριση outliers ή χειρισμό των outliers, ο όρος "μέθοδοι φιλτραρίσματος" μπορεί πιο συχνά να συσχετίζεται με απλούστερες αριθμητικές ή στατιστικές προσεγγίσεις. Η χρήση αλγορίθμων μηχανικής μάθησης για αυτόν τον σκοπό μπορεί να προσφέρει πιο εξελιγμένους και προσαρμοστικούς τρόπους για τον εντοπισμό ακραίων τιμών, ειδικά σε πολύπλοκα και υψηλών διαστάσεων σύνολα δεδομένων.

Όταν χρησιμοποιούμε μεθόδους φιλτραρίσματος, είναι σημαντικό να επιλέγουμε προσεκτικά τα όρια ή τα κριτήρια με βάση τα χαρακτηριστικά των δεδομένων μας και τους στόχους της ανάλυσής μας. Οι μέθοδοι φιλτραρίσματος μπορεί να είναι πιο κατάλληλες για σύνολα δεδομένων όπου τα ακραία σημεία είναι σχετικά εύκολο να καθοριστούν και να εντοπιστούν με βάση συγκεκριμένα αριθμητικά κριτήρια.

ΚΕΦΑΛΑΙΟ 4 ΤΟ ΠΡΟΤΕΙΝΟΜΕΝΟ ΜΟΝΤΕΛΟ

ΤΙ ΕΙΝΑΙ ΤΟ ONE CLASS SVM

Ήταν το 1999 που οι Schölkopf et al. πρότεινε επέκταση στο SVM για την μη εποπτευόμενη μάθηση και με μεγαλύτερη ακρίβεια για την ανίχνευση καινοτομίας.

Ο αλγόριθμος διανύσματος υποστήριξης μιας κατηγορίας (one-class SVM) επιδιώκει να περιβάλλει τους inliers. Ο στόχος είναι να διαχωριστούν τα δεδομένα σε δύο κατηγορίες (με βάση μια συνάρτηση απόφασης), τη θετική που θεωρείται ως η κατηγορία των εσωτερικών(inliers) και την αρνητική ως την κατηγορία των ακραίων τιμών(outliers). Εκτός αυτού, τα περισσότερα δεδομένα εκπαίδευσης πρέπει να ανήκουν στη θετική τάξη, ενώ ο όγκος του φακέλου είναι ελάχιστος. Εμπίπτει στην κατηγορία των μηχανών με διανύσματα υποστήριξης (support vector machines), αλλά η κύρια εστίασή του είναι στην εκμάθηση των χαρακτηριστικών των κανονικών σημείων δεδομένων και στον εντοπισμό περιπτώσεων που αποκλίνουν σημαντικά από την κανονική κατανομή.

ΓΙΑΤΙ ΧΡΗΣΙΜΟΠΟΙΗΘΗΚΕ

Ο κύριος στόχος του One-Class SVM είναι να δημιουργήσει ένα όριο γύρω από τα περισσότερα σημεία δεδομένων, αντιμετωπίζοντάς τα ως "εσωτερικά" ή κανονικές περιπτώσεις. Στο παρεχόμενο πρόγραμμα, το One-Class SVM (OC-SVM) χρησιμοποιείται ως αλγόριθμος μάθησης χωρίς επίβλεψη. Σε αντίθεση με τους εποπτευόμενους αλγόριθμους μάθησης, το OC-SVM δεν απαιτεί δεδομένα με ετικέτα τόσο με κανονικές όσο και με ανώμαλες περιπτώσεις για εκπαίδευση. Αντίθετα, εκπαιδεύεται αποκλειστικά σε κανονικές περιπτώσεις για να μάθει τα χαρακτηριστικά της κανονικής συμπεριφοράς.

Στο πλαίσιο της ανίχνευσης ανωμαλιών, η υπόθεση είναι ότι η πλειονότητα των δεδομένων είναι φυσιολογικά και οι ανωμαλίες αναμένεται να είναι σπάνιες και διαφορετικές από τις κανονικές περιπτώσεις. Κατά τη διάρκεια της εκπαίδευσης, το OC-SVM κατασκευάζει ένα όριο απόφασης ή ένα υπερεπίπεδο που ενσωματώνει τις κανονικές περιπτώσεις στον χώρο χαρακτηριστικών. Στη συνέχεια, κατά τη διάρκεια της δοκιμής ή κατά την εφαρμογή του μοντέλου σε νέα δεδομένα, περιπτώσεις που βρίσκονται σημαντικά έξω από αυτό το μαθημένο όριο θεωρούνται πιθανές ανωμαλίες.

Η μη εποπτευόμενη φύση του OC-SVM το καθιστά ιδιαίτερα χρήσιμο σε σενάρια όπου τα επισημασμένα δεδομένα για ανωμαλίες είναι σπάνια ή μη διαθέσιμα, καθώς μπορεί να προσδιορίσει outliers με βάση τα χαρακτηριστικά κανονικών περιπτώσεων και μόνο.

ΥΛΟΠΟΙΗΣΗ

Στο πρόγραμμα υπάρχουν ορισμένες τροποποιήσεις και πρόσθετα στοιχεία που υλοποιούνται ειδικά για τον αλγόριθμο One-Class SVM (OC-SVM). Ακολουθούν βασικά στοιχεία που σχετίζονται με το OC-SVM στο πρόγραμμα:

- **Initialization:**

Το πρόγραμμα χρησιμοποιεί την κλάση One-Class SVM από το scikit-learn για να προετοιμάσει το μοντέλο OC-SVM. Η παράμετρος του 'kernel' έχει οριστεί σε 'linear' και η παράμετρος 'nu' έχει οριστεί σε 0,05. Η παράμετρος 'nu' ελέγχει την αναλογία των σφαλμάτων εκπαίδευσης και το περιθώριο του υπερεπιπέδου.

```
# fit the One-Class SVM model
svm = OneClassSVM(kernel='linear', nu=0.05)
```

- **Training & Prediction:**

Το μοντέλο OC-SVM εκπαιδεύεται χρησιμοποιώντας τη μέθοδο προσαρμογής στα κανονικά σημεία δεδομένων. Στη συνέχεια, η μέθοδος πρόβλεψης χρησιμοποιείται για τη λήψη ετικετών ακραίων τιμών (1 για inliers, -1 outliers) για ολόκληρο το σύνολο δεδομένων.

```
svm.fit(data_points)

y_pred = svm.predict(data_points)
```

- **Accuracy:**

Η ακρίβεια του μοντέλου OC-SVM υπολογίζεται με τη μέτρηση του λόγου των εσωτερικών στοιχείων προς τα συνολικά σημεία δεδομένων.

```
accuracy = np.sum(y_pred == 1) / len(y_pred)
```

• **Buffering and Sorting:**

Το πρόγραμμα χρησιμοποιεί ένα OutlierBuffer για την αποθήκευση ακραίων τιμών που εντοπίζονται από το OC-SVM. Η συνάρτηση sort_outliers χρησιμοποιείται για το φιλτράρισμα και την ταξινόμηση των ακραίων τιμών με βάση μια υπολογισμένη βαθμολογία ταξινόμησης.

```
# store good data points in a list and outliers in the buffer
good_data_points_svm= data_points[y_pred == 1].tolist()
outlier_buffer_svm = OutlierBuffer(w)
outliers_detected_svm = data_points[y_pred == -1]
for outlier in outliers_detected_svm:
    outlier_buffer_svm.add(outlier)
outliers_detected_svm = outlier_buffer_svm.get_all()
threshold = 0.99999999999781  # threshold
```

```
# Get sorted outliers
sorted_outliers_svm,good_data_points_svm = sort_outliers(
```

• **Dynamic Adjustments:**

Το πρόγραμμα προσαρμόζει δυναμικά τα όρια φίλτρου με βάση τα χαρακτηριστικά των ακραίων τιμών που είναι αποθηκευμένα στο buffer τόσο για το Isolation Forest όσο και για το OC-SVM.

```
# Calculate filter bounds
xmin_outlier, xmax_outlier, ymin_outlier, ymax_outlier = calculate_filter_bounds(sorted_outliers_svm)

# Get filtered outliers based on updated filter bounds
filtered_outliers = sorted_outliers_svm[
    (sorted_outliers_svm[:, 0] >= xmin_outlier[0]) &
    (sorted_outliers_svm[:, 0] <= xmax_outlier[0]) &
    (sorted_outliers_svm[:, 1] >= ymin_outlier[1]) &
    (sorted_outliers_svm[:, 1] <= ymax_outlier[1])
]
```

Αυτές οι τροποποιήσεις αφορούν ειδικά την ενσωμάτωση του αλγόριθμου OC-SVM στη διαδικασία ανίχνευσης ανωμαλιών και τη διασφάλιση ότι ευθυγραμμίζεται με την άνευ επίβλεψης φύση του OC-SVM για ανίχνευση των outliers.

Σε αυτή την ενότητα θα αναλύσουμε και θα περιγράψουμε την λειτουργία των συναρτήσεων που χρησιμοποιήθηκαν για την αναγνώριση ανωμαλιών και πως με αυτές τις συναρτήσεις διαχειρίστηκαν αυτές οι ανωμαλίες.

OutlierBuffer:

Η κλάση OutlierBuffer είναι μια δομή δεδομένων που διατηρεί μια προσωρινή μνήμη καθορισμένου μεγέθους. Ο κύριος σκοπός του είναι να αποθηκεύει τα outliers που ανιχνεύονται κατά τη διαδικασία αναγνώρισης ανωμαλιών. Το buffer λειτουργεί με τον τρόπο first-in-first-out, πράγμα που σημαίνει ότι όταν η προσωρινή μνήμη φτάσει στη μέγιστη χωρητικότητά της τα παλαιότερα στοιχεία αφαιρούνται για να δημιουργηθεί χώρος για νέα. Η μέθοδος 'add' προσθέτει νέα στοιχεία στο buffer διατηρώντας το καθορισμένο όριο μεγέθους και η μέθοδος 'get_all' ανακτά όλα τα στοιχεία που είναι αποθηκευμένα αυτήν τη στιγμή στην προσωρινή μνήμη.

Ας δούμε πως υλοποιείται και πως λειτουργεί:

Initialization ("_ _init_ _"):

Η κλάση OutlierBuffer αρχικοποιειται με ένα καθορισμένο μέγεθος ('size'). Αυτό το μέγεθος αντιπροσωπεύει τον μέγιστο αριθμό περιπτώσεων που μπορεί να αποθηκεύσει το buffer.

```
class OutlierBuffer:
    def __init__(self, size):
        self.size = size
        self.buffer = []
```

Adding outliers ("add"):

Η μέθοδος προσθήκης είναι υπεύθυνη για την προσθήκη μιας νέας παρουσίας outlier στο buffer. Εάν το μέγεθος του buffer υπερβαίνει το καθορισμένο όριο ('size'), το παλαιότερο outlier αφαιρείται για να διατηρηθεί το μέγεθος του buffer.

```
def add(self, item):
    self.buffer.append(item)
    if len(self.buffer) > self.size:
        self.buffer.pop(0)
```

Getting all the outliers (“get_all”):

Η μέθοδος get_all επιστρέφει όλες τις παρουσίες outlier που είναι αποθηκευμένες μια συγκεκριμένη στιγμή στο buffer.

```
def get_all(self):
    return self.buffer
```

Buffer size:

Το μέγεθος του buffer είναι μια παράμετρος που επιτρέπει στο χρήστη να ελέγχει πόσα πρόσφατα outliers αποθηκεύονται. Ένα μεγαλύτερο buffer μπορεί να καταγράψει ένα πιο εκτεταμένο ιστορικό outliers, αλλά απαιτεί περισσότερη μνήμη.

Usage:

Το buffer χρησιμοποιείται για την αποθήκευση των outliers που αναγνωρίζονται από τον αλγόριθμο SVM One-Class. Στο πρόγραμμα, το buffer ενημερώνεται με νέα outliers κατά τη διάρκεια κάθε επανάληψης του αλγορίθμου.

Retrieval:

Η μέθοδος get_all επιτρέπει την ανάκτηση όλων των παρουσιών των outliers που είναι αποθηκευμένοι στο buffer. Αυτό μπορεί να είναι χρήσιμο για περαιτέρω ανάλυση, οπτικοποίηση ή δυναμικές προσαρμογές στη διαδικασία αναγνώρισης ανωμαλιών.

Dynamic adjustments:

Το buffer παίζει ρόλο στη δυναμική προσαρμογή των ορίων του φίλτρου με βάση τα χαρακτηριστικά των outliers. Καταγράφοντας τους πρόσφατους outliers, το πρόγραμμα μπορεί να προσαρμόσει τα κριτήρια φιλτραρίσματος στην εξελισσόμενη φύση των δεδομένων.

Συνοπτικά, η κλάση OutlierBuffer χρησιμεύει ως εργαλείο για την αποθήκευση και τη διαχείριση πρόσφατων outliers, επιτρέποντας δυναμικές προσαρμογές και παρέχοντας πληροφορίες για την ιστορική ανώμαλη συμπεριφορά των δεδομένων

Sort_Outliers:

Η συνάρτηση sort_outliers είναι υπεύθυνη για την κατηγοριοποίηση και την ταξινόμηση των ακραίων τιμών με βάση την απόστασή τους από τα κανονικά σημεία δεδομένων και το κέντρο. Υπολογίζει αποστάσεις, εφαρμόζει συγκεκριμένους τύπους για να εκχωρήσει βαθμολογίες ταξινόμησης σε κάθε ακραίο σημείο και τις κατηγοριοποιεί είτε ως πιθανές ανωμαλίες είτε ως κανονικά σημεία δεδομένων. Η συνάρτηση χρησιμοποιεί διάφορες παραμέτρους, όπως gamma, delta, gamma_prime, delta_prime και ένα threshold, για να καθορίσει εάν ένα outlier θα πρέπει να ταξινομηθεί ως ανωμαλία ή ως κανονικό σημείο δεδομένων.

Το παρακάτω κομμάτι κώδικα είναι η συνάρτηση αυτή:

```
51
52  def sort_outliers(outliers_detected, good_data_points, centroid, gamma, delta, gamma_prime, delta_prime, threshold):
53
54      if len(good_data_points) == 0:
55          return np.array([]),np.array([])
56      sorted_outliers = []
57      for outlier in outliers_detected:
58          # Calculate distances
59          distance_a = np.linalg.norm(outlier - good_data_points, axis=1)
60          distance_b = np.linalg.norm(outlier - centroid)
61          x1 = gamma * distance_a + delta
62          x2 = gamma_prime * distance_b + delta_prime
63          sort_score_a = 1 / (1 + np.exp(-x1))
64          sort_score_b = 1 / (1 + np.exp(-x2))
65          sort_score = sort_score_a * sort_score_b
66          # Add outlier as a good data point if the sort score is below threshold
67          if np.all(sort_score < threshold):
68              good_data_points.append(outlier)
69          else:
70              sorted_outliers.append(outlier)
71      return np.array(sorted_outliers),good_data_points
```

Input parameters:

• **outliers_detected**: Πίνακας που περιέχει τους outliers που ανιχνεύονται από τον αλγόριθμο αναγνώρισης ανωμαλιών.

• **good_data_points**: Λίστα που περιέχει τα τρέχοντα καλά σημεία δεδομένων.

• **centroid:** Το κέντρο για τα καλά δεδομένα.

• **gamma,delta:** Παράμετροι που χρησιμοποιούνται στον υπολογισμό του x1 για την ταξινόμηση των outliers.

• **gamma_prime,delta_prime:** Παράμετροι που χρησιμοποιούνται στον υπολογισμό του x2 για την ταξινόμηση των outliers.

• **threshold:** Η τιμή του threshold χρησιμοποιείται για τον προσδιορισμό του εάν ένας outlier πρέπει να θεωρείται καλό σημείο δεδομένων με βάση τις βαθμολογίες ταξινόμησης.

Functionality:

Η συνάρτηση επαναλαμβάνεται σε κάθε outlier που αναγνωρίζεται και υπολογίζει τις βαθμολογίες ταξινόμησης (sort_score_a και sort_score_b) με βάση τις αποστάσεις και τις παραμέτρους. Εάν η συνδυασμένη βαθμολογία ταξινόμησης (sort_score) για ένα outlier είναι κάτω από το καθορισμένο όριο (threshold), το outlier θεωρείται καλό σημείο δεδομένων και προστίθεται στη λίστα των καλών σημείων δεδομένων. Διαφορετικά, προστίθεται στη λίστα των ταξινομημένων outliers.

Output:

Η συνάρτηση επιστρέφει δύο πίνακες: sorted_outliers (outliers που δεν πληρούν το όριο και θεωρούνται ταξινομημένα) και good_data_points (ενημερωμένη λίστα καλών σημείων δεδομένων).

Dynamic adjustments:

Η συνάρτηση αποτελεί μέρος της διαδικασίας δυναμικής προσαρμογής, όπου τα outliers αξιολογούνται με βάση τις βαθμολογίες ταξινόμησης τους και είτε διατηρούνται ως καλά σημεία δεδομένων είτε ταξινομούνται με βάση το όριο.

Sort_scores:

Οι βαθμολογίες ταξινόμησης υπολογίζονται χρησιμοποιώντας αποστάσεις και παραμέτρους (gamma, delta, gamma_prime, delta_prime). Η σιγμοειδής συνάρτηση (1 / (1 + np.exp(-x))) χρησιμοποιείται για να μετασχηματίσει αυτές τις βαθμολογίες.

Iteration:

Η συνάρτηση επαναλαμβάνεται μέσω κάθε εντοπισμένου outlier, εφαρμόζοντας τα κριτήρια ταξινόμησης και ενημερώνοντας ανάλογα τις λίστες.

Συνοπτικά, η συνάρτηση sort_outliers διαδραματίζει κρίσιμο ρόλο στη διαδικασία προσαρμοστικής αναγνώρισης ανωμαλιών του προγράμματος ταξινομώντας δυναμικά τα outliers βάσει υπολογισμένων βαθμολογιών ταξινόμησης και ενός καθορισμένου ορίου. Επιτρέπει στο πρόγραμμα να ενημερώνει συνεχώς την κατανόησή του για το τι συνιστά φυσιολογική συμπεριφορά στα εξελισσόμενα δεδομένα.

Calculate_Filter_Bounds:

Η συνάρτηση 'calculate_filter_bounds' υπολογίζει τα όρια του φίλτρου βάσει των ταξινομημένων outliers. Προσδιορίζει τις ελάχιστες και μέγιστες τιμές και για τις διαστάσεις x και y και προσαρμόζει αυτά τα όρια με βάση τον αριθμό των outliers που εμπίπτουν σε ορισμένα κατώφλια. Η συνάρτηση υπολογίζει το διάνυσμα 'w_prime', που αποτελείται από μετρήσεις για διαφορετικές κατηγορίες ακραίων τιμών. Χρησιμοποιώντας αυτό το διάνυσμα, η συνάρτηση ενημερώνει δυναμικά τα όρια του φίλτρου για να συλλάβει αποτελεσματικά τα ακραία σημεία που εμπίπτουν στα καθορισμένα όρια.

Το παρακάτω κομμάτι κώδικα είναι η συνάρτηση αυτή:

```
24
25  def calculate_filter_bounds(sorted_outliers):
26      sorted_outliers = np.array(sorted_outliers)
27      xmin_outlier = [np.min(sorted_outliers[:, 0]), np.argmin(sorted_outliers[:, 0])]
28      xmax_outlier = [np.max(sorted_outliers[:, 0]), np.argmax(sorted_outliers[:, 0])]
29      ymin_outlier = [np.min(sorted_outliers[:, 1]), np.argmin(sorted_outliers[:, 1])]
30      ymax_outlier = [np.max(sorted_outliers[:, 1]), np.argmax(sorted_outliers[:, 1])]
31
32      # Calculate w_prime vector (a_count, b_count, c_count, d_count)
33      a_count = np.sum(np.isclose(sorted_outliers, [xmin_outlier[0], 0]))
34      b_count = np.sum(np.isclose(sorted_outliers, [xmax_outlier[0], 0]))
35      c_count = np.sum(np.isclose(sorted_outliers, [0, ymin_outlier[1]]))
36      d_count = np.sum(np.isclose(sorted_outliers, [0, ymax_outlier[1]]))
37      w_prime = np.array([a_count, b_count, c_count, d_count])
38      theta_prime = 5
39      # Check and update filter bounds based on w_prime vector
40      for sorted_outliers in w_prime:
41          if a_count > theta_prime:
42              xmin_outlier[0] *= 0.8
43          if b_count > theta_prime:
44              xmax_outlier[0] *= 1.2
45          if c_count > theta_prime:
46              ymin_outlier[1] *= 0.8
47          if d_count > theta_prime:
48              ymax_outlier[1] *= 1.2
49      return xmin_outlier, xmax_outlier, ymin_outlier, ymax_outlier
50
```

Input parameters:

sorted_outliers: Πίνακας που περιέχει τα outliers που έχουν ταξινομηθεί με βάση τα κριτήρια που καθορίζονται στη συνάρτηση sort_outliers.

Functionality: Η συνάρτηση παίρνει τα ταξινομημένα outliers και υπολογίζει συγκεκριμένα όρια (xmin_outlier, xmax_outlier, ymin_outlier, ymax_outlier) με βάση τις ελάχιστες και μέγιστες τιμές κατά μήκος κάθε άξονα.

Vector 'w_prime': Η συνάρτηση υπολογίζει ένα διάνυσμα w_prime που αντιπροσωπεύει πλήθος εμφανίσεων σε συγκεκριμένα τεταρτημόρια με βάση τα ταξινομημένα outliers.

Threshold: Το όριο (theta_prime) ορίζεται σε μια προκαθορισμένη. Το όριο εφαρμόζεται για να καθοριστεί πότε πρέπει να ρυθμιστούν τα όρια του φίλτρου. Εάν η καταμέτρηση σε ένα συγκεκριμένο τεταρτημόριο υπερβαίνει το όριο, το αντίστοιχο όριο φίλτρου ενημερώνεται.

Filter Bounds Update: Η συνάρτηση ελέγχει τις μετρήσεις σε κάθε τεταρτημόριο (a_count, b_count, c_count, d_count) σε σχέση με το όριο και ενημερώνει τα αντίστοιχα όρια φίλτρου (xmin_outlier, xmax_outlier, ymin_outlier, ymax_outlier) ανάλογα. Η συνάρτηση υπολογίζει τα όρια του φίλτρου με βάση τη χωρική κατανομή των ταξινομημένων outliers. Αυτά τα όρια χρησιμοποιούνται για το φιλτράρισμα των outliers στα επόμενα βήματα του αλγορίθμου.

Quadrant Counts: Οι μετρήσεις τεταρτημόριων υπολογίζονται για την κατανόηση της κατανομής των outliers σε διαφορετικές περιοχές του χώρου χαρακτηριστικών.

Adaptability: Η προσαρμοστική φύση των ορίων του φίλτρου επιτρέπει στον αλγόριθμο να προσαρμόζει δυναμικά τα κριτήρια φιλτραρίσματος με βάση την παρατηρούμενη κατανομή των outliers.

Iterations: Η συνάρτηση καλείται επαναληπτικά ως μέρος της συνολικής διαδικασίας αναγνώρισης ανωμαλιών. Μετά τον υπολογισμό των ορίων του φίλτρου, τα επόμενα βήματα του αλγορίθμου χρησιμοποιούν αυτά τα όρια για να φιλτράρουν δυναμικά τα outliers.

Συνοπτικά, η συνάρτηση 'calculate_filter_bounds' παίζει κρίσιμο ρόλο στη δυναμική προσαρμογή των ορίων του φίλτρου με βάση την κατανομή των ταξινομημένων outliers. Αυτή η προσαρμοστικότητα επιτρέπει στο πρόγραμμα να εντοπίζει και να διαχειρίζεται αποτελεσματικά τις ανωμαλίες στα εξελισσόμενα δεδομένα.

ΣΥΜΠΕΡΑΣΜΑ

Το μοντέλο ενσωματώνει τα πλεονεκτήματα του αλγόριθμου SVM One-Class, χρησιμοποιεί ένα ακραίο buffer για δυναμική διαχείριση και προσαρμόζει παραμέτρους για αποτελεσματική ανίχνευση ανωμαλιών. Οι μηχανισμοί ταξινόμησης και φιλτραρίσματος συμβάλλουν στην προσαρμοστικότητα του μοντέλου, καθιστώντας το κατάλληλο για διάφορα σύνολα δεδομένων με εξελισσόμενα μοτίβα των outliers. Η χρήση του αλγόριθμου SVM One-Class παρέχει μια ισχυρή προσέγγιση, εξισορροπώντας τα δυνατά σημεία μεμονωμένων τεχνικών για βελτιωμένη απόδοση αναγνώρισης ανωμαλιών.

ISOLATION FOREST

Το Isolation Forest είναι ένας αλγόριθμος για την ανίχνευση ανωμαλιών δεδομένων που αναπτύχθηκε αρχικά από τον Fei Tony Liu το 2008. Το Isolation Forest ανιχνεύει ανωμαλίες χρησιμοποιώντας δυαδικά δέντρα. Ο αλγόριθμος έχει γραμμική πολυπλοκότητα χρόνου και χαμηλή απαίτηση μνήμης, η οποία λειτουργεί καλά με δεδομένα μεγάλου όγκου. Ουσιαστικά, ο αλγόριθμος βασίζεται στα χαρακτηριστικά των ανωμαλιών, δηλαδή στο ότι είναι λίγα και διαφορετικά, προκειμένου να ανιχνεύσει ανωμαλίες. Δεν πραγματοποιείται εκτίμηση πυκνότητας στον αλγόριθμο. Ο αλγόριθμος είναι διαφορετικός από τους αλγόριθμους δένδρων αποφάσεων καθώς χρησιμοποιείται μόνο η μέτρηση μήκους διαδρομής ή η προσέγγιση για τη δημιουργία της βαθμολογίας ανωμαλίας, δεν χρειάζονται στατιστικά στοιχεία κόμβου φύλλου για την κατανομή κλάσης ή την τιμή στόχου.

Το Isolation Forest είναι γρήγορο γιατί χωρίζει τυχαία τον χώρο δεδομένων, χρησιμοποιώντας τυχαία επιλεγμένο χαρακτηριστικό και τυχαία επιλεγμένο σημείο διαχωρισμού. Η βαθμολογία της ανωμαλίας συνδέεται ανεστραμμένα με το μήκος διαδρομής, καθώς οι ανωμαλίες χρειάζονται λιγότερες διασπάσεις για να απομονωθούν, λόγω του γεγονότος ότι είναι λίγες και διαφορετικές.

Η υπόθεση του αλγορίθμου Isolation Forest είναι ότι τα ανώμαλα σημεία δεδομένων είναι ευκολότερο να διαχωριστούν από το υπόλοιπο δείγμα. Προκειμένου να απομονωθεί ένα σημείο δεδομένων, ο αλγόριθμος δημιουργεί αναδρομικά διαμερίσματα στο δείγμα επιλέγοντας τυχαία ένα χαρακτηριστικό και, στη συνέχεια, επιλέγοντας τυχαία μια τιμή διαχωρισμού μεταξύ των ελάχιστων και μέγιστων τιμών που επιτρέπονται για αυτό το χαρακτηριστικό.

ΓΙΑΤΙ ΧΡΗΣΙΜΟΠΟΙΗΘΗΚΕ

Στο παρεχόμενο πρόγραμμα, ο κύριος στόχος του αλγορίθμου Isolation Forest είναι ο εντοπισμός και η απομόνωση ανωμαλιών (outliers) μέσα σε ένα σύνολο δεδομένων. Ο αλγόριθμος Isolation Forest επιτυγχάνει αυτόν τον στόχο μέσω μιας προσέγγισης συνόλου που βασίζεται σε δέντρα που εστιάζει στην απομόνωση περιπτώσεων που θεωρούνται σπάνιες ή ασυνήθιστες. Πέρα από αυτό χρησιμοποιείται για να γίνει σύγκριση μεταξύ αυτού και του αλγορίθμου που αναφέραμε προηγουμένως τον One-Class SVM. Η σύγκριση μεταξύ των αλγορίθμων SVM One-Class και Isolation Forest στο πρόγραμμά περιλαμβάνει την αξιολόγηση της απόδοσής τους στον εντοπισμό outliers υπό διάφορες συνθήκες.

ΥΛΟΠΟΙΗΣΗ

Στο πρόγραμμα υλοποιούνται οι αλγόριθμοι One-Class SVM όσο και Isolation Forest με συγκεκριμένες παραμέτρους και ρυθμίσεις. Θα εστιάσουμε στο τμήμα Isolation Forest και ας δούμε τις τροποποιήσεις και συγκεκριμένες διαμορφώσεις:

Initialization: Η παράμετρος ‘contamination’ ορίζεται στο 0,05, υποδεικνύοντας την αναμενόμενη αναλογία ακραίων τιμών στο σύνολο δεδομένων.

```
# Create an Isolation Forest model
isolation_forest = IsolationForest(contamination=0.05, random_state=42)
```

Outlier label prediction: Η μέθοδος πρόβλεψης χρησιμοποιείται για τη λήψη ετικετών ακραίων τιμών (1 για inliers, -1 για outliers) με βάση το μοντέλο.

```
# Predict outlier labels (1 for inliers, -1 for outliers)
outlier_labels = isolation_forest.predict(data_points)
```

Accuracy: Η ακρίβεια υπολογίζεται με βάση την αναλογία των inliers προς τον συνολικό αριθμό των σημείων δεδομένων.

```
# Calculate accuracy (ratio of inliers to total data points)
accuracy = np.sum(outlier_labels == 1) / len(outlier_labels)
```

Buffer management: Το πρόγραμμα χρησιμοποιεί την κλάση OutlierBuffer για τη διαχείριση των outliers με επαναληπτικό τρόπο. Το όριο για την ανίχνευση των outliers προσαρμόζεται δυναμικά με βάση το μέγεθος της προσωρινής μνήμης outliers.

Διαπιστώνουμε ότι το πρόγραμμα ακολουθεί παρόμοια δομή και για τους δυο αλγορίθμους, με διαφορές κυρίως στην επιλογή αλγορίθμου, παραμέτρων και συγκεκριμένων βημάτων επεξεργασίας.

ΚΕΦΑΛΑΙΟ 5 ΠΕΙΡΑΜΑΤΑ

ΕΙΣΑΓΩΓΗ

Η αναγνώριση ανωμαλιών είναι μια κρίσιμη πτυχή της ανάλυσης δεδομένων, περιλαμβάνει τον εντοπισμό περιπτώσεων που αποκλίνουν σημαντικά από τον κανόνα σε ένα σύνολο δεδομένων. Σε αυτή τη μελέτη, εμβαθύνουμε στη σφαίρα της ανίχνευσης ακραίων τιμών χρησιμοποιώντας δύο διακριτούς αλγόριθμους: SVM One-Class (Support Vector Machine) και Isolation Forest. Και οι δύο αλγόριθμοι είναι καθιερωμένες μεθοδολογίες στον τομέα της μάθησης χωρίς επίβλεψη (unsupervised learning), καθένας από τους οποίους προσφέρει μοναδικές προσεγγίσεις για τον εντοπισμό ανωμαλιών σε σύνολα δεδομένων.

Το κύριο κίνητρο για αυτήν τη συγκριτική ανάλυση πηγάζει από τα διαφορετικά χαρακτηριστικά των συνόλων δεδομένων που συναντώνται σε σενάρια πραγματικού κόσμου. Η κατανόηση των πλεονεκτημάτων και των αδυναμιών διαφορετικών αλγορίθμων αναγνώρισης ανωμαλιών είναι ζωτικής σημασίας για την επιλογή της καταλληλότερης λύσης για συγκεκριμένες περιπτώσεις χρήσης. Ως εκ τούτου, αυτή η μελέτη στοχεύει να παρέχει πληροφορίες σχετικά με τον τρόπο απόδοσης του SVM One-Class και του Isolation Forest όσον αφορά την αποτελεσματικότητα, την ακρίβεια και την προσαρμοστικότητα σε δυναμικά σύνολα δεδομένων.

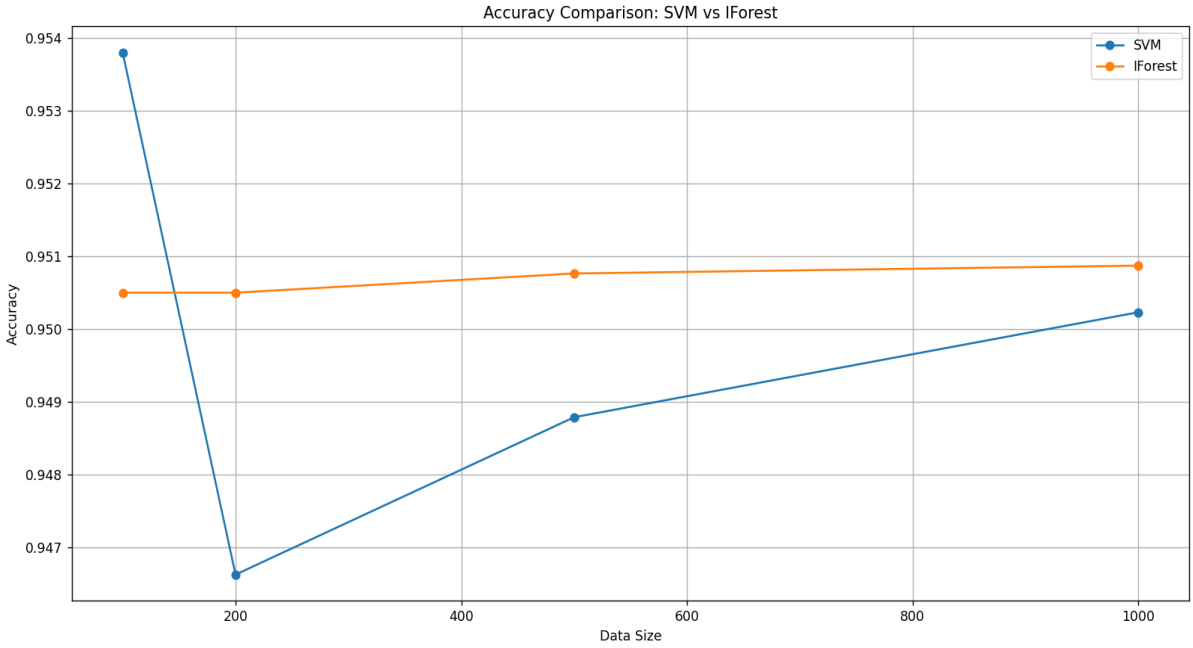
Για την επιδίωξη μιας ολοκληρωμένης αναγνώρισης ανωμαλιών, διεξήχθη μια σειρά από έξι πειράματα, καθένα από τα οποία εισήγαγε διακριτές παραλλαγές στις παραμέτρους και τα χαρακτηριστικά. Τα πειράματα στόχευαν να αξιολογήσουν την προσαρμοστικότητα των αλγορίθμων SVM One-Class και Isolation Forest υπό διαφορετικές συνθήκες. Βασικοί παράγοντες, συμπεριλαμβανομένου του μεγέθους buffer (w), των περιορισμών διαστήματος (low,high) και της δυναμικής φύσης των συνόλων δεδομένων, χειραγωγήθηκαν συστηματικά για τον έλεγχο της αλγοριθμικής απόδοσης.

Τα αποτελέσματα αυτών των πειραμάτων αναμένεται να δώσουν πολύτιμες πληροφορίες σχετικά με τη διαφοροποιημένη συμπεριφορά του One-Class SVM και του Isolation Forest κάτω από διάφορα σενάρια αναγνώρισης ανωμαλιών.

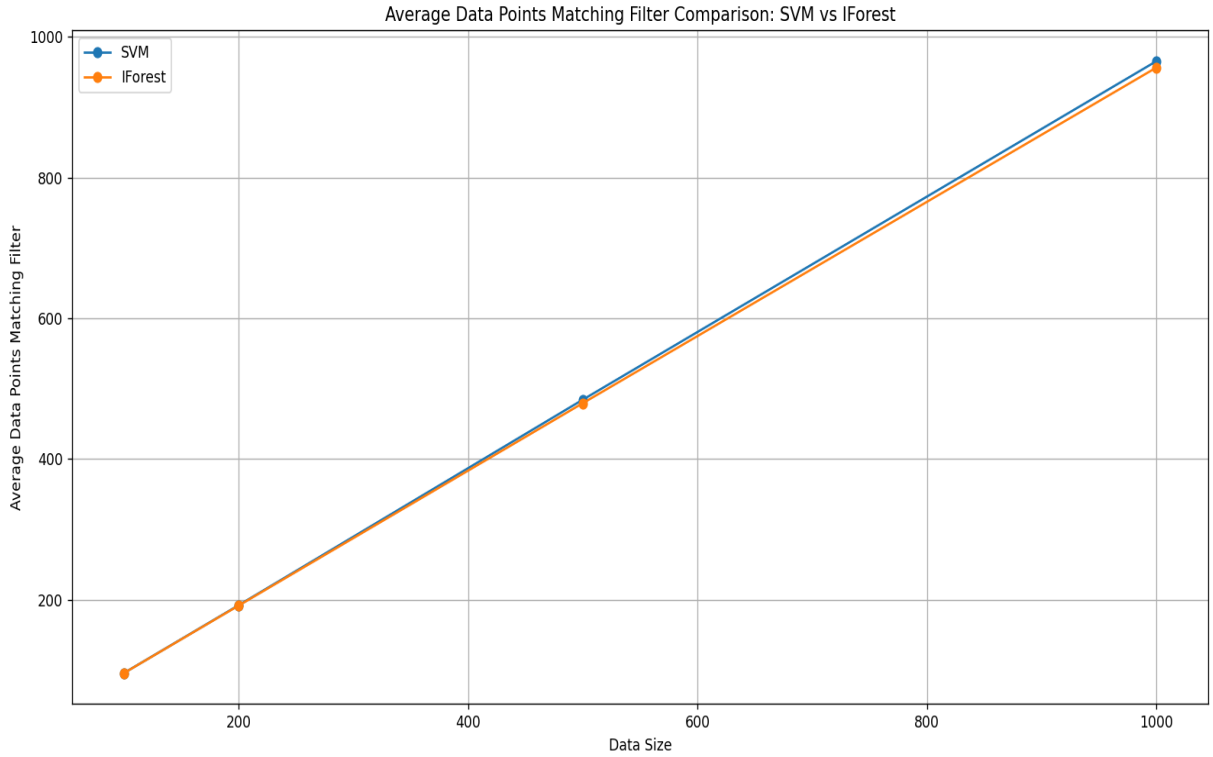
Στο πρώτο πείραμα, διατηρήσαμε τις αρχικές τιμές και τις διαμορφώσεις όπως ορίζονται στο παρεχόμενο πρόγραμμα. Αυτή η βασική ρύθμιση επέτρεψε τη δημιουργία ενός σημείου αναφοράς, παρέχοντας πληροφορίες για την εγγενή συμπεριφορά του One-Class SVM και του Isolation Forest χωρίς σκόπιμες αλλαγές στις παραμέτρους.

- **Παράμετροι:**

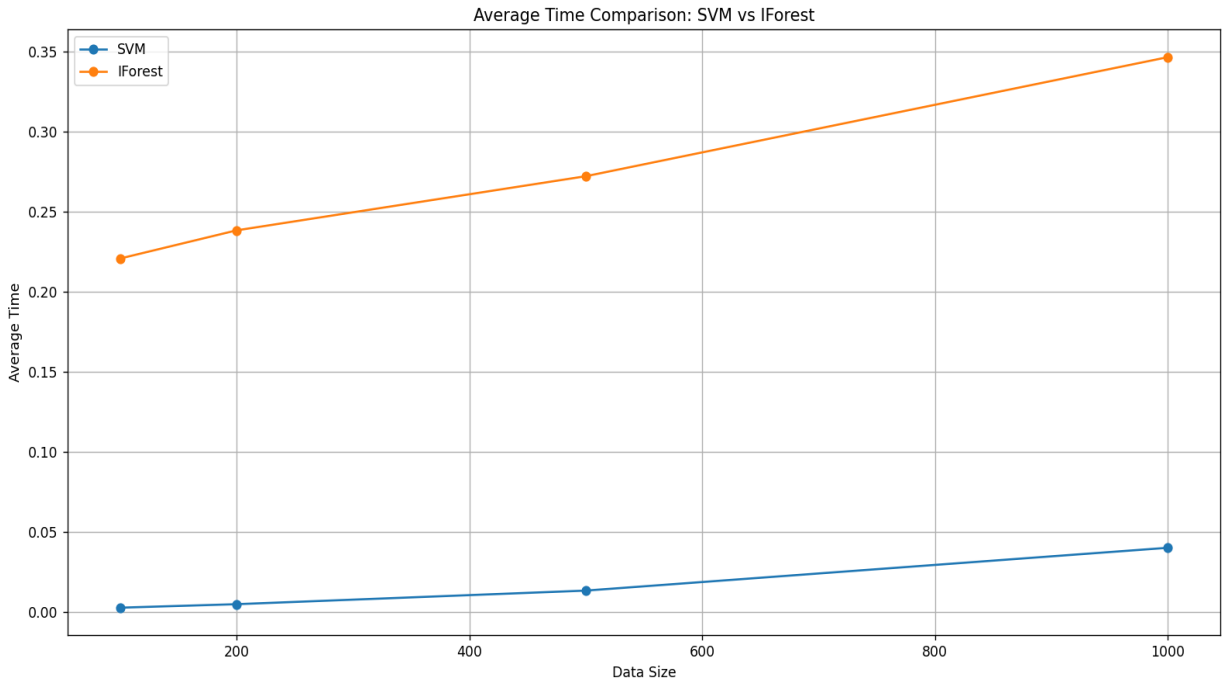
W=100
Low=0
High=30



Το παραπάνω γράφημα μας δείχνει την ακρίβεια (accuracy) των αλγόριθμων. Παρατηρούμε ότι και οι δυο αλγόριθμοι έχουν πολύ καλές αποδόσεις για κάθε μέγεθος δεδομένων. Στο μέγεθος με τα 100 δεδομένα το One-Class SVM είναι λίγο καλύτερο με 0.953 ενώ το Isolation Forest έχει 0.95 και παραμένει σταθερό στα επόμενα σύνολα δεδομένων σε αντίθεση με το One-Class Svm. Στο μέγεθος με τα 200 δεδομένα το One-Class SVM έπεσε σε ακρίβεια με σκορ 0.947, βέβαια αυτό δεν έχει μεγάλη διαφορά από το προηγούμενο σκορ και δεν θα επηρεάσει τα αποτελέσματα και τις επιδόσεις του αλγορίθμου. Στα επόμενα μεγέθη δεδομένων παρατηρούμε ότι το SVM ανεβαίνει σε ακρίβεια και φτάνει τον Isolation Forest. Πρέπει να σημειωθεί ότι δεν υπάρχει μεγάλη διαφορά των αλγόριθμων στην ακρίβεια.

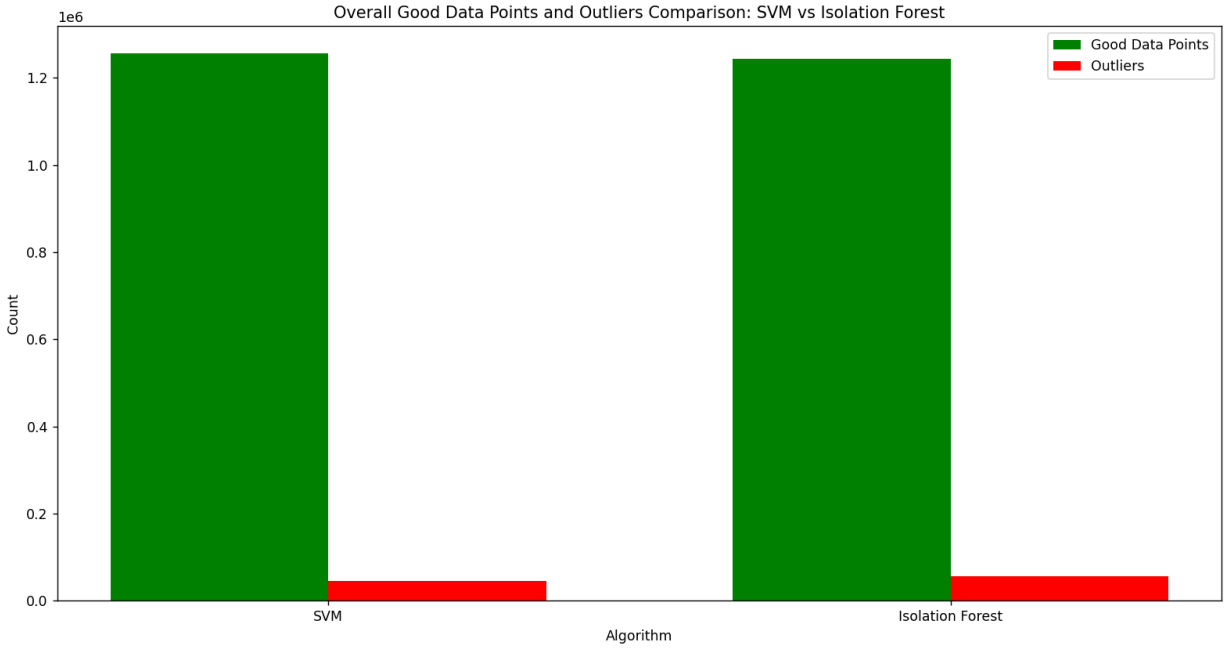


Το παραπάνω γράφημα μας δείχνει την αντιστοιχία κατά μέσο όρο σημείων δεδομένων που φιλτράρονται. Διαπιστώνουμε ότι και οι δυο αλγόριθμοι διαχειρίζονται πολύ καλά τις ανωμαλίες δεδομένων και ότι πολλά δεδομένα φιλτραρίζονταν αποτελεσματικά σε καλά δεδομένα. Βέβαια στο μέγεθος με τα 1000 δεδομένα παρατηρούμε ότι το SVM έχει λίγο καλύτερη απόδοση από το Isolation Forest και το ξεπερνά. Αυτό μπορεί να φαίνεται πως δεν έχει μεγάλη διαφορά αλλά στα αποτελέσματα θα διαπιστώσουμε ότι είναι αρκετά κρίσιμη η διαφορά τους στο φιλτράρισμα των δεδομένων.

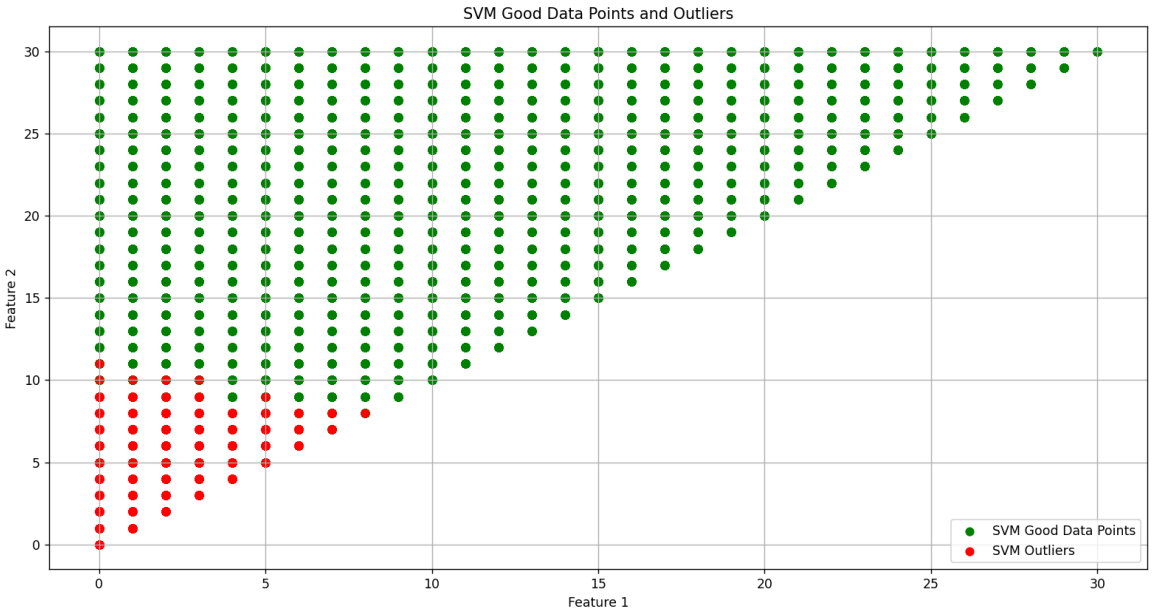


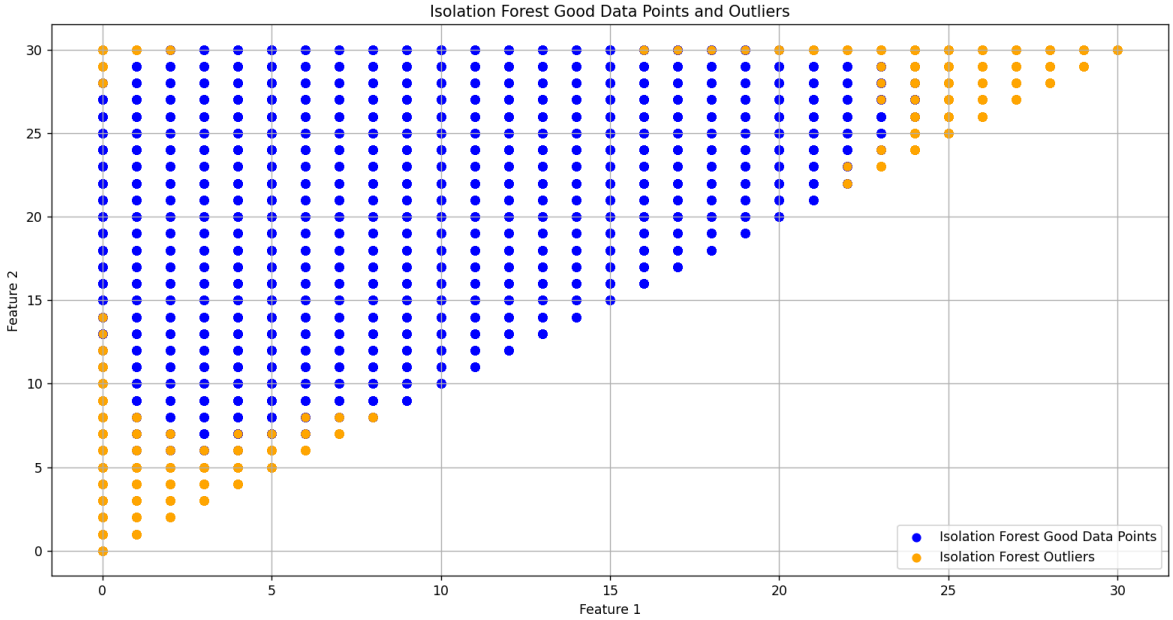
Το παραπάνω γράφημα μας δείχνει τον μέσο χρόνο που χρειάστηκαν οι αλγόριθμοι για κάθε σημείο δεδομένων. Εδώ φαίνεται η διαφορά του χρόνου μεταξύ των δυο αλγορίθμων με τον SVM να είναι διακριτά πιο γρήγορος από των Isolation Forest. Ας αναλύσουμε τις τιμές αυτές για να έχουμε μια ιδέα το πόσο πιο γρήγορα εκτελείτε ο SVM σε σύγκριση με τον Isolation Forest.

Ας πάρουμε για παράδειγμα το μέγεθος δεδομένων που είναι 100. Το SVM κάνει περίπου 0.001 δευτερόλεπτα για την αναγνώριση και την επεξεργασία κάθε σημείου σε αυτό το σύνολο δεδομένων όποτε συνολικά σε αυτό το μέγεθος κάνει 0,1 δευτερόλεπτα. Από την άλλη πλευρά το Isolation Forest κάνει περίπου 0,24 δευτερόλεπτα για την αναγνώριση και την επεξεργασία κάθε σημείου σε αυτό το σύνολο δεδομένων οπότε συνολικά ο χρόνος που θα χρειαστεί να κάνει σε αυτό το σύνολο δεδομένων είναι 24 δευτερόλεπτα. Άρα μπορούμε να καταλάβουμε ότι στα επόμενα σύνολα δεδομένων ο Isolation Forest είναι θα είναι πιο αργός από τον SVM.



Σε αυτό το γράφημα βλέπουμε τα δεδομένα που έχουν βρει και διαχειριστεί οι δυο αλγόριθμοι. Μπορούμε να δούμε ότι και τα καλά δεδομένα και τα outliers είναι σχεδόν ίδια και στους δυο αλγορίθμους, βέβαια μπορεί να μην φαίνεται άμεσα αλλά ο SVM ξεπερνάει λίγο τον Isolation Forest. Αυτό θα φανεί στα επόμενα γραφηματα.





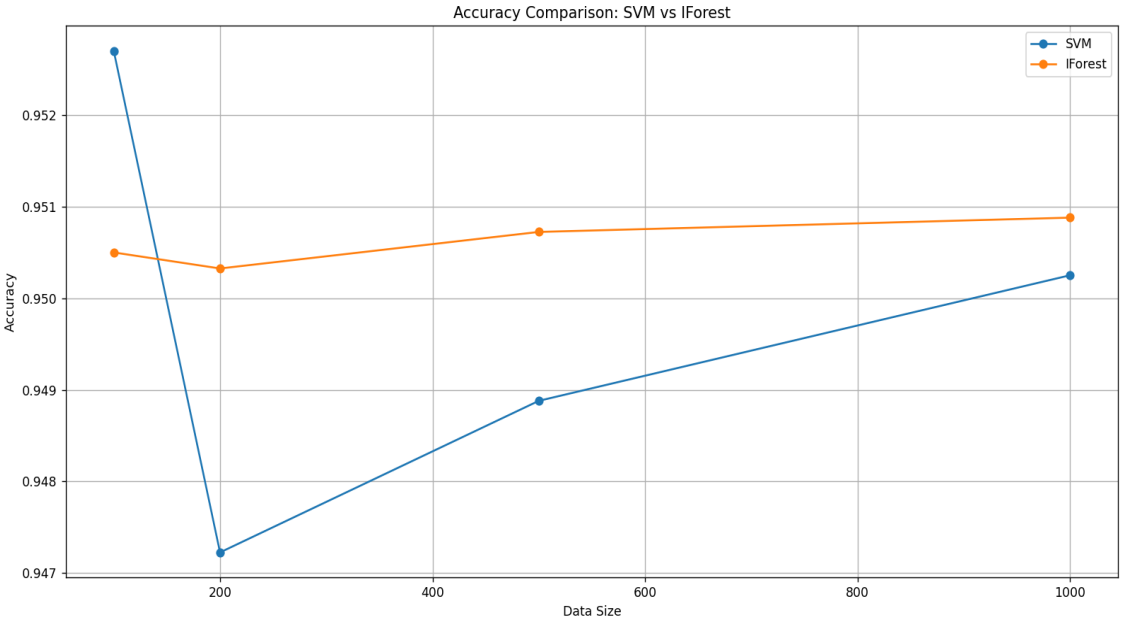
Εδώ φαίνεται η διαφορά στα δεδομένα του κάθε αλγορίθμου παρατηρούμε ότι όντως ο SVM έχει περισσότερα καλά δεδομένα σε σύγκριση με τον Isolation Forest.

ΠΕΙΡΑΜΑ #2

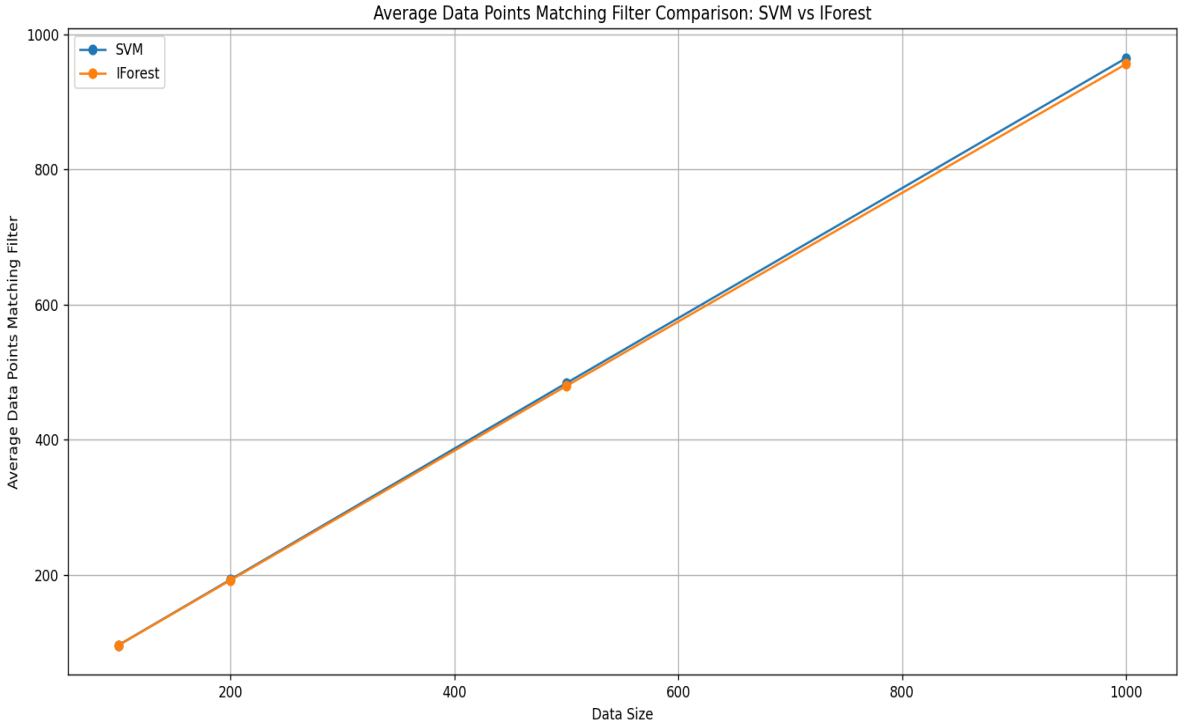
Στο δεύτερο πείραμα, έγινε μια προσαρμογή στο μέγεθος του buffer (w) εντός του μηχανισμού Outlier Buffer. Η αρχική τιμή του w αυξήθηκε από 100 σε 200, διερευνώντας την επίδραση ενός μεγαλύτερου buffer στην απόδοση του One-Class SVM και του Isolation Forest στη διαχείριση και προσαρμογή των outliers.

- **Παράμετροι:**

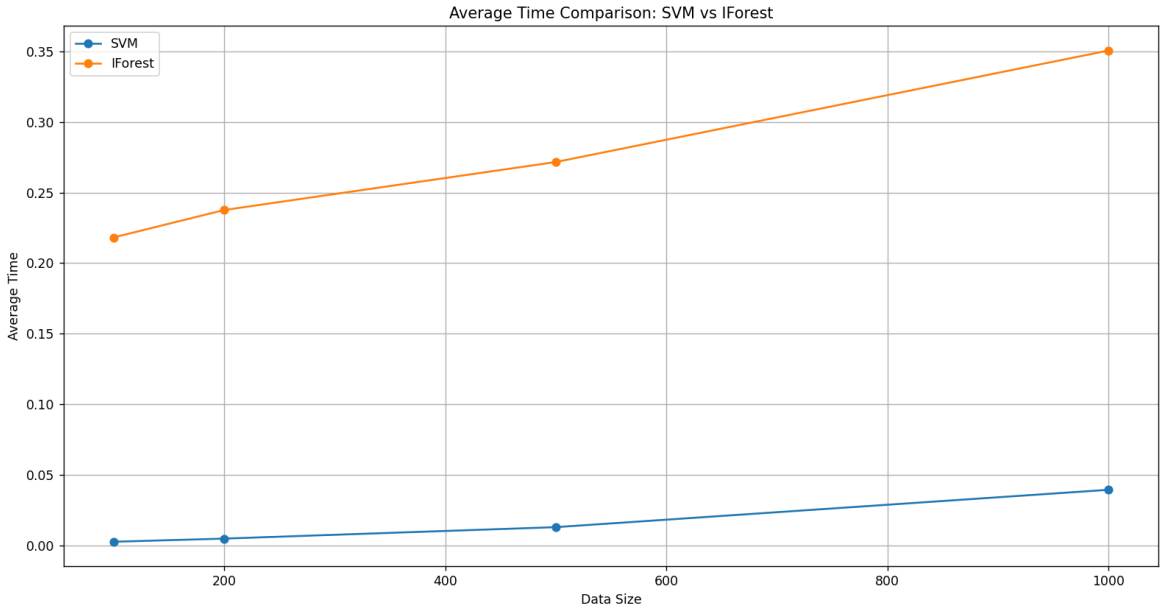
W=200
Low=0
High=30



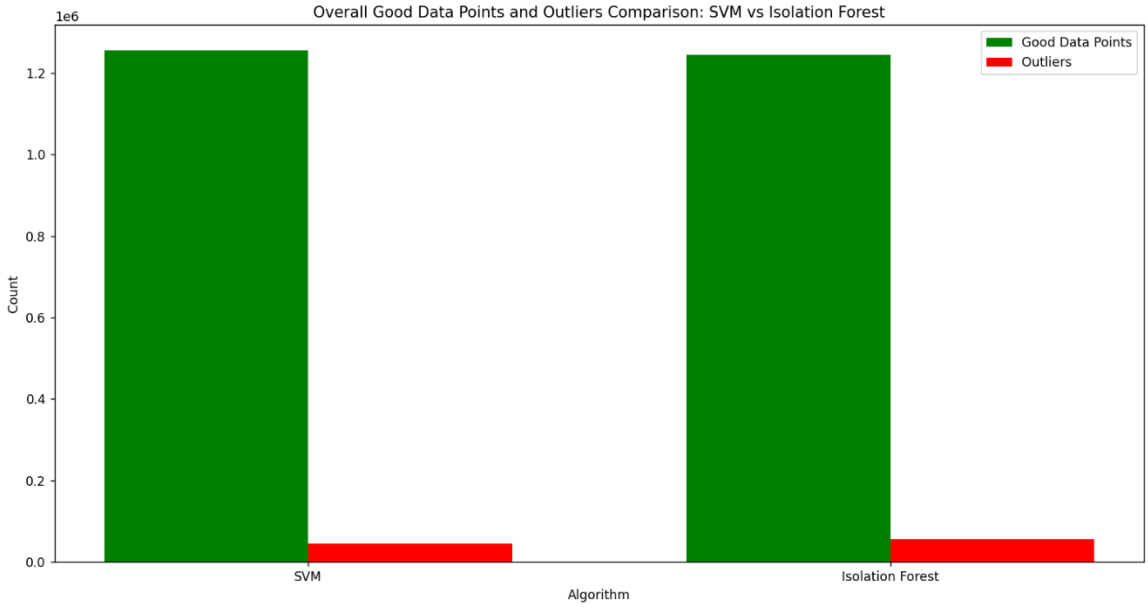
Το γράφημα αυτό είναι παρόμοιο με το γράφημα του πρώτου πειράματος με βάση την ακρίβεια. Μπορούμε να διαπιστώσουμε ότι η ακρίβεια του κάθε μοντέλου δεν έχει να κάνει τόσο με το μέγεθος του buffer και δεν υπάρχουν κάποιες επιπτώσεις στα αποτελέσματα.



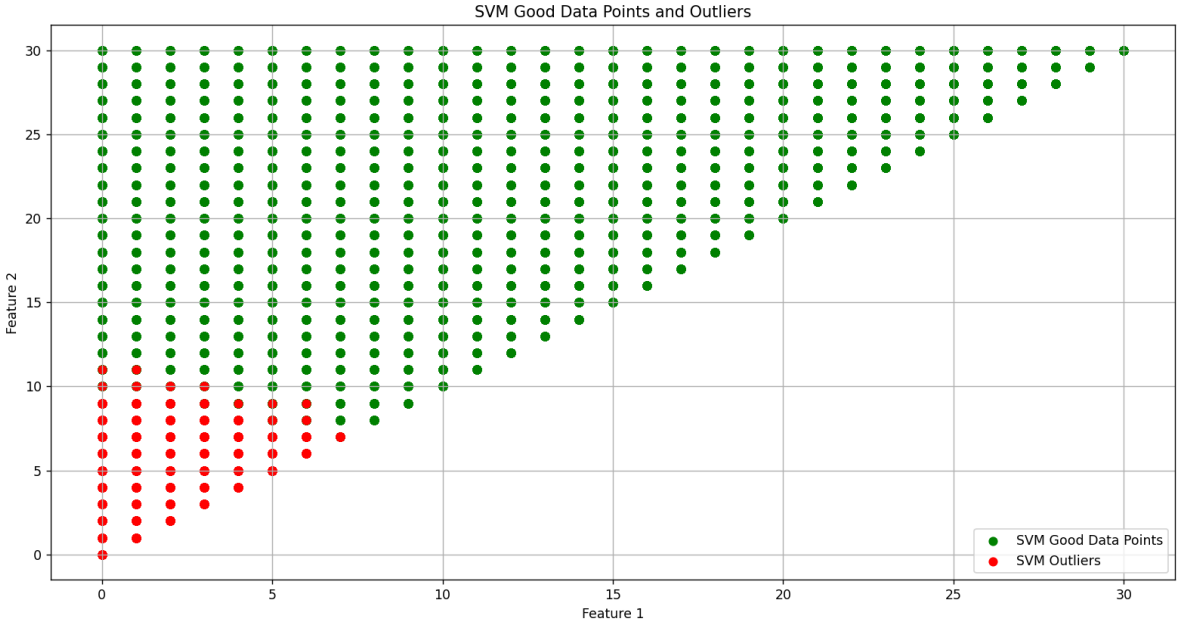
Το ίδιο και στο γράφημα για τα δεδομένα που αντιστοιχούν στο φίλτρο. Τα αποτελέσματα σε αυτό το κομμάτι είναι ίδια με το πρώτο πείραμα.

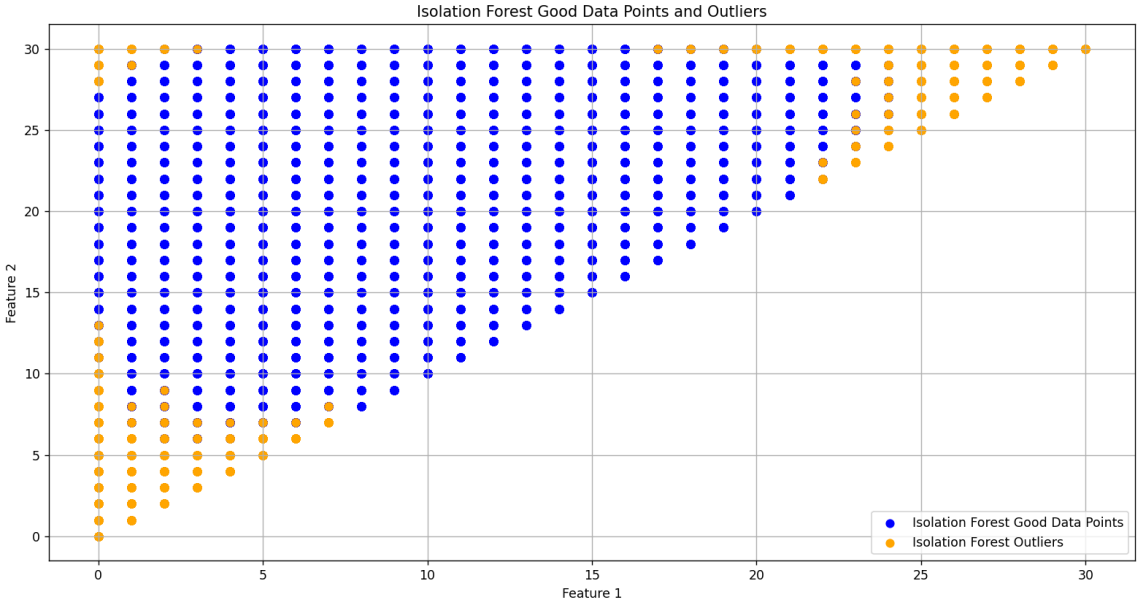


Σε αυτό το γράφημα παρατηρούμε ότι τα αποτελέσματα σχετικά με τον μέσο χρόνο κάθε αλγορίθμου είναι το ίδιο με το πρώτο πείραμα. Φαίνεται δηλαδή ότι ο αλγόριθμος SVM είναι αρκετά πιο γρήγορος απ' ότι ο Isolation Forest.



Το παραπάνω γράφημα έχει εξίσου παρόμοια αποτελέσματα με του πρώτου πειράματος σχετικά με τα δεδομένα που έχουν διαχειριστεί και αναγνωρίσει οι αλγόριθμοι.



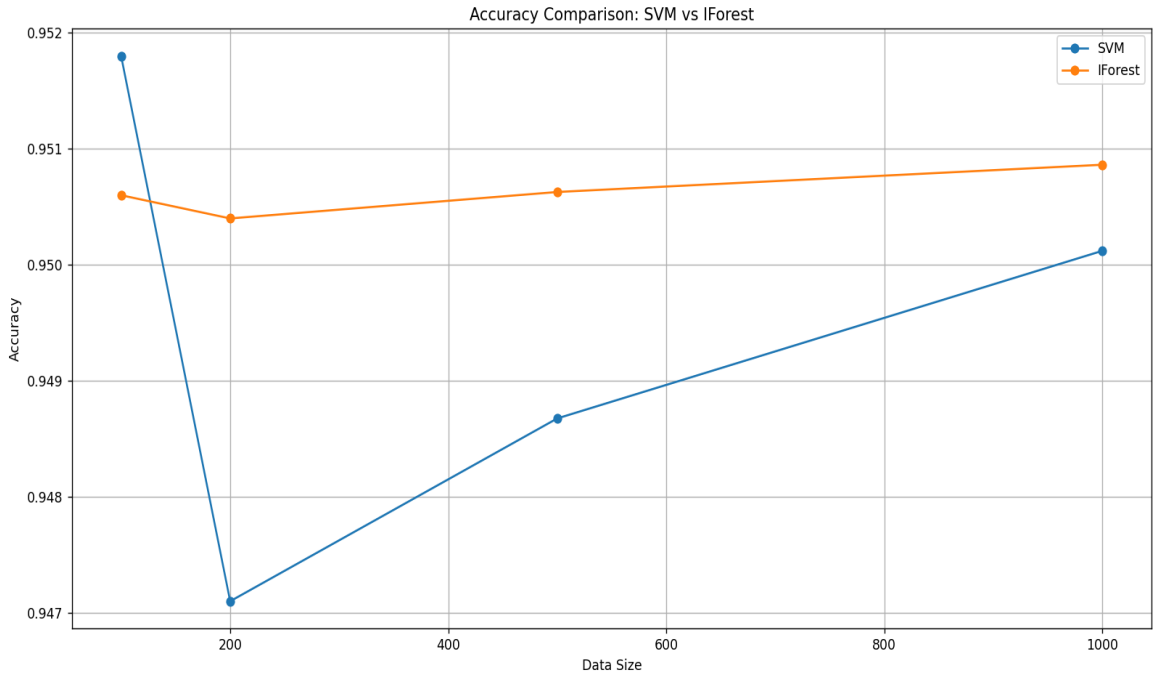


Απ' ότι φαίνεται στα δυο γραφήματα τα αποτελέσματα είναι αρκετά ίδια με αυτά του πρώτου πειραματος.

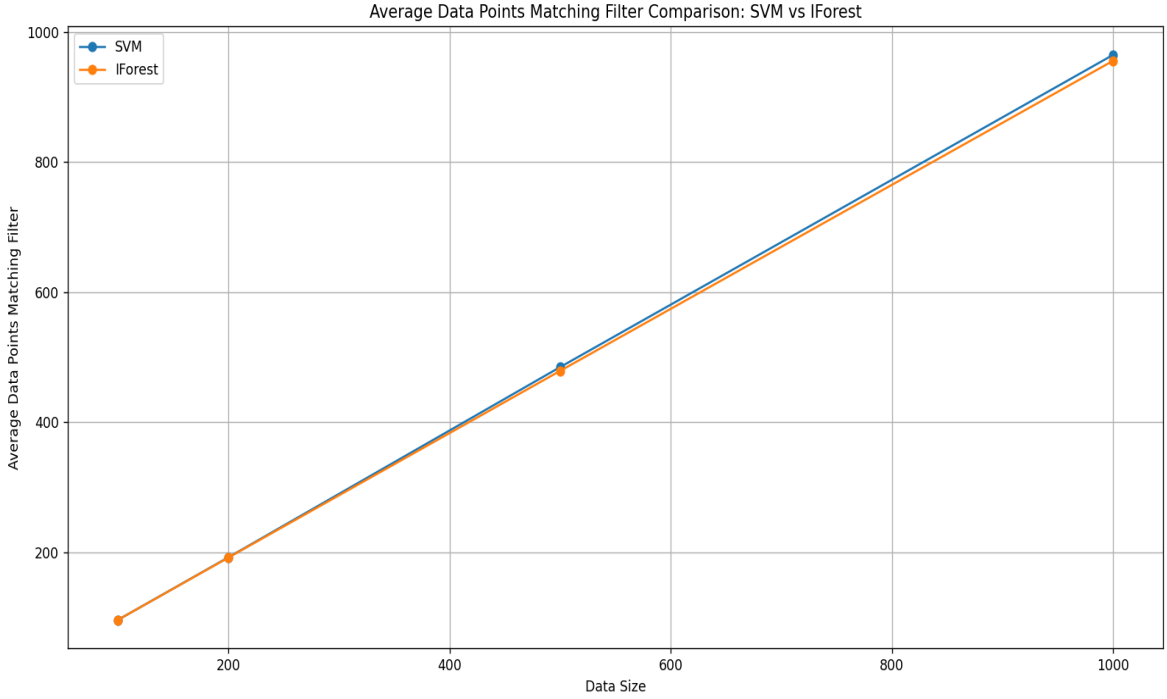
Στο τρίτο πείραμα, έγινε μια προσαρμογή στο μέγεθος του buffer (w) εντός του μηχανισμού Outlier Buffer. Η τιμή του w αυξήθηκε από 200 σε 300, διερευνώντας την επίδραση ενός ακόμη μεγαλύτερου buffer στην απόδοση του One-Class SVM και του Isolation Forest στη διαχείριση και προσαρμογή των outliers.

- **Παράμετροι:**

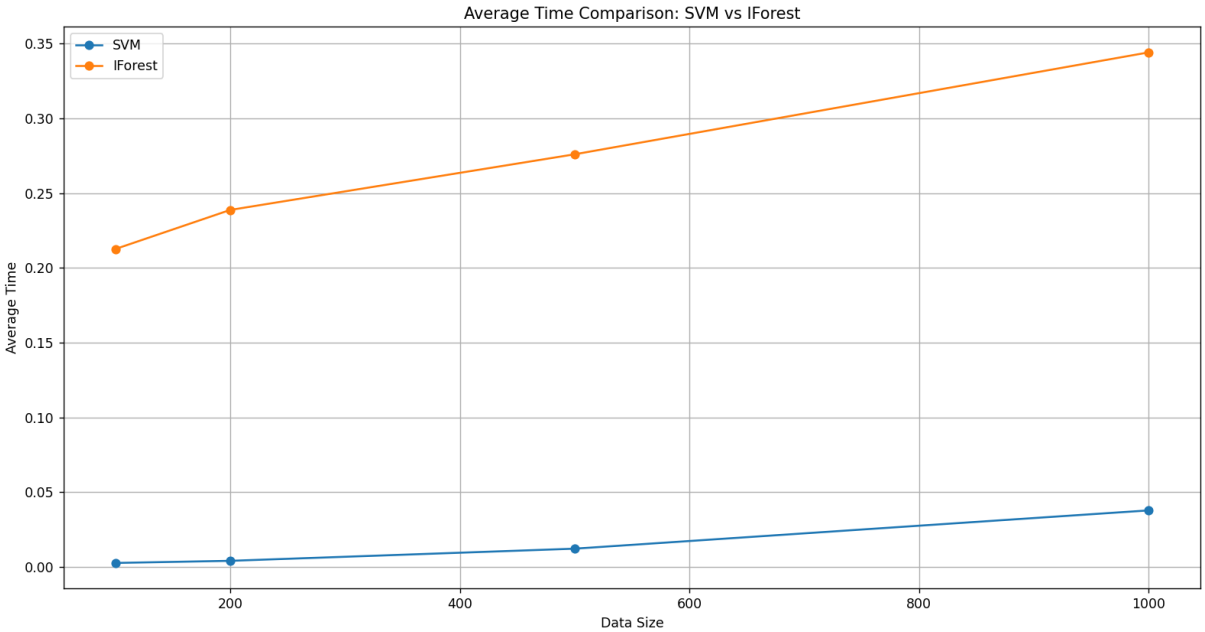
W=300
Low=0
High=30



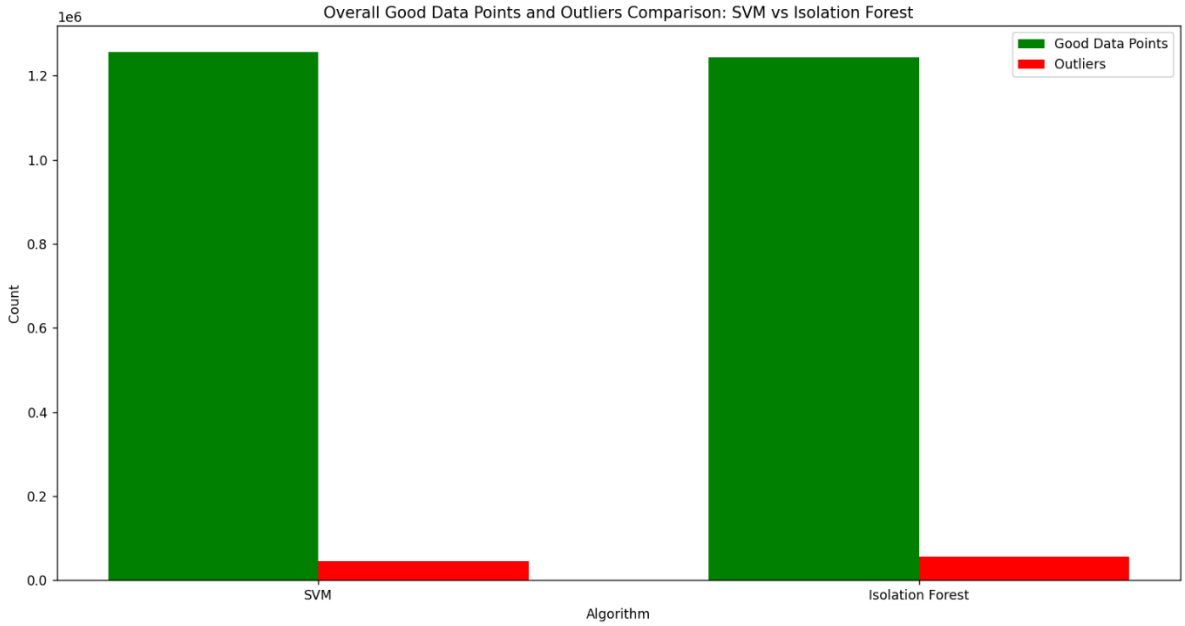
Στο παραπάνω γράφημα παρατηρούμε ότι η ακρίβεια και των δυο αλγορίθμων έχει το ίδιο μοτίβο με τα προηγούμενα πειράματα.



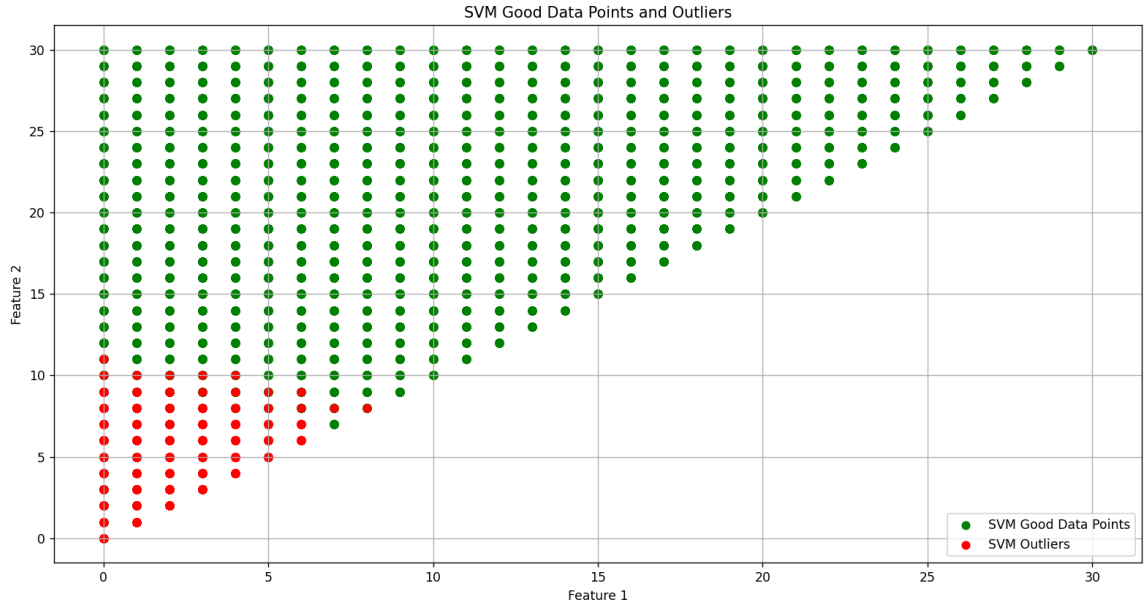
Το ίδιο και στο γράφημα για τα δεδομένα που αντιστοιχούν στο φίλτρο. Τα αποτελέσματα σε αυτό το κομμάτι είναι ίδια με το πρώτο και το δεύτερο πείραμα.

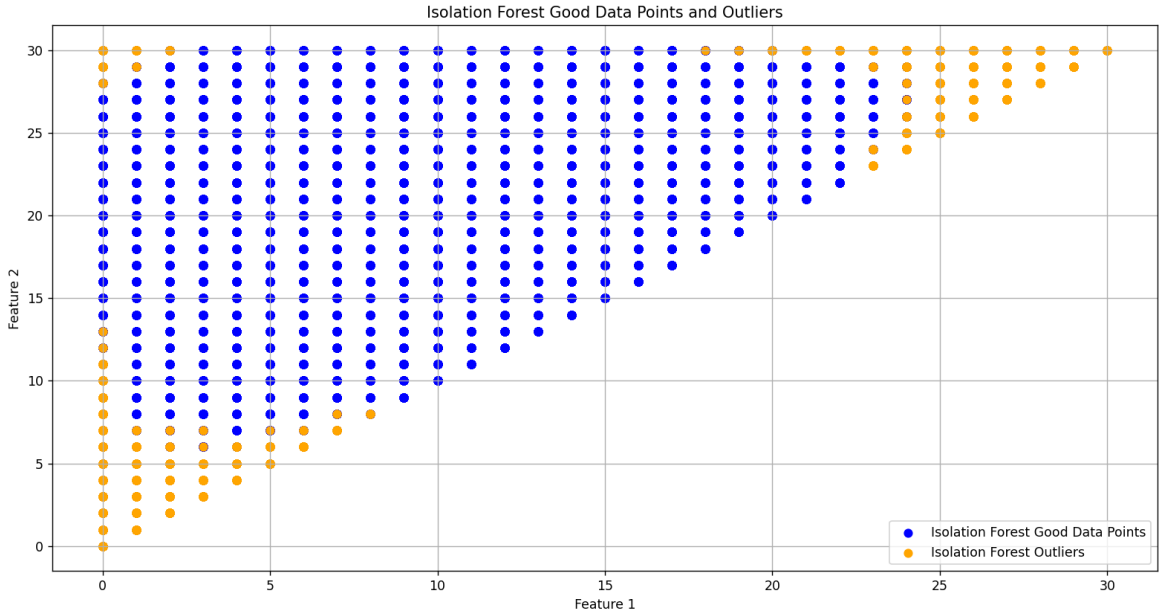


Στο γράφημα για τον μέσο χρόνο παρατηρούμε ότι είναι παρόμοιο με τα προηγούμενα πειράματα με τον SVM πάντα να είναι εξαιρετικά πιο γρήγορος από τον Isolation Forest.



Σε αυτό το γράφημα τα δεδομένα που έχουν διαχειριστεί και οι δυο αλγόριθμοι είναι παρόμοια με τα προηγούμενα γραφήματα.



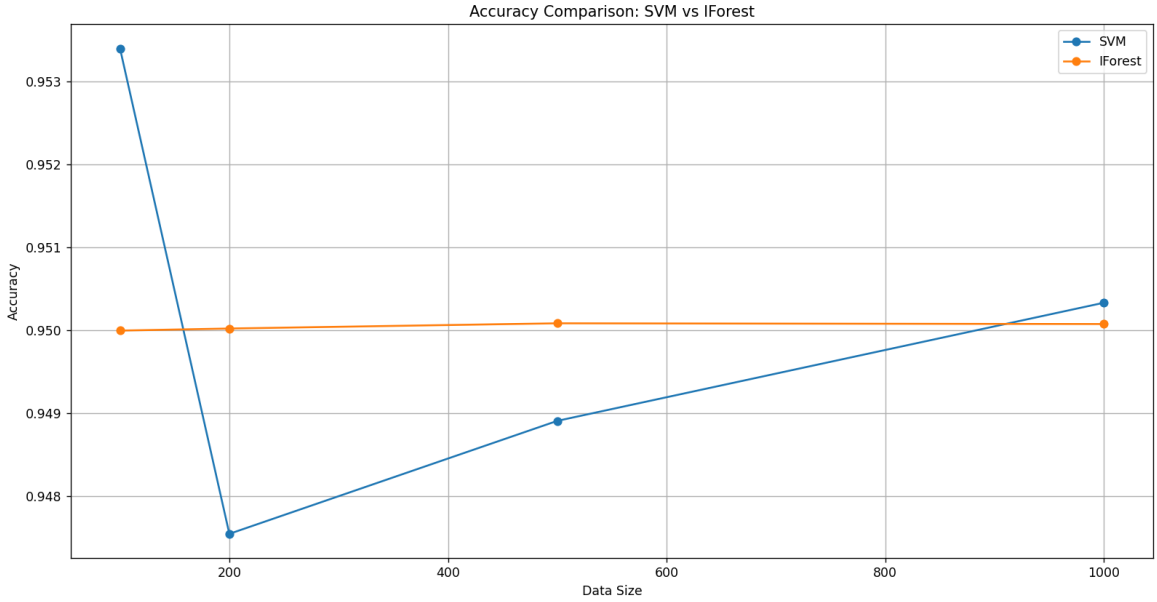


Απ’ ότι φαίνεται στα δυο γραφήματα τα αποτελέσματα είναι αρκετά παρόμοια με αυτά των προηγουμένων πειραμάτων.

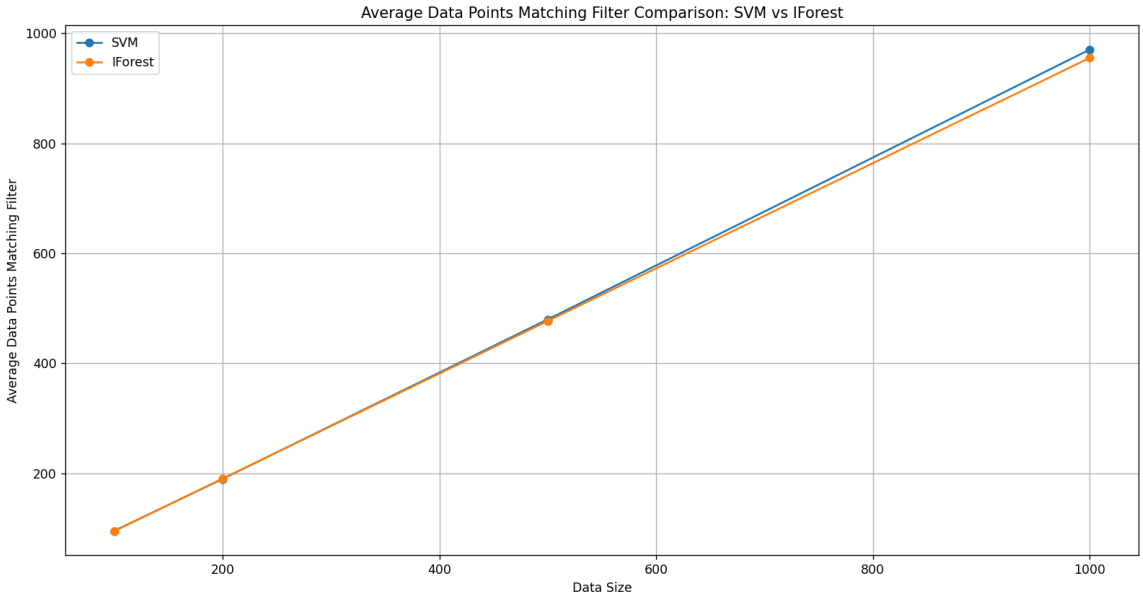
Στο τέταρτο πείραμα, έγινε μια προσαρμογή στο μέγεθος του buffer (w) εντός του μηχανισμού Outlier Buffer. Η τιμή του w γύρισε πίσω στην αρχική τιμή της δηλαδή 100. Μια ακόμη σημαντική αλλαγή έγινε στα low-high διαστήματα, το όριο το επεκτείναμε από 0-30 σε 0-100, αλλάζοντας το εύρος τιμών δεδομένων.

- **Παράμετροι:**

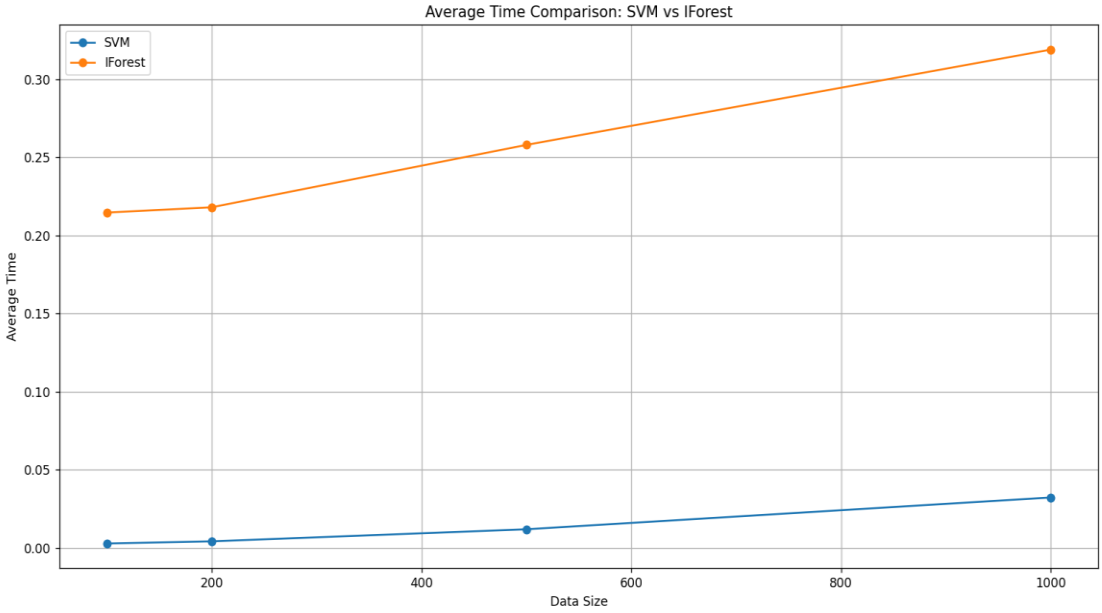
W=100
Low=0
High=100



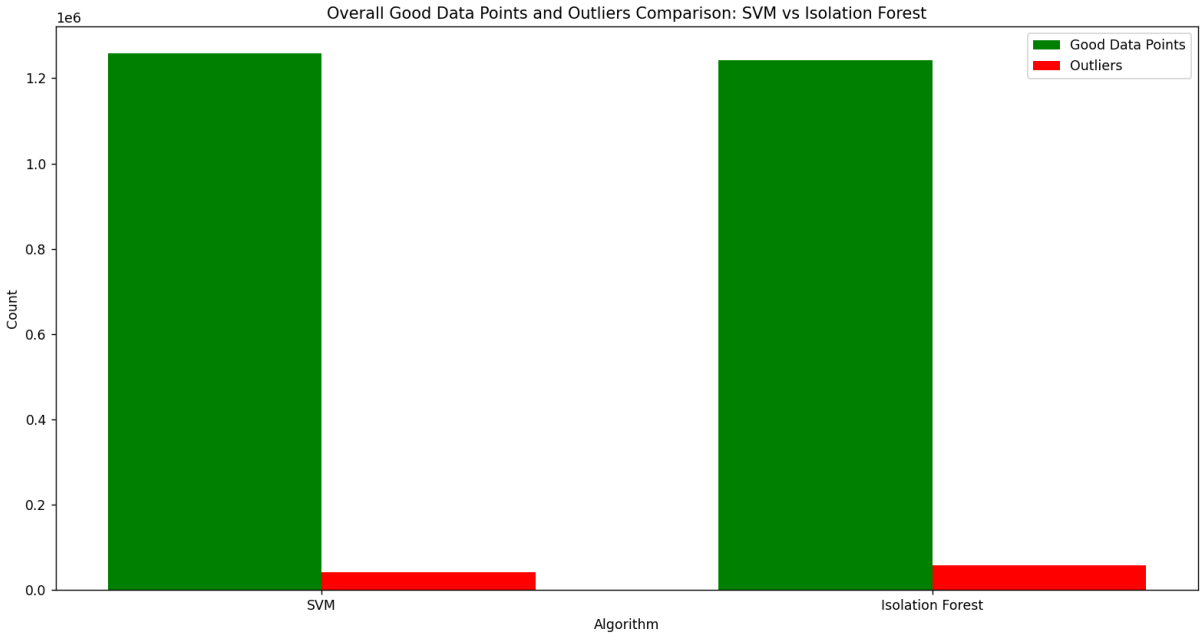
Στο παραπάνω γράφημα παρατηρούμε ότι ακολουθεί το ίδιο μοτίβο με τα προηγούμενα με την μόνη διαφορά η ακρίβεια του SVM να φτάνει και να ξεπερνάει την ακρίβεια του Isolation Forest. Πρέπει να σημειωθεί ότι δεν υπάρχει τόσο μεγάλη διαφορά στην ακρίβεια των δυο αλγορίθμων αλλά αυτή η μικρή αλλαγή είναι φανερή.



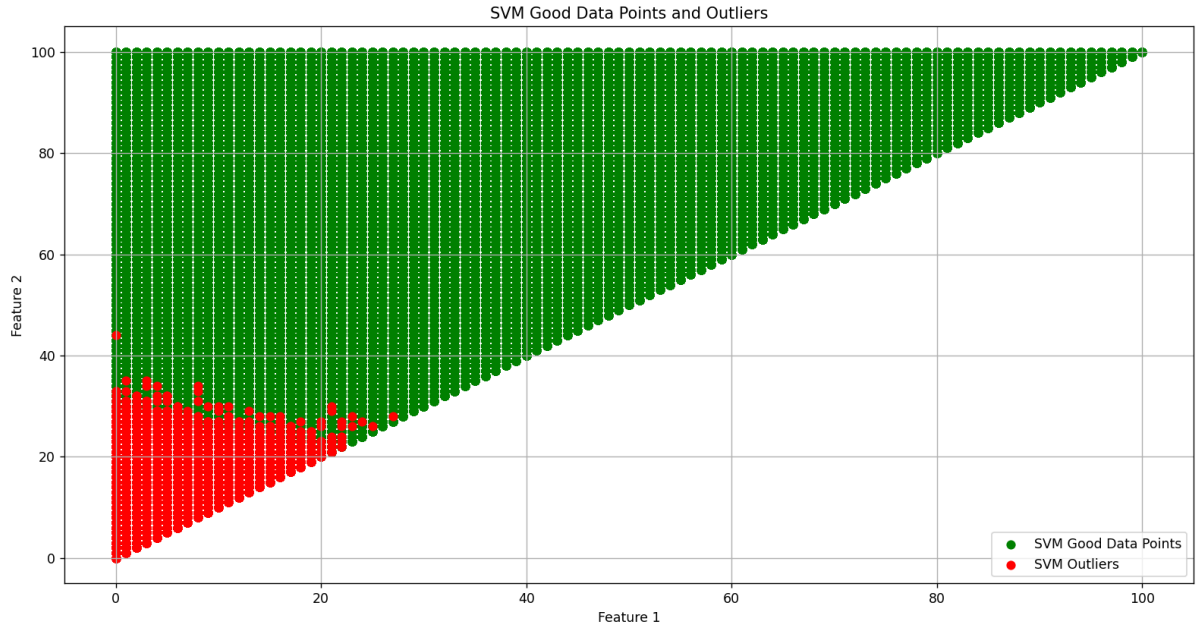
Σε αυτό το γράφημα παρατηρούμε ότι τα δεδομένα που αντιστοιχούν στο φίλτρο είναι παρόμοια με τα προηγούμενα πειράματα.

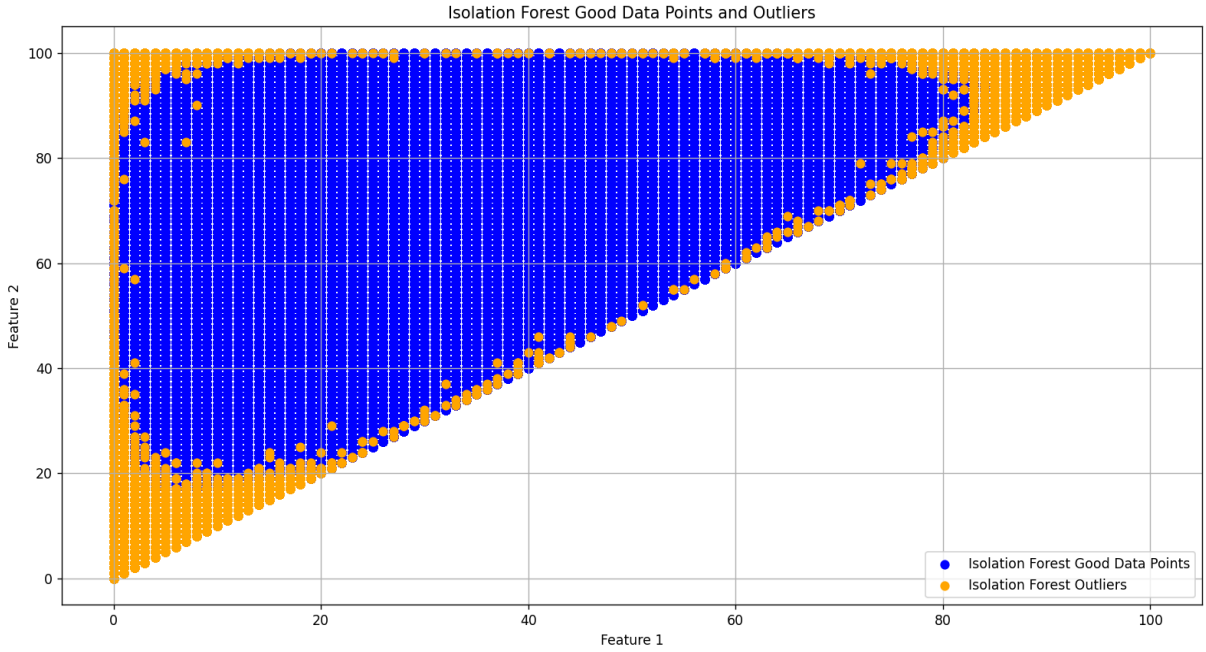


Στο γράφημα για τον μέσο χρόνο παρατηρούμε ότι είναι παρόμοιο με τα προηγούμενα πειράματα με τον SVM πάντα να είναι εξαιρετικά πιο γρήγορος από τον Isolation Forest. Πρέπει να σημειωθεί βέβαια ότι ο χρόνος του Isolation Forest βελτιώθηκε για το τελευταίο μέγεθος δεδομένων που είναι 1000 από 0.35 δευτερόλεπτα είναι πιο κοντά στα 0.30 δευτερόλεπτα.



Σε αυτό το γράφημα τα δεδομένα που έχουν αναγνωρίσει και διαχειριστεί και οι δυο αλγόριθμοι είναι παρόμοια με τα προηγούμενα γραφήματα.



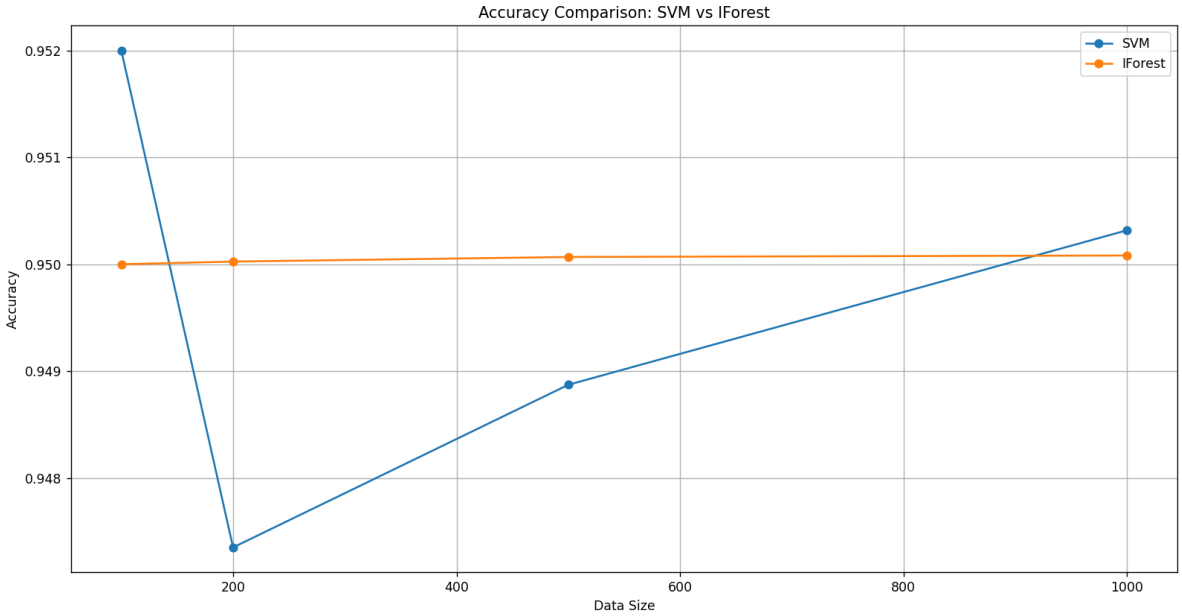


Στα δυο αυτά γραφήματα φαίνεται αρκετά η διαφορά με τα προηγούμενα όπου είχαμε τα όρια σε 0-30. Επειδή σε αυτό το πείραμα έχουν αλλάξει τα όρια σε 0-100 αλλάζει και το εύρος τιμών των δεδομένων αλλά υπάρχει και ένα ευρύτερο φάσμα τιμών για τα χαρακτηριστικά του συνόλου δεδομένων μας. Αυτή η επέκταση στους περιορισμούς του διαστήματος επιτρέπει τη δημιουργία ενός πιο διαφορετικού συνόλου σημείων δεδομένων.

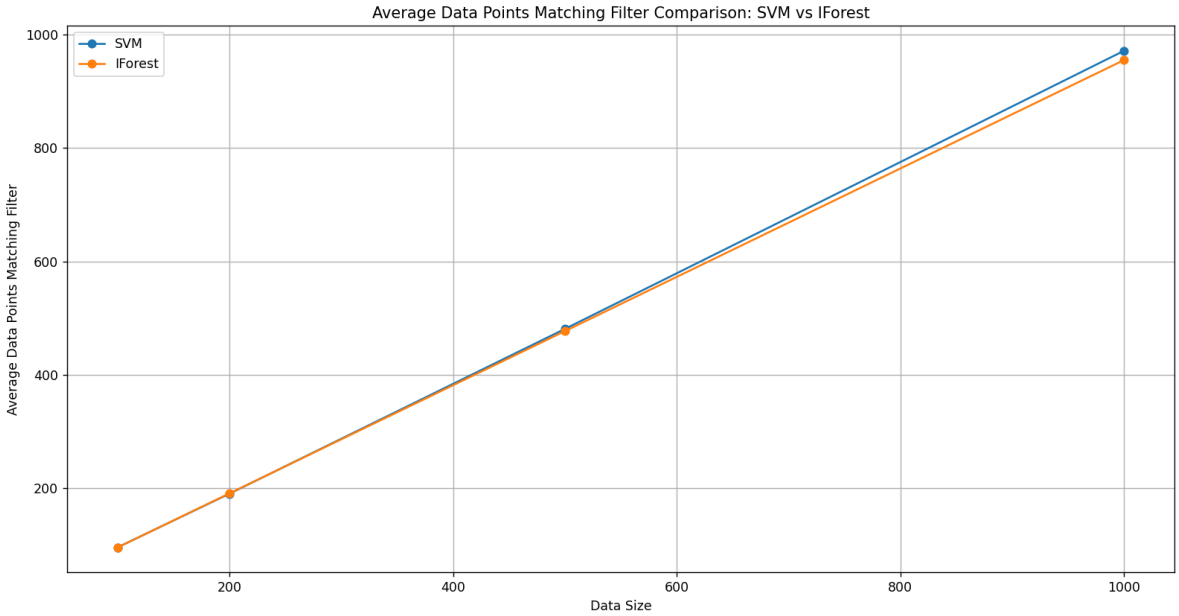
Στο πέμπτο πείραμα, έγινε μια προσαρμογή στο μέγεθος του buffer (w) εντός του μηχανισμού Outlier Buffer. Η τιμή του w αυξήθηκε από 100 σε 200, διερευνώντας την επίδραση ενός ακόμη μεγαλύτερου buffer. Τα low-high διαστήματα παρέμειναν στην ίδια τιμή με το προηγούμενο πείραμα δηλαδή, το όριο είναι και πάλι 0-100.

- **Παράμετροι:**

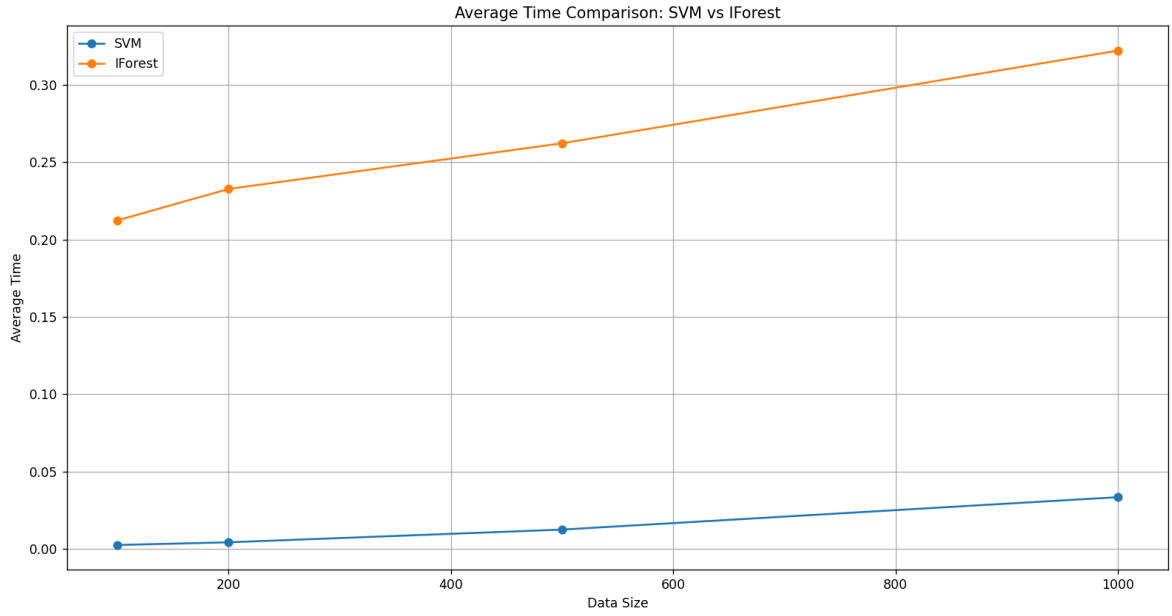
W=200
Low=0
High=100



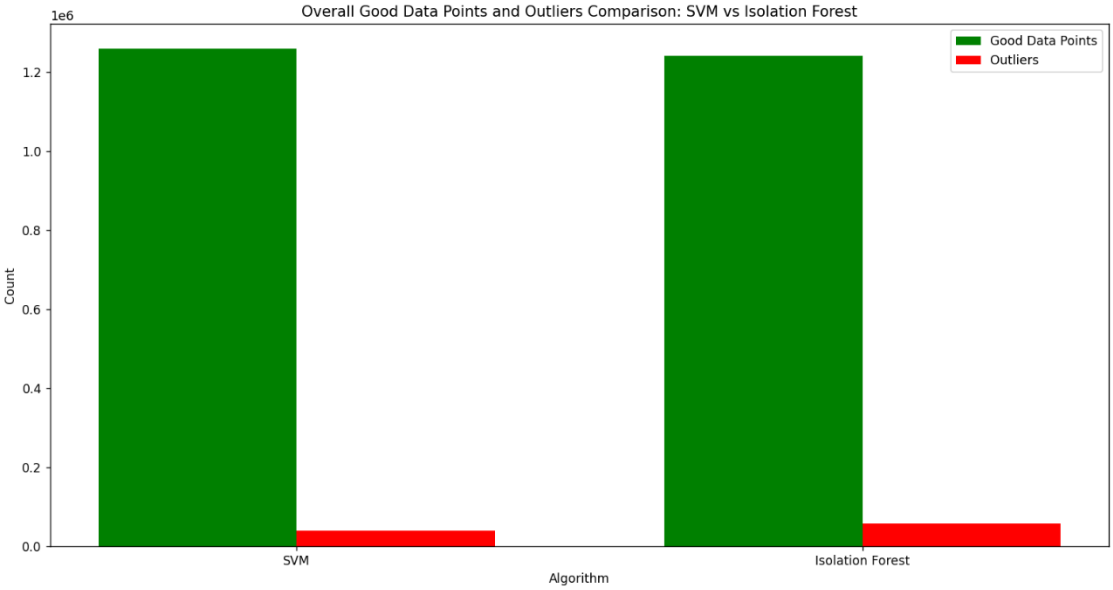
Το γράφημα ακρίβειας των δυο αλγορίθμων είναι παρόμοιο με το γράφημα του προηγούμενου πειράματος.



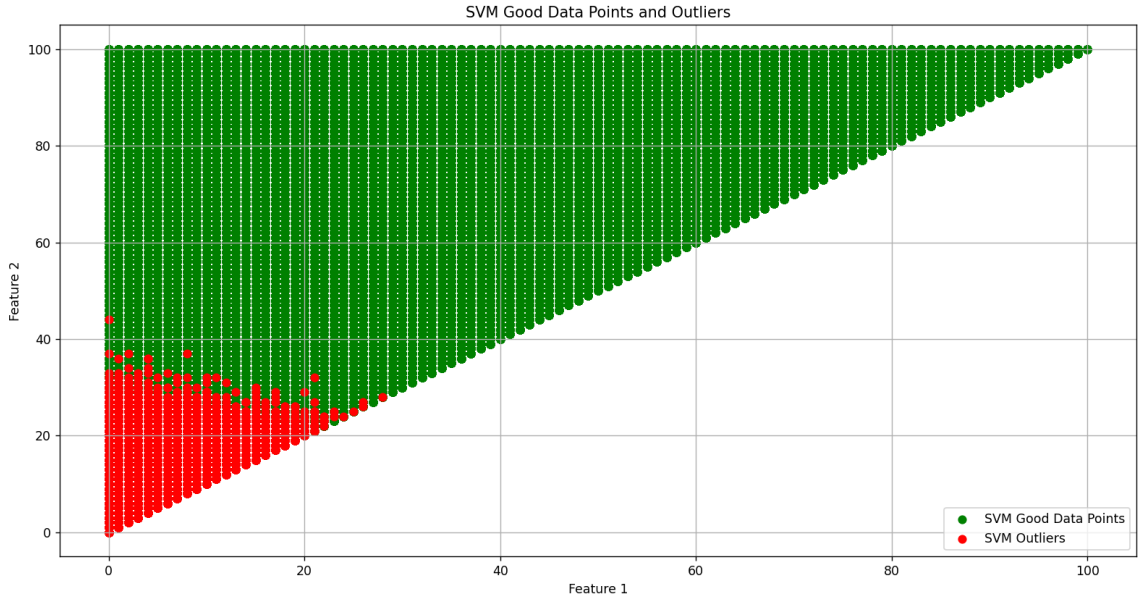
Σε αυτό το γράφημα παρατηρούμε ότι τα δεδομένα που αντιστοιχούν στο φίλτρο είναι παρόμοια με τα προηγούμενα πειράματα.

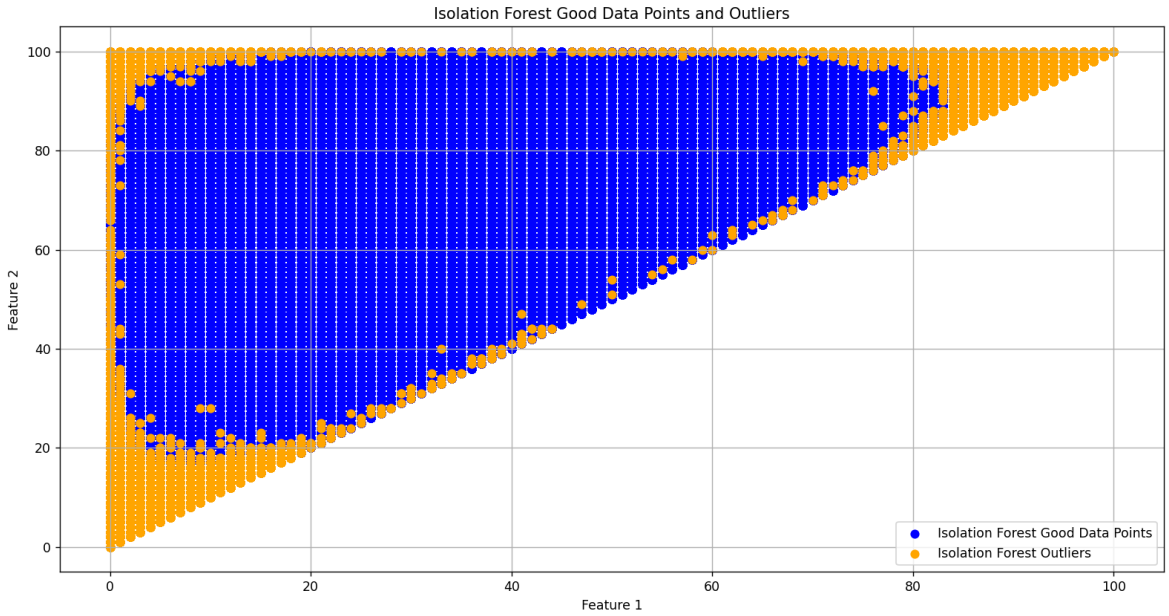


Στο γράφημα μέσου χρόνου παρατηρούμε ότι είναι παρόμοιο με το γράφημα του προηγούμενου πειράματος. Ο αλγόριθμος SVM παραμένει να είναι πιο γρήγορος από τον Isolation Forest.



Σε αυτό το γράφημα τα δεδομένα που έχουν αναγνωρίσει και διαχειριστεί και οι δυο αλγόριθμοι είναι παρόμοια με τα προηγούμενα γραφήματα.



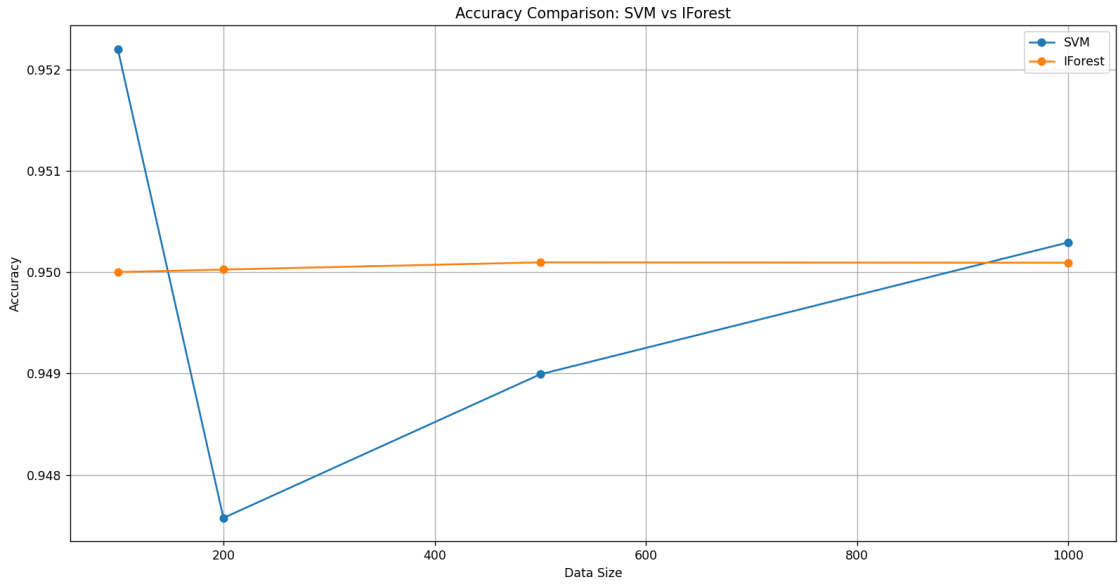


Σε αυτά τα δυο γραφήματα παρατηρούμε ότι είναι αρκετά παρόμοια με τα γραφήματα του τεταρτου πειραματος.

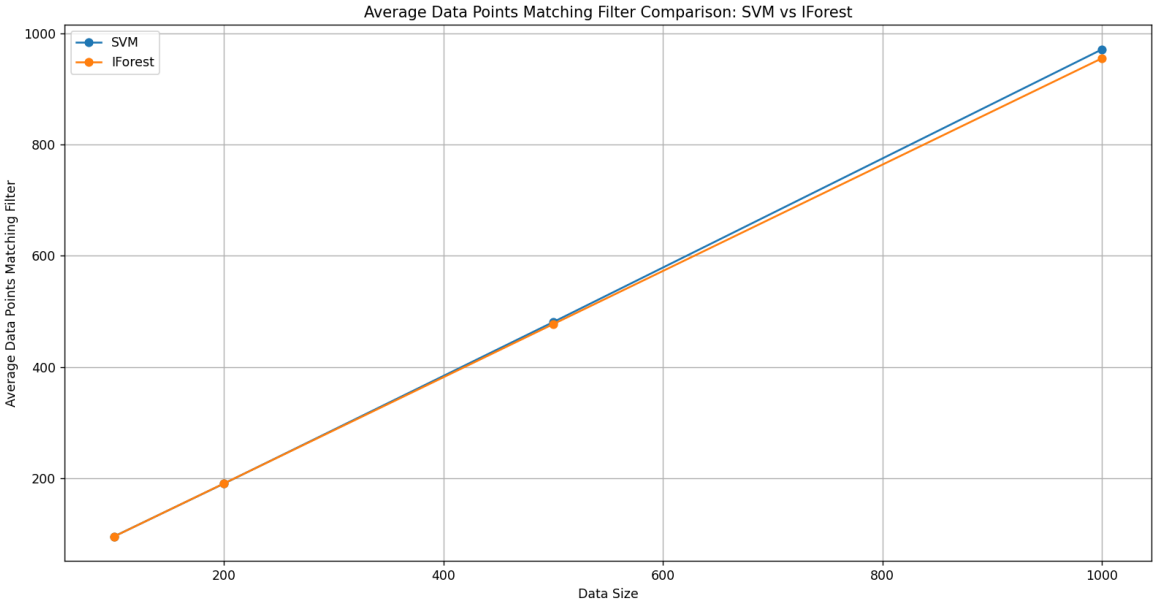
Στο έκτο και τελευταίο πείραμα, έγινε μια προσαρμογή στο μέγεθος του buffer (w) εντός του μηχανισμού Outlier Buffer. Η τιμή του w αυξήθηκε από 200 σε 300, διερευνώντας την επίδραση ενός ακόμη μεγαλύτερου buffer. Τα low-high διαστήματα παρέμειναν στην ίδια τιμή με το προηγούμενο πείραμα δηλαδή, το όριο είναι και πάλι 0-100.

- **Παράμετροι:**

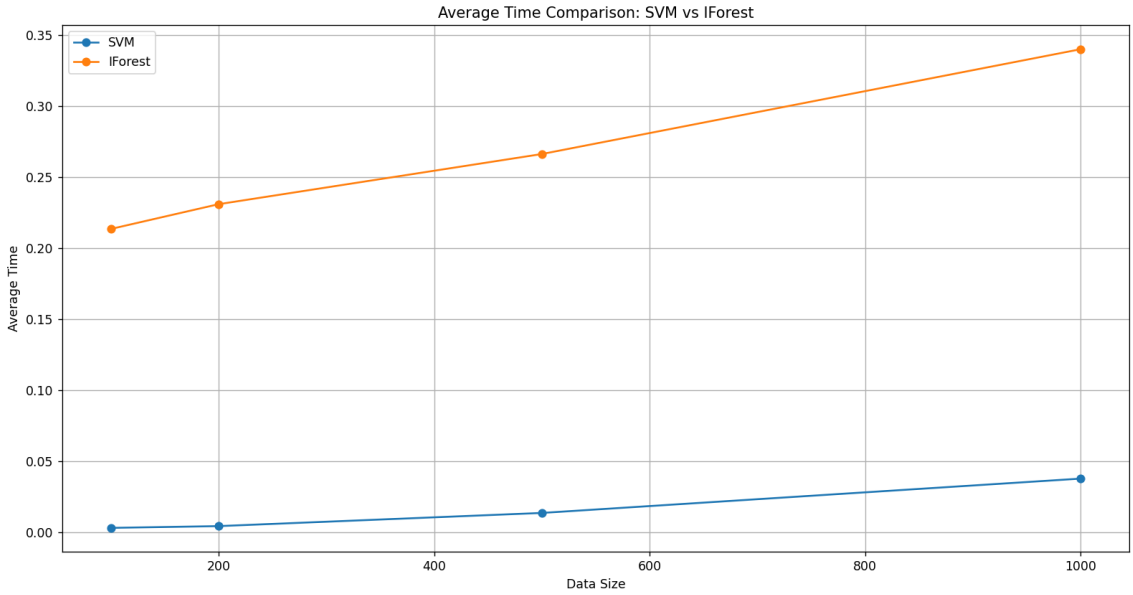
W=300
Low=0
High=100



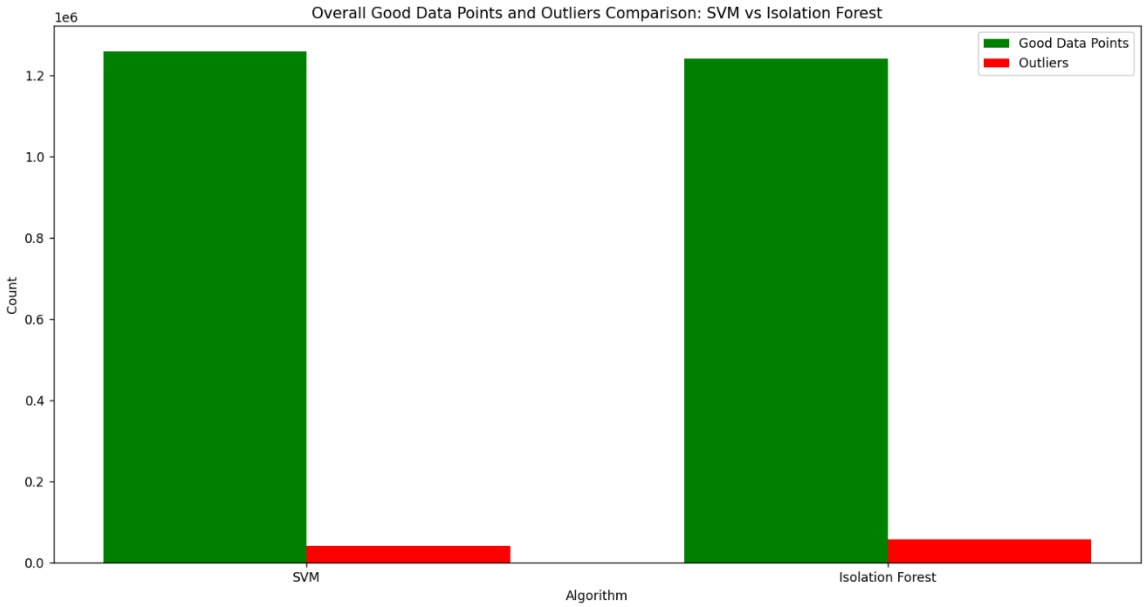
Στο παραπάνω γράφημα παρατηρούμε ότι η ακρίβεια των αλγόριθμων έχει το ίδιο μοτίβο με τα προηγούμενα πειραματα.



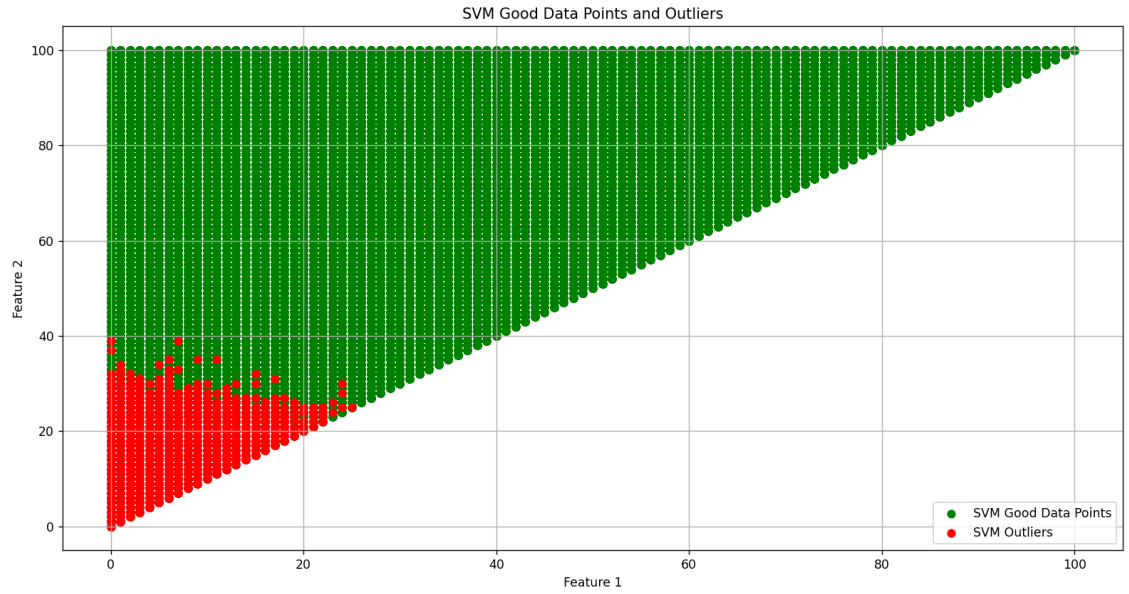
Σε αυτό το γράφημα παρατηρούμε ότι τα δεδομένα που αντιστοιχούν στο φίλτρο είναι παρόμοια με τα προηγούμενα πειράματα. Ίσως θα μπορούσε να πει κάποιος ότι φαίνεται περισσότερο η διαφορά στο τελευταίο μέγεθος των δεδομένων σε σύγκριση με τα προηγούμενα γραφήματα.

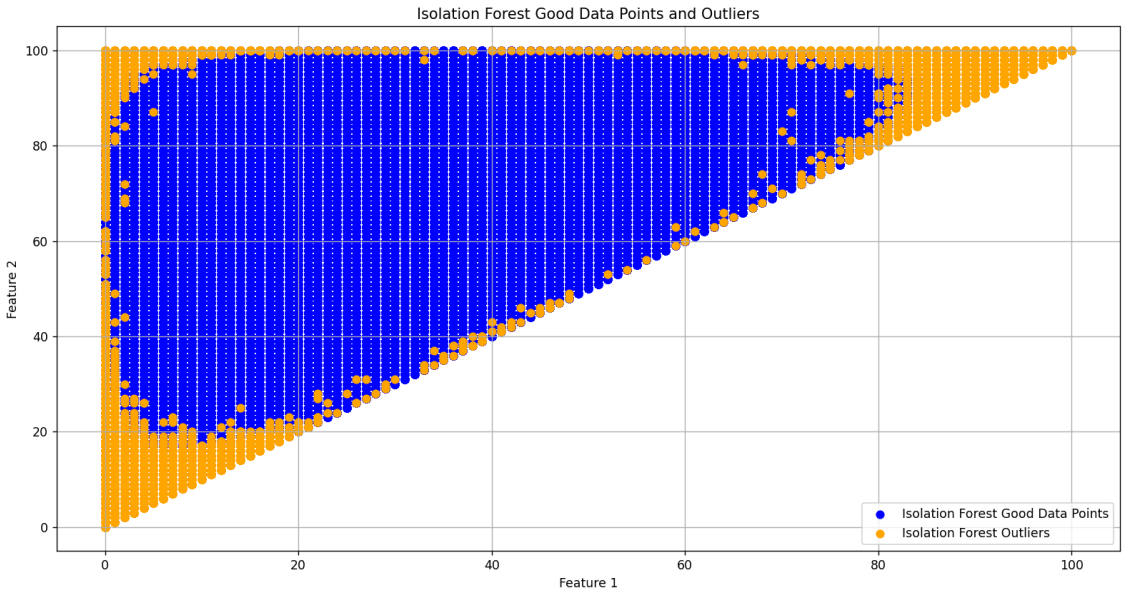


Στο γράφημα μέσου χρόνου παρατηρούμε ότι είναι παρόμοιο με το γράφημα του προηγούμενου πειράματος. Ο αλγόριθμος SVM παραμένει να είναι πιο γρήγορος από τον Isolation Forest.



Σε αυτό το γράφημα τα δεδομένα που έχουν αναγνωρίσει και διαχειριστεί και οι δυο αλγόριθμοι είναι παρόμοια με τα προηγούμενα γραφήματα.





Σε αυτά τα δυο γραφήματα παρατηρούμε ότι είναι αρκετά παρόμοια με τα γραφήματα του προηγούμενου πειράματος.

ΚΕΦΑΛΑΙΟ 6 : ΣΥΜΠΕΡΑΣΜΑΤΑ

ΑΠΟΤΕΛΕΣΜΑΤΑ & ΤΕΛΙΚΟ ΣΥΜΠΕΡΑΣΜΑ

Η πειραματική φάση είχε ως στόχο να εμβαθύνει στις περιπλοκές της αναγνώρισης και διαχείρισης ανωμαλιών χρησιμοποιώντας αλγόριθμους SVM One-Class και Isolation Forest. Μέσα από μια σειρά από προσεκτικά σχεδιασμένα πειράματα, διερευνήσαμε τις απαντήσεις των αλγορίθμων σε διαφορετικά σενάρια, προσαρμοζόμενοι σε δυναμικές αλλαγές στα σύνολα δεδομένων και προσαρμόζοντας τις σημαντικές παραμέτρους.

Σε αυτήν την ενότητα, παρουσιάζουμε μια περιεκτική ανάλυση των αποτελεσμάτων που προέκυψαν από τα έξι πειράματα. Τα αποτελέσματα προσφέρουν πολύτιμες γνώσεις για τη συμπεριφορά αυτών των αλγορίθμων υπό διαφορετικές συνθήκες, βοηθώντας να κατανοήσουμε στα δυνατά τους σημεία, τους περιορισμούς και τις πιθανές οδούς βελτίωσης.

Low,high=0,30

W	ACCURACY	TIME	DATA MATCHING
100	0.9497	0.0164	434.28
200	0.9497	0.0175	434.34
300	0.9495	0.0180	434.27

W	ACCURACY	TIME	DATA MATCHING
100	0.9507	0.269	430.79
200	0.9507	0.280	430.70
300	0.9506	0.279	430.75

Low,high=0,100			
W	ACCURACY	TIME	DATA MATCHING
100	0.9499	0.0215	434.07
200	0.9499	0.0214	434.06
300	0.9502	0.0205	434.32

W	ACCURACY	TIME	DATA MATCHING
100	0.9501	0.325	429.40
200	0.9501	0.303	429.56
300	0.9501	0.289	429.42

Στους παραπάνω πίνακες έχουμε τα αποτελέσματα για κάθε “w” που χρησιμοποιήθηκε στα όρια 0-30 και στα όρια 0-100. Το κάθε αποτέλεσμα μας δείχνει τον μέσο όρο στην ακρίβεια, στον χρόνο και στην αντιστοίχιση (φιλτράρισμα) των δεδομένων για κάθε αλγόριθμο. Πρέπει να σημειωθεί ότι στους δυο πίνακες τα πράσινα αντιστοιχούν με τον αλγόριθμο One-Class SVM και τα μπλε με τον αλγόριθμο Isolation Forest.

Παρατηρούμε ότι ο αλγόριθμος Isolation Forest έχει λίγο παραπάνω ακρίβεια από τον One-Class SVM αλλά χωρίς να έχει ιδιαίτερη σημασία γιατί η διαφορά τους στην ακρίβεια είναι μηδαμινή. Μπορούμε να συμπεράνουμε βέβαια ότι και οι δυο αλγόριθμοι έχουν πολύ καλή ακρίβεια.

Στην συνέχεια θα αναλύσουμε τον μέσο χρόνο που χρειάστηκαν οι αλγόριθμοι για την αναγνώριση και διαχείριση κάθε σημείου δεδομένων. Από ότι φαίνεται στους πίνακες και όπως είδαμε και στα πειράματα στο προηγούμενο κεφάλαιο η διαφορά των δυο αλγορίθμων στο χρόνο είναι μεγάλη. Ο αλγόριθμος One-Class SVM είναι εξαιρετικά πιο γρήγορος από τον Isolation Forest. Ο μειωμένος χρόνος επεξεργασίας του SVM υποδηλώνει την καταλληλότητά του για εφαρμογές όπου η αναγνώριση ανωμαλιών σε πραγματικό χρόνο είναι επιτακτική. Αυτή η αποτελεσματικότητα θα μπορούσε να αποδοθεί στα χαρακτηριστικά του αλγορίθμου SVM, όπως η γραμμική πολυπλοκότητά του ως προς τον αριθμό των σημείων δεδομένων.

Τέλος για την αντιστοιχία των δεδομένων παρατηρούμε ότι το SVM εμφανίζει σταθερά μεγαλύτερο αριθμό σημείων δεδομένων που ταιριάζουν με τα κριτήρια φιλτραρίσματος σε σύγκριση με το Isolation Forest. Η μέτρηση της αντιστοίχισης δεδομένων χρησιμεύει ως κρίσιμος δείκτης της ακρίβειας του αλγορίθμου στον εντοπισμό και τη διατήρηση των σχετικών σημείων δεδομένων, ενώ φιλτράρει τους outliers.

Συμπερασματικά, τα αποτελέσματα των πειραμάτων δείχνουν σταθερά τον One-Class SVM ως ανώτερη επιλογή για αναγνώριση και διαχείριση ανωμαλιών σε σύγκριση με τον αλγόριθμο Isolation Forest στο μοντέλο που αναπτύξαμε.

ΜΕΛΛΟΝΤΙΚΕΣ ΕΠΕΚΤΑΣΕΙΣ

Ενώ τα τρέχοντα πειράματα παρέχουν πολύτιμες πληροφορίες για τη συγκριτική απόδοση των αλγορίθμων Support Vector Machine (SVM) και Isolation Forest, υπάρχουν αρκετοί τρόποι για μελλοντική εξερεύνηση και βελτίωση:

Parameter Tuning: Μπορεί να πραγματοποιηθεί μια πιο εξαντλητική εξερεύνηση των διαμορφώσεων υπερπαραμέτρων τόσο για το SVM όσο και για το Isolation Forest για τον εντοπισμό βέλτιστων ρυθμίσεων που μπορεί να βελτιώσουν περαιτέρω την απόδοσή τους υπό διαφορετικές συνθήκες.

Ensemble Methods: Μπορεί να γίνει διερεύνηση της σκοπιμότητας του συνδυασμού πολλαπλών αλγορίθμων αναγνώρισης ανωμαλιών σε ένα πλαίσιο συνόλου. Οι μέθοδοι συνόλου μπορούν δυνητικά να αξιοποιήσουν τα δυνατά σημεία διαφορετικών αλγορίθμων, βελτιώνοντας τη συνολική απόδοση του μοντέλου.

Dynamic Adaptation: Να γίνει εξερεύνηση για μηχανισμούς για δυναμική προσαρμογή αλγοριθμικών παραμέτρων με βάση τα μεταβαλλόμενα χαρακτηριστικά δεδομένων. Αυτό θα μπορούσε να περιλαμβάνει προσαρμοστικούς ρυθμούς μάθησης, δυναμικό μέγεθος προσωρινής μνήμης ή άλλες τεχνικές για τη διασφάλιση της προσαρμοστικότητας σε εξελισσόμενα σύνολα δεδομένων.

Visualization Tools: Ανάπτυξη εργαλείων οπτικοποίησης για την παροχή εύχρηστων αναπαραστάσεων των αποτελεσμάτων αναγνώρισης ανωμαλιών. Οι αποτελεσματικές απεικονίσεις μπορούν να βοηθήσουν στην ερμηνεία των αποτελεσμάτων και να διευκολύνουν τη λήψη αποφάσεων σε εφαρμογές πραγματικού κόσμου.

Outlier Labeling: Εξερεύνηση μεθόδων για την παροχή πρόσθετων πληροφοριών σχετικά με τα αναγνωρισμένα outliers, όπως η εκχώρηση ετικετών ή κατηγοριών με βάση τη φύση των ανωμαλιών. Αυτό μπορεί να βοηθήσει στην κατανόηση του πλαισίου και των πιθανών αιτιών περιστατικών των outliers.

Feedback Loop: Μπορεί να εφαρμοστεί ένας βρόχος ανατροφοδότησης που επιτρέπει στους χρήστες ή στους ειδικούς του τομέα να παρέχουν σχόλια σχετικά με τα outliers που έχουν εντοπιστεί. Αυτή η ανατροφοδότηση μπορεί να χρησιμοποιηθεί για να βελτιώσει επαναληπτικά το μοντέλο αναγνώρισης ανωμαλιών και να μειώσει τα false positives ή negatives.

Feature Engineering: Πειραματισμός με την εισαγωγή πρόσθετων χαρακτηριστικών ή μηχανικών χαρακτηριστικών που μπορεί να αποτυπώνουν καλύτερα τα χαρακτηριστικά των outliers.

Anomaly Ranking: Μπορεί να αναπτυχθεί ένας μηχανισμός για την κατάταξη των outliers με βάση το πόσο σημαντικοί είναι ή τον πιθανό αντίκτυπό τους. Αυτό μπορεί να βοηθήσει στην ιεράρχηση των ανταποκρίσεων ή των παρεμβάσεων για τις πιο κρίσιμες ανωμαλίες, ενισχύοντας την πρακτική χρησιμότητα του συστήματος.

Handling Strategies: Μπορεί να δημιουργηθεί ένα σύνολο διαφορετικών στρατηγικών για τον χειρισμό ανωμαλιών, όπως αφαίρεση, μετασχηματισμό ή καταλογισμό. Η επιλογή μιας συγκεκριμένης στρατηγικής μπορεί να βασίζεται στον τύπο και τη σοβαρότητα των outliers που έχουν αναγνωριστεί.

Integration with Decision Support Systems: Μπορεί να γίνει ενοποίηση με συστήματα υποστήριξης αποφάσεων που παρέχουν χρήσιμες πληροφορίες ή συστάσεις βάσει των αναγνωρισμένων ανωμαλιών. Αυτό μπορεί να διευκολύνει την τεκμηριωμένη λήψη αποφάσεων ως απάντηση σε περιστάσεις που υπάρχουν outliers.

Αυτές οι μελλοντικές επεκτάσεις που προτείνονται τόσο για την αναγνώριση όσο και για τον χειρισμό ανωμαλιών στοχεύουν στη βελτίωση της συνολικής αποτελεσματικότητας, προσαρμοστικότητας και ερμηνείας του συστήματος.

ΒΙΒΛΙΟΓΡΑΦΙΑ

• https://el.wikipedia.org/wiki/Μηχανικη_Μαθηση

• https://en.wikipedia.org/wiki/Outlier

• https://en.wikipedia.org/wiki/Anomaly_detection

• https://www.xlstat.com/en/solutions/features/1-class-support-vector-machine

• https://www.analyticsvidhya.com/blog/2022/06/one-class-classification-using-support-vector-machines/

• https://en.wikipedia.org/wiki/Isolation_forest

• https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.IsolationForest.html

• https://scikit-learn.org/stable/modules/generated/sklearn.svm.OneClassSVM.html

• Breunig, M. M., Kriegel, H. P., Ng, R. T., & Sander, J. (2000). LOF: Identifying Density-Based Local Outliers. In Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data.

• Schölkopf, B., Platt, J. C., Shawe-Taylor, J., Smola, A. J., & Williamson, R. C. (2001). Estimating the Support of a High-Dimensional Distribution. Neural Computation, 13(7), 1443–1471.

• Liu, F. T., Ting, K. M., & Zhou, Z. H. (2008). Isolation Forest. In 2008 Eighth IEEE International Conference on Data Mining.

• Zhang, Z., Chen, P., Phillips, P., Wang, J., & Ji, G. (2005). LOF: A Leader-Based Approach for Identifying Density-Based Local Outliers. Pattern Recognition, 40(3), 1021–1031.

• Hodge, V. J., & Austin, J. (2004). A Survey of Outlier Detection Methodologies. Artificial Intelligence Review, 22(2), 85–126.

• Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Vanderplas, J. (2011). Scikit-learn: Machine

Learning in Python. Journal of Machine Learning Research, 12, 2825–2830.

• Aggarwal, C. C. (2013). Outlier Analysis. Springer.

• Tax, D. M., & Duin, R. P. (2004). Support vector data description. Machine Learning, 54(1), 45–66.

• Campello, R. J., Moulavi, D., & Sander, J. (2013). Density-Based Clustering Based on Hierarchical Density Estimates. In Pacific-Asia Conference on Knowledge Discovery and Data Mining.

• Goldstein, M., & Dengel, A. (2012). Histogram-based Outlier Score (HBOS): A Fast Unsupervised Anomaly Detection Algorithm. In KI 2012: Advances in Artificial Intelligence.