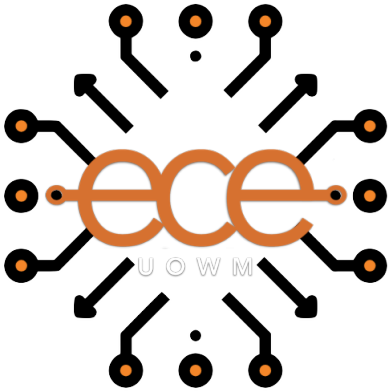


Πανεπιστήμιο Δυτικής Μακεδονίας

Τμήμα Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών

Σύγκριση αλγόριθμων δημιουργίας υποκατάστατων μοντέλων

Νικόδημος Ιακωβάκης (ΑΜ: 1426)

Επιβλέπων Καθηγητής: Νικόλαος Πλόσκας

Εργαστήριο Ευφυών Συστημάτων & Βελτιστοποίησης

18 Οκτωβρίου 2022

Περίληψη

Τα περισσότερα προβλήματα του πραγματικού κόσμου είναι πολύ περίπλοκα και για να γίνει μια μεμονωμένη προσομοίωση μπορεί να διαρκέσει πολλές ώρες ακόμα και μέρες για να ολοκληρωθεί. Αυτό έχει ως αποτέλεσμα να καθιστά τις προσομοιώσεις μη προσιτή τεχνική. Ένας τρόπος για να μετριαστεί το συγκεκριμένο πρόβλημα είναι η χρήση υποκατάστατων μοντέλων τα οποία μιμούνται τη συμπεριφορά του συστήματος όσο το δυνατόν πιο πολύ. Εν αντιθέσει των προσομοιώσεων, τα υποκατάστατα μοντέλα είναι μια υπολογιστικά φθηνή διαδικασία. Στην παρούσα διπλωματική εργασία δόθηκε έμφαση στα προβλήματα μαύρου κουτιού όπου τα υποκατάστατα μοντέλα δεν έχουν πρόσβαση στις πραγματικές παρατηρήσεις και μπορούν να είναι τόσο καλά όσο το μοντέλο του μαύρου κουτιού. Είναι πιθανό να γίνουν κακές προβλέψεις. Για αυτόν τον λόγο πρέπει να έχουμε ένα ξεχωριστό σύνολο δεδομένων δοκιμών, για να μπορούμε να ελέγξουμε την ακρίβεια του μοντέλου. Χρησιμοποιήθηκαν εννέα αλγόριθμοι και δύο λογισμικά (TensorFlow, Alamo) για να ελέγξουμε την ακρίβεια και τα σφάλματα των αλγόριθμων σε διάφορα προβλήματα. Συγκεκριμένα χρησιμοποιήθηκε το SMT το οποίο παρέχει διάφορους αλγόριθμους υποκατάστατων μοντέλων και είναι ανοιχτού κώδικα. Έπειτα προστέθηκαν επιπλέον αλγόριθμοι για να συγκρίνουμε την ακρίβεια τους με τις κλασικές μεθόδους που παρέχει το SMT. Όλοι οι αλγόριθμοι εξετάστηκαν και σε δεδομένα παλινδρόμησης αλλά και σε δεδομένα κατηγοριοποίησης. Τέλος, με την βοήθεια του λογισμικού Alamo πραγματοποιήθηκε η σύγκριση των πραγματικών συναρτήσεων με τις συναρτήσεις όπου προέβλεψε το μοντέλο.

Λέξεις κλειδιά: Python, Μηχανική Μάθηση, Υποκατάστατα Μοντέλα, Προβλήματα Μαύρου Κουτιού

Abstract

Most real-world problems are very complex and a single simulation can take many hours and even days to complete. This has the effect of making simulations an unapproachable technique. One way to mitigate this problem is to use surrogate models that mimic the behavior of the system as closely as possible. Unlike simulations surrogate models are a computationally inexpensive evaluation. In this thesis, the focus was on black box problems where the surrogate models do not have access to the actual observations and can be as good as the black box model. It is possible to make poor predictions. For this reason, we need to have a separate set of test data to be able to check the accuracy of the model. Nine algorithms and two software (TensorFlow, Alamo) were used to test the accuracy and errors of the algorithms on different problems. In particular, SMT was used which provides several surrogate model algorithms and is open source. Additional algorithms were then added to compare their accuracy with the classical methods provided by SMT. All algorithms were tested on both regression and classification data. At last, with the help of Alamo, the comparison of the real functions with the functions that the surrogate model predicted was performed.

Keywords: Python, Machine Learning, Surrogate Models, Black-Box Problems

Δήλωση Πνευματικών Δικαιωμάτων

Δήλωση Πνευματικών Δικαιωμάτων Δηλώνω ρητά ότι, σύμφωνα με το άρθρο 8 του Ν. 1599/1986 και τα άρθρα 2,4,6 παρ. 3 του Ν. 1256/1982, η παρούσα Διπλωματική Εργασία με τίτλο ''Σύγκριση αλγόριθμων δημιουργίας υποκατάστατων μοντέλων'' καθώς και τα ηλεκτρονικά αρχεία και πηγαίοι κώδικες που αναπτύχθηκαν ή τροποποιήθηκαν στα πλαίσια αυτής της εργασίας και αναφέρονται ρητώς μέσα στο κείμενο που συνοδεύουν, και η οποία έχει εκπονηθεί στο Τμήμα Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών του Πανεπιστημίου Δυτικής Μακεδονίας, υπό την επίβλεψη του μέλους του Τμήματος κ. Νικολάου Πλόσκα αποτελεί αποκλειστικά προϊόν προσωπικής εργασίας και δεν προσβάλλει κάθε μορφής πνευματικά δικαιώματα τρίτων και δεν είναι προϊόν μερικής ή ολικής αντιγραφής, οι πηγές δε που χρησιμοποιήθηκαν περιορίζονται στις βιβλιογραφικές αναφορές και μόνον. Τα σημεία όπου έχω χρησιμοποιήσει ιδέες, κείμενο, αρχεία ή / και πηγές άλλων συγγραφέων, αναφέρονται ευδιάκριτα στο κείμενο με την κατάλληλη παραπομπή και η σχετική αναφορά περιλαμβάνεται στο τμήμα των βιβλιογραφικών αναφορών με πλήρη περιγραφή.

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

Υπογραφή Φοιτητή

Περιεχόμενα

Κατάλογος σχημάτων

Κατάλογος αλγορίθμων

Κατάλογος πινάκων

Κεφάλαιο 1

Εισαγωγή

1.1 Ορισμός του προβλήματος

Οι προσομοιώσεις δεν είναι γενικά φθηνές: στη βιομηχανία, μια μεμονωμένη προσομοίωση συνήθως διαρκεί μέρες για να ολοκληρωθεί. Κατά συνέπεια, οι αναλύσεις που απαιτούν πολλές εκτελέσεις προσομοίωσης θα προκαλούσαν απαγορευτικό υπολογιστικό κόστος, καθιστώντας τις έτσι ανέφικτες στην πράξη. Τα υποκατάστατα μοντέλα προσφέρουν λύση στο συγκεκριμένο πρόβλημα κατασκευάζοντας ένα στατιστικό μοντέλο (ή υποκατάστατο μοντέλο) για να προσεγγίσει με ακρίβεια το αποτέλεσμα της προσομοίωσης. Στη συνέχεια, αυτό το εκπαιδευμένο υποκατάστατο μοντέλο μπορεί να αναπτυχθεί για να αντικαταστήσει την αρχική προσομοίωση εκτελώντας ανάλυση ευαισθησίας, βελτιστοποίησης ή ανάλυση κινδύνων.

Δεδομένου ότι μια μεμονωμένη αξιολόγηση του εκπαιδευμένου υποκατάστατου μοντέλου είναι γενικά πολύ πιο γρήγορη από μια μεμονωμένη αξιολόγηση της αρχικής προσομοίωσης, η εκτέλεση εκατοντάδων και χιλιάδων αξιολογήσεων εξόδου με διάφορους συνδυασμούς παραμέτρων σχεδιασμού δεν αποτελεί πλέον πρόβλημα. Πιο απλά, οι τεχνικές υποκατάστατων μοντέλων καθιστούν αυτές τις πολύ ακριβές αναλύσεις προσιτές.

Ο στόχος των υποκατάστατων μοντέλων είναι η εύρεση μιας προσεγγιστικής συνάρτησης που μιμείται τη συμπεριφορά του αρχικού συστήματος, αλλά μπορεί να αξιολογηθεί πολύ πιο γρήγορα. Αυτή η λειτουργία κατασκευάζεται με την εκτέλεση πολλαπλών προσομοιώσεων (που ονομάζονται δείγματα) σε βασικά σημεία του σχεδιαστικού χώρου, αναλύοντας τα αποτελέσματα και επιλέγοντας ένα μοντέλο που προσεγγίζει τα δείγματα και τη συνολική συμπεριφορά του συστήματος

αρκετά καλά.

Υπάρχει μεγάλη ποικιλία διαθέσιμων μοντέλων και ποιο είναι το καταλληλότερο εξαρτάται σε μεγάλο βαθμό από το σύστημα που πρόκειται να μοντελοποιηθεί. Δημοφιλείς επιλογές είναι οι πολυωνυμικές συναρτήσεις (QP), μοντέλα Kriging, Τεχνητά Νευρωνικά Δίκτυα, μοντέλα Συναρτήσεων Ακτινικής Βάσης (RBF).

1.2 Κίνητρα και Στόχοι Υλοποίησης

Κίνητρο για την παρούσα διπλωματική εργασία αποτέλεσε η πρόοδος της αριθμητικής προσομοίωσης η οποία έχει βοηθήσει σημαντικά τη μελέτη πολλών πολύπλοκων φυσικών φαινομένων και γίνεται ολοένα και πιο διαδεδομένη. Τέτοια υπολογιστικά μοντέλα είναι συχνά πολύπλοκα, περιλαμβάνουν πολλούς κλάδους, πολλές παραμέτρους εισόδου και μεγάλους χρόνους υπολογισμού. Λόγω της μεγάλης πολυπλοκότητας οι προσομοιώσεις χρειάζονται συνήθως μέρες για να ολοκληρωθούν και για αυτό το λόγο είναι απρόσιτες.

Υποκατάστατα μοντέλα χαμηλότερης πολυπλοκότητας που είναι φθηνά στην αξιολόγηση και την προσεγγίζουν με ακρίβεια το μοντέλο μπορούν να διευκολύνουν σημαντικά τις εργασίες ανάλυσης. Τα υποκατάστατα μοντέλα έχουν αποδειχθεί ότι είναι αποτελεσματικές προσεγγίσεις για υπολογιστικά ακριβά μοντέλα, όπως η αεριοπροώθηση και ο σχεδιασμός αεροτομών. Παρά αυτές τις αποδεδειγμένες επιτυχίες, μια βασική πρόκληση που απομένει είναι η εξαγωγή υποκατάστατων μοντέλων για συστήματα μαύρου κουτιού μεγάλης κλίμακας με πολλές παραμέτρους εισόδου.

1.3 Διάρθρωση κειμένου

Η παρούσα διπλωματική εργασία χωρίζεται σε τέσσερα κεφάλαια. Στο δεύτερο κεφάλαιο γίνεται εισαγωγή στη μηχανική μάθηση και στο θεωρητικό υπόβαθρο που χρειάζεται για τα επόμενα κεφάλαια. Στην αρχή του κεφαλαίου γίνεται εισαγωγή στις έννοιες της μηχανικής μάθησης, στα είδη και στις κατηγορίες που χωρίζεται.

Στο τρίτο κεφάλαιο αναλύονται οι αλγόριθμοι που χρησιμοποιήθηκαν στην παρούσα εργασία. Επίσης, γίνεται και η μαθηματική περιγραφή κάθε αλγορίθμου. Στο τέλος αναφέρονται τα λογισμικά τα οποία χρησιμοποιήθηκαν (Alamo, TensorFlow)

και η χρησιμότητά τους.

Στο τέταρτο κεφάλαιο παρουσιάζονται τα αποτελέσματα των αλγορίθμων που αναλύθηκαν στο τρίτο κεφάλαιο, τόσο σε προβλήματα παλινδρόμησης αλλά και σε προβλήματα κατηγοριοποίησης. Στο τέλος γίνεται σύγκριση της συνάρτησης που προέβλεψε το λογισμικό Alamo με την πραγματική και σχολιάζεται κατά πόσο έπεσε κοντά στην πραγματική συνάρτηση.

Στο πέμπτο και τελευταίο κεφάλαιο της παρούσας διπλωματικής εργασίας διαπιστώνονται τα συμπεράσματα από τη σύγκριση των αλγορίθμων.

Κεφάλαιο 2

Μηχανική Μάθηση

2.1 Εισαγωγή

Μηχανική μάθηση (machine learning) είναι ένα πεδίο της επιστήμης των υπολογιστών που αναπτύχθηκε από τη μελέτη της αναγνώρισης προτύπων και της υπολογιστικής θεωρίας μάθησης στην τεχνητή νοημοσύνη. Ο όρος μηχανική μάθηση επινοήθηκε για πρώτη φορά από τον Arthur Samuel το 1959. Το λογισμικό για το παιχνίδι της Ντάμας που ανέπτυξε ήταν ανάμεσα στα πρώτα επιτυχημένα προγράμματα υπολογιστή που μπορούσαν να μαθαίνουν από μόνα τους. Ο ίδιος όρισε τη μηχανική μάθηση ως: "Πεδίο μελέτης που δίνει στους υπολογιστές την ικανότητα να μαθαίνουν, χωρίς να έχουν ρητά προγραμματιστεί για αυτό το σκοπό" [1].

Τροφοδοτούμενη από την πρόοδο της στατιστικής και την επιστήμη των υπολογιστών, καθώς και από τα καλύτερα σύνολα δεδομένων και την ανάπτυξη των νευρωνικών δικτύων, η μηχανική μάθηση έχει πραγματικά απογειωθεί τα τελευταία χρόνια. Λόγω της ταχύτατης ανάπτυξης, η μηχανική μάθηση είναι παντού - αυτοματοποιημένη μετάφραση, αναγνώριση εικόνας, φωνητική αναζήτηση, αυτοκίνητα που οδηγούνε μόνα τους και πολλά άλλα ακόμα.

Η μηχανική μάθηση είναι μια αυτοματοποιημένη διαδικασία που επιτρέπει στις μηχανές να επιλύουν προβλήματα με ελάχιστη ή καθόλου ανθρώπινη συνεισφορά και να αναλαμβάνουν ενέργειες βασισμένες σε παρατηρήσεις του παρελθόντος. Αντί να προγραμματίζονται αλγόριθμοι μηχανικής μάθησης για την εκτέλεση εργασιών, τροφοδοτούνται με δεδομένα εκπαίδευσης, τα οποία τους βοηθούν να κάνουν υπολογισμούς, να επεξεργάζονται δεδομένα και να αναγνωρίζουν μοτίβα από μόνοι τους. Η μηχανική μάθηση μπορεί να λειτουργήσει σε τεράστιο όγκο δεδομένων και

αποδίδει με μεγαλύτερη ακρίβεια από τους ανθρώπους.

Ο Tom M. Mitchell πρότεινε έναν επίσημο ορισμό για τη μηχανική μάθηση 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**” [2].

2.2 Είδη Μηχανικής Μάθησης

Σε έναν τόσο ευρύ και ταχύτατα αναπτυσσόμενο τομέα σαν τη μηχανική μάθηση λογικό είναι να έχουμε διαφορετικές τεχνικές οι οποίες αναπτύσσονται συνεχώς. Τα τρία βασικότερα είδη στα οποία χωρίζεται η μηχανική μάθηση είναι: επιβλεπόμενη μάθηση, μη επιβλεπόμενη μάθησης και της ενισχυτική μάθησης.

2.2.1 Επιβλεπόμενη Μάθηση (Supervised Learning)

Ο αλγόριθμος διαβάζει τα δεδομένα εκπαίδευσης και τις αντίστοιχες γνωστές τιμές εξόδου. Ο στόχος είναι να μάθουμε γενικούς κανόνες (επίσης αποκαλούμενους συχνά ”μοντέλο”) που αντιστοιχίζουν τις εισόδους σε εξόδους, έτσι ώστε να είναι δυνατή η πρόβλεψη της εξόδου για νέα αόρατα δεδομένα, όπου έχουμε παρατηρήσει τιμές εισόδου αλλά όχι τη σχετική έξοδο.

Παλινδρόμηση (Regression)

Είναι μια τεχνική για τη διερεύνηση της σχέσης μεταξύ ανεξάρτητων μεταβλητών ή χαρακτηριστικών και μιας εξαρτημένης μεταβλητής ή αποτελέσματος. Η τιμή εξόδου αναζητείται μέσα από ένα συνεχές σύνολο τιμών (για παράδειγμα το σύνολο των πραγματικών αριθμών).

- Ένα πρόβλημα παλινδρόμησης απαιτεί την πρόβλεψη μιας ποσότητας.
- Μπορεί να έχει πραγματική αξία ή διακριτές μεταβλητές εισόδου.
- Ένα πρόβλημα με πολλαπλές μεταβλητές εισόδου ονομάζεται συχνά πρόβλημα παλινδρόμησης πολλαπλών μεταβλητών (multivariate).

- Ένα πρόβλημα παλινδρόμησης όπου οι μεταβλητές εισόδου ταξινομούνται με βάση το χρόνο ονομάζεται πρόβλημα πρόβλεψης χρονοσειρών.

Επειδή ένα μοντέλο πρόβλεψης παλινδρόμησης προβλέπει μια ποσότητα, η ικανότητα του μοντέλου πρέπει να αναφέρεται ως σφάλμα σε αυτές τις προβλέψεις. Υπάρχουν πολλοί τρόποι εκτίμησης της ικανότητας ενός μοντέλου πρόβλεψης παλινδρόμησης, αλλά ίσως ο πιο συνηθισμένος είναι ο υπολογισμός του μέσου τετραγώνου σφάλματος (mean squared error).

Ταξινόμηση (Classification)

Τα δεδομένα χωρίζονται σε δύο ή περισσότερες κλάσεις και ο αλγόριθμος δημιουργεί ένα μοντέλο πρόβλεψης διακριτών τιμών. Το οποίο θα αντιστοιχίζει τα δεδομένα σε μια ή περισσότερες κλάσεις. Συνήθως χρησιμοποιείται επιβλεπόμενη μάθηση αλλά χρησιμοποιείται και η μη επιβλεπόμενη μάθηση [3].

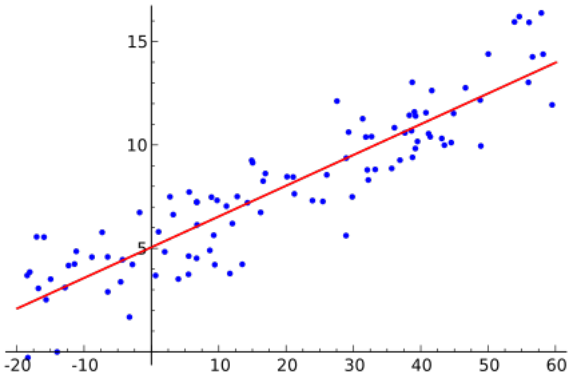
- Ένα πρόβλημα ταξινόμησης απαιτεί τα παραδείγματα να ταξινομηθούν σε μία από δύο ή περισσότερες κατηγορίες.
- Μια ταξινόμηση μπορεί να έχει μεταβλητές εισόδου πραγματικής αξίας ή διακριτές.
- Ένα πρόβλημα με δύο κλάσεις ονομάζεται συχνά πρόβλημα ταξινόμησης δύο κλάσεων ή δυαδικής ταξινόμησης.
- Ένα πρόβλημα με περισσότερες από δύο κλάσεις ονομάζεται συχνά πρόβλημα ταξινόμησης πολλαπλών κλάσεων.
- Ένα πρόβλημα όπου ένα παράδειγμα έχει εκχωρηθεί σε πολλές κλάσεις ονομάζεται πρόβλημα ταξινόμησης πολλαπλών ετικετών (multi-label).

Υπάρχουν πολλοί τρόποι για να εκτιμηθεί η ικανότητα ενός μοντέλου πρόβλεψης ταξινόμησης, αλλά ίσως ο πιο συνηθισμένος είναι ο υπολογισμός της ακρίβειας (accuracy).

Γραμμικη Παλινδρόμηση (Linear Regression)

Μέσω της γραμμικής παλινδρόμησης αναγνωρίζεται η σχέση μεταξύ μιας εξαρτώμενης μεταβλητής και μίας ή περισσοτέρων ανεξάρτητων με σκοπό να γίνονται

προβλέψεις για μελλοντικά αποτελέσματα. Όταν έχουμε μία ανεξάρτητη μεταβλητή τότε ονομάζεται απλή γραμμική παλινδρόμηση, ενώ αν αυξηθούν οι ανεξάρτητες μεταβλητές το ονομάζουμε πολλαπλή γραμμική παλινδρόμηση. Το τελικό αποτέλεσμα (ανεξαρτήτου τι τύπου γραμμικής παλινδρόμησης θα έχουμε) θα είναι μια ευθεία γραμμή με όσο το δυνατόν μικρότερο σφάλμα.



Σχήμα 2.1: Παράδειγμα γραμμικής παλινδρόμησης .

Στο Σχήμα 2.1 φαίνεται ένα απλό παράδειγμα γραμμικής παλινδρόμησης, η κόκκινη ευθεία είναι η εξίσωση $y = a + bx$, που μοντελοποιεί τα σημεία.

Λογιστική Παλινδρόμηση (Logistic Regression)

Χρησιμοποιεί την ίδια τεχνική με τη γραμμική παλινδρόμηση αλλά έχουν διαφορετικό τύπο δεδομένων.Στην περίπτωση της λογιστικής παλινδρόμησης τα δεδομένα είναι κατηγορηματικά δηλαδή διακριτά και πολλές φορές δυαδικά (αληθή ή ψευδή).

Αλγόριθμος Κ-κοντινότερων γειτονικών κόμβων(K-nearest Neighbor (KNN) Algorithm)

Είναι ένας μη παραμετρικός αλγόριθμος, ο οποίος χρησιμοποιεί την εγγύτητα για να κάνει ταξινομήσεις ή προβλέψεις σχετικά με την ομαδοποίηση ενός μεμονωμένου σημείου δεδομένων. Ενώ μπορεί να χρησιμοποιηθεί είτε για προβλήματα παλινδρόμησης είτε για προβλήματα ταξινόμησης, συνήθως χρησιμοποιείται ως αλγόριθμος ταξινόμησης, με βάση την υπόθεση ότι παρόμοια σημεία μπορούν να βρεθούν το ένα κοντά στο άλλο. Υπολογίζει την Ευκλείδεια απόστασης (μπορεί να χρησιμοποιηθεί και άλλο μέτρο για τον υπολογισμό της απόστασης, όπως για παράδειγμα η απόσταση Manhattan) μεταξύ των δεδομένων και στη συνέχεια αποδίδει μια κατηγορία

βασισμένος στην πιο συχνή κατηγορία ή στο μέσο.

Naive Bayes Ταξινόμηση (Naive Bayes Classifier)

Χρησιμοποιούνται για τη μοντελοποίηση μιας πιθανοτικής σχέσης μεταξύ των χαρακτηριστικών και της κατηγορίας στόχος τους είναι δηλαδή η πρόβλεψη της πιθανότητας ένα δείγμα «A» να ανήκει σε κάποια κατηγορία «B» [4]. Η αρχή στην οποία βασίζεται είναι το θεώρημα Bayes, παρακάτω στην εξίσωση 2.1 φαίνεται ο μαθηματικός τύπος του θεωρήματος.

$$P(A|B) = P(A)\frac{P(B|A)}{P(B)} \tag{2.1}$$

- $P(A|B)$: η πιθανότητα να πραγματοποιηθεί το με δεδομένο ότι συμβαίνει το B.
- $P(B|A)$: η πιθανότητα του δεδομένου με δεδομένο ότι η υπόθεση ήταν αληθής.
- $P(A)$: η πιθανότητα η υπόθεση να είναι αληθής.
- $P()$: η πιθανότητα του δεδομένου ανεξάρτητα από την υπόθεση .

Bayesian Δίκτυα

Είναι ένα γραφικό μοντέλο για την αναπαράσταση σχέσεων πιθανότητας μεταξύ συνόλου μεταβλητών. Αρχικά, πρέπει να απεικονιστεί το δίκτυο και στη συνέχεια καθορίζονται παράμετροι που καθιστούν δύσκολη την εφαρμογή του χωρίς τη γνώμη ειδικού. Οι προηγούμενες πληροφορίες σχετικά με το πρόβλημα μπορούν να αναπαρασταθούν ως δομική σχέση μεταξύ των χαρακτηριστικών του. Ωστόσο, το δίκτυο Bayes δεν είναι πολύ επιτυχημένο με σύνολα δεδομένων υψηλών διαστάσεων επειδή τα μεγάλα δίκτυα δεν είναι εφικτά από άποψη χρόνου και χώρου.

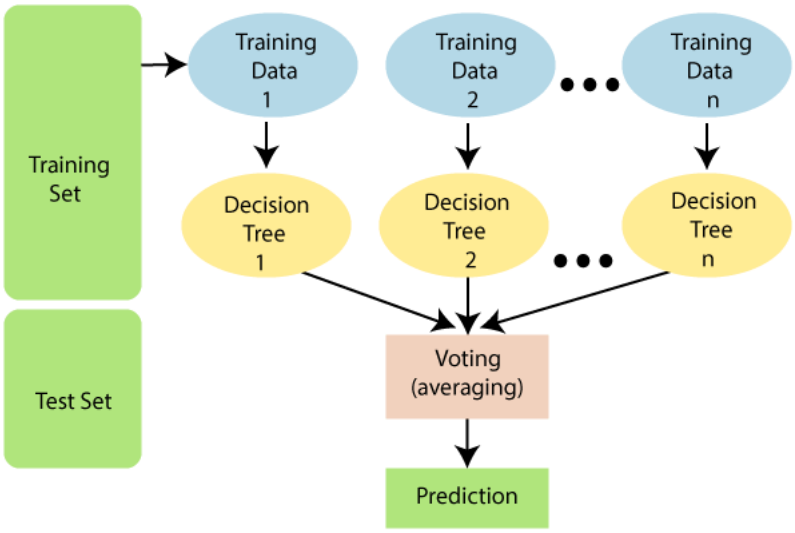
Δέντρα Απόφασης (Decision Trees)

Τα δέντρα απόφασης χειρίζονται εύκολα τις αλληλεπιδράσεις μεταξύ των χαρακτηριστικών. Δεδομένου ότι είναι μη παραμετρικό, οι ακραίες τιμές δεν επηρεάζουν πολύ το μοντέλο και μπορεί να αντιμετωπίσει γραμμικά αδιαχώριστα δεδομένα. Μπορούν να χειριστούν ποικιλία δεδομένων (ονομαστικά, αριθμητικά, κείμενο), τιμές που λείπουν και περιττές ιδιότητες. Έχουν καλή ικανότητα γενίκευσης και είναι

ανθεκτικά στο θόρυβο, παρέχουν υψηλή απόδοση για σχετικά μικρή υπολογιστική προσπάθεια.

Τυχαία Δάση (Random Forests)

Τα τυχαία δάση είναι μια μέθοδος συνόλου που λειτουργεί εκπαιδεύοντας έναν αριθμό δέντρων απόφασης και επιστρέφοντας την τάξη με την πλειοψηφία σε όλα τα δέντρα του συνόλου. Είναι γρήγορα, επεκτάσιμα, ανθεκτικά στο θόρυβο, ερμηνεύονται εύκολα και οπτικοποιούνται χωρίς παραμέτρους για διαχείριση. Ωστόσο, καθώς ο αριθμός των δέντρων αυξάνεται, ο αλγόριθμος γίνεται αργός για την πρόβλεψη σε πραγματικό χρόνο.



Σχήμα 2.2: Σχηματική απεικόνιση του Random Forest

Στο Σχήμα 2.2 βλέπουμε τη λειτουργία του αλγορίθμου Random Forest.

Μηχανή διανυσματικής υποστήριξης (Support Vector Machine)

Είναι ένας πολύπλοκος αλγόριθμος, αλλά μπορεί να παρέχει υψηλή ακρίβεια. Αποτρέπει επίσης θεωρητικές εγγυήσεις σχετικά με την υπερβολική τοποθέτηση και με έναν κατάλληλο πυρήνα (kernel), μπορούν να λειτουργήσουν καλά ακόμα κι αν τα δεδομένα σας δεν διαχωρίζονται γραμμικά στον βασικό χώρο χαρακτηριστικών. Βασίζονται στην έννοια της μεγιστοποίησης της ελάχιστης απόστασης από το υπερεπίπεδο στο πλησιέστερο σημείο δείγματος. Μπορεί να χρησιμοποιηθεί για ταξινόμηση άλλα και για παλινδρόμηση [5].

2.2.2 Μη επιβλεπόμενη Μάθηση(Unsupervised Learning)

Ο αλγόριθμος δημιουργεί ένα μοντέλο με βάση ένα σύνολο εισόδων σε μορφή παρατηρήσεων χωρίς να γνωρίζει τις επιθυμητές τιμές εξόδου. Συνήθως χρησιμοποιείται σε προβλήματα:

- Ομαδοποίησης (Clustering)
- Κανόνες συσχέτισης (Association Rules)

Ομαδοποίηση

Είναι η διαίρεση των σημείων δεδομένων σε έναν αριθμό ομάδων έτσι ώστε τα σημεία δεδομένων στις ίδιες ομάδες να έχουν μεγαλύτερη ομοιότητα με άλλα σημεία δεδομένων στην ίδια ομάδα και ανόμοια με τα σημεία δεδομένων σε άλλες ομάδες. Είναι βασικά μια συλλογή αντικειμένων με βάση την ομοιότητα και την ανομοιότητα μεταξύ τους.

Οι Αλγόριθμοι ομαδοποίησης μπορούν να χωριστούν σε κατηγορίες:

- Αποκλειστικοί (Exclusive)
- Επικαλυπτόμενοι (Overlapping)
- Ιεραρχίας (hierarchy)
- Πιθανοτήτων (Probabilistic)
- Ασαφής (Fuzzy)

Κανόνες Συσχέτισης

Εμφανίστηκε αρκετά αργότερα από τη μηχανική μάθηση και έχει περισσότερες επιρροές από την ερευνητική περιοχή των βάσεων δεδομένων. Αποσκοπεί στον εντοπισμό ισχυρών κανόνων που ανακαλύφθηκαν σε βάσεις δεδομένων χρησιμοποιώντας ορισμένα μέτρα ενδιαφέροντος. Σε κάθε δεδομένη συναλλαγή με μια ποικιλία στοιχείων, οι κανόνες συσχέτισης προορίζονται να ανακαλύψουν τους κανόνες που καθορίζουν πώς ή γιατί συνδέονται ορισμένα στοιχεία. Η μάθηση με κανόνες συσχέτισης μπορεί να χωριστεί σε τρεις τύπους αλγορίθμων:

- Apriori
- Eclat
- F-P Growth Algorithm (Αλγόριθμος ανάπτυξης)

2.2.3 Ενισχυτική Μάθηση (Reinforcement Learning)

Ο αλγόριθμος μαθαίνει μέσω της άμεσης αλληλεπίδρασης με το περιβάλλον. Χρησιμοποιεί μόνο τα πρότυπα εισόδου χωρίς να γνωρίζει αν πάει καλά ή όχι, παρά μόνο στο τέλος. Τότε μόνο γνωρίζει αν πέτυχε το στόχο του και χρησιμοποιεί αυτή την πληροφορία για να βελτιωθεί την επόμενη φορά. Χρησιμοποιείται κυρίως σε προβλήματα Σχεδιασμού (Planning), όπως για παράδειγμα ο έλεγχος κίνησης ρομπότ. Το διασημότερο παράδειγμα είναι έξυπνες μηχανές που παίζουν σκάκι (συνήθως χρησιμοποιείται τεχνητό νευρωνικό δίκτυο). Τα κύρια στοιχεία ενός συστήματος ενισχυτικής μάθησης είναι:

- Ο πράκτορας.
- Το περιβάλλον με το οποίο αλληλεπιδρά ο πράκτορας.
- Η πολιτική που ακολουθεί ο πράκτορας για να προβεί σε ενέργειες.
- Η ανταμοιβή που παρατηρεί ο πράκτορας κατά τη λήψη αποφάσεων.

2.2.4 Ημι-Επιβλεπόμενη Μάθηση (Semi Supervised Machine Learning)

Είναι μια ευρεία κατηγορία τεχνικών μηχανικής εκμάθησης που χρησιμοποιεί δεδομένα τόσο με ετικέτα όσο και χωρίς ετικέτα. Με αυτόν τον τρόπο, όπως υποδηλώνει το όνομα, είναι μια υβριδική τεχνική μεταξύ επιβλεπόμενης και μη επιβλεπόμενης μάθησης. Γενικά, η βασική ιδέα της ημι-επιβλεπόμενης μάθησης είναι η διαφορετική αντιμετώπιση ενός σημείου δεδομένων με βάση το αν έχει ετικέτα ή όχι: για σημεία με ετικέτα, ο αλγόριθμος θα χρησιμοποιήσει την παραδοσιακή επίβλεψη για να ενημερώσει τα βάρη του μοντέλου και για σημεία χωρίς ετικέτα, ο αλγόριθμος ελαχιστοποιεί τη διαφορά στις προβλέψεις μεταξύ άλλων παρόμοιων παραδειγμάτων εκπαίδευσης. Η ημι-επιβλεπόμενη μάθηση είναι χρήσιμη όταν το κόστος που σχετίζεται με την επισήμανση είναι πολύ υψηλό για να δώσει τη δυνατότητα για

μια πλήρως επισημασμένη διαδικασία κατάρτισης. Τα πρώτα παραδείγματα αυτού περιλαμβάνουν την αναγνώριση του προσώπου ενός ατόμου σε μια κάμερα ιστού.

2.2.5 Νευρωνικά Δίκτυα(Neural Networks)

Τα Τεχνητά Νευρωνικά Δίκτυα χαρακτηρίζονται από την προσπάθειά τους να προσομοιώσουν τη λειτουργία των βιολογικών Νευρωνικών Δικτύων με βάση κάποιο μαθηματικό μοντέλο. Πρόκειται για εξελιγμένες τεχνικές μη γραμμικής μοντελοποίησης, ικανές να μοντελοποιήσουν πολύπλοκες λειτουργίες. Τις περισσότερες φορές που η γραμμική προσέγγιση αποτυγχάνει, τα Νευρωνικά Δίκτυα δίνουν λύσεις, επιτρέποντας τη μη γραμμικότητα μέσω των μη γραμμικών συναρτήσεων ενεργοποίησης [6]. Γενικά θεωρούμε πως υπάρχουν τριών ειδών νευρώνες:

- Οι νευρώνες εισόδου, των οποίων η εργασία είναι να διοχετεύσουν στους υπολογιστικούς νευρώνες την είσοδο του προβλήματος
- Οι νευρώνες εξόδου, χρησιμοποιούνται για να παρουσιάσουν στο περιβάλλον την απάντηση του τεχνητού νευρωνικού δικτύου σε κάποιο πρόβλημα, όπως για παράδειγμα την εκτίμηση της κατηγορίας ενός προβλήματος κατηγοριοποίησης.
- Οι υπολογιστικοί νευρώνες ή κρυμμένοι νευρώνες, οι οποίοι πολλαπλασιάζουν κάθε είσοδό που δέχονται από νευρώνες εισόδου ή από άλλους νευρώνες επεξεργασίας με μια τιμή συσχετισμένη με αυτούς που ονομάζεται βάρος. Το συνολικό αποτέλεσμα εισάγεται σε μια συνάρτηση που θα ονομάζουμε συνάρτηση ενεργοποίησης και είτε παρουσιάζεται στην έξοδο είτε δίδεται σε κάποιον άλλο νευρώνα επεξεργασίας.

Εκπαίδευση των Τεχνιτών Νευρωνικών Δικτύων

- **Εκπαίδευση με επίβλεψη**, όπου το νευρωνικό δίκτυο μαθαίνει να απεικονίζει δεδομένες εισόδους σε εξόδους εκ των προτέρων γνωστές, με απώτερο στόχο τη γενίκευση της αναγνώρισης αυτής και για παρεμφερείς εισόδους στο μέλλον.
- **Εκπαίδευση χωρίς επίβλεψη**, όπου το νευρωνικό δίκτυο κατασκευάζει απεικονίσεις από μια αναπαράσταση σε μια άλλη.

- **Ενισχυτική Εκπαίδευση**, όπου ο αλγόριθμος μαθαίνει μια στρατηγική ενεργειών για δεδομένου τύπου παρατηρήσεις.

2.3 Διασταυρωμένη Επικύρωση (Cross Validation)

Η εκπαίδευση ενός αλγορίθμου μάθησης γίνεται με την χρήση των διαθέσιμων δεδομένων. Το πρόβλημα που παρουσιάζεται είναι κατά πόσο είναι αξιόπιστα τα αποτελέσματα που λαμβάνονται κατά τη διαδικασία της γενίκευσης, όταν εισάγονται νέα δεδομένα στο σύστημα με άγνωστη τιμή εξόδου. Πρέπει να ελεγχθεί η ικανότητα του αλγορίθμου στη γενίκευση, να αποδίδει σωστά και σε άγνωστα δεδομένα. Ο τρόπος ελέγχου αναγκαστικά θα πρέπει να βασιστεί στα γνωστά διαθέσιμα δεδομένα, τα οποία θα προσεγγιστούν όχι ως στοιχεία εκπαίδευσης αλλά ελέγχου. Για το λόγο αυτό το σύνολο δεδομένων διαιρείται σε δύο τμήματα, εκπαίδευσης (training data) και ελέγχου (test data). Αφού έχει εκπαιδευτεί το μοντέλο στα δεδομένα εκπαίδευσης αξιολογείται η απόδοση του στα δεδομένα ελέγχου. Εισάγονται τα δεδομένα ελέγχου στο μοντέλο και η έξοδος του ελέγχεται με τις γνωστές τιμές εξόδου των δεδομένων, έτσι προκύπτει ένα ποσοστό επιτυχίας της εκπαίδευσης [7].

2.4 Γιατί τα δεδομένα είναι σημαντικά στη μηχανική μάθηση

Τα δεδομένα είναι η πιο σημαντική και απαραίτητη τροφή για τη μηχανική μάθηση. Η μηχανική μάθηση χωρίς δεδομένα δεν είναι παρά μια γυμνή μηχανή χωρίς ψυχή και μυαλό. Αυτά τα δεδομένα κάνουν τις μηχανές να κάνουν τέτοιες εκπληκτικές εργασίες, τις οποίες δεν είχαμε φανταστεί πριν λίγα χρόνια.

Παρόλο που τα δεδομένα έχουν τεράστια σημασία, οι μηχανές δεν καταλαβαίνουν τι αντιπροσωπεύουν τα δεδομένα. Δεν καταλαβαίνουν γιατί το «cat» είναι «cat» και γιατί γράφεται με αυτόν τον τρόπο [8].

2.4.1 Δεδομένα

Μπορεί να είναι οποιοδήποτε γεγονός, κείμενο, σύμβολα, εικόνες, βίντεο κ.λ.π., αλλά σε μη επεξεργασμένη μορφή. Όταν τα δεδομένα υποβάλλονται σε επεξεργασία, είναι γνωστά ως πληροφορίες.

- **Αριθμητικά Δεδομένα** (Numerical Data) αναφέρονται σε δεδομένα που έχουν τη μορφή αριθμών και όχι σε οποιαδήποτε γλώσσα ή περιγραφική μορφή.
- **Κατηγορηματικά Δεδομένα** (Categorical Data) αναφέρονται σε έναν τύπο δεδομένων που μπορεί να αποθηκευτεί και να αναγνωριστεί με βάση τα ονόματα ή τις ετικέτες που τους δίνονται(π.χ. φύλο, μάρκα αυτοκινήτου, κ.λπ.).

2.4.2 Ποιότητα των Δεδομένων

Στη μηχανική μάθηση ισχύει ότι: μη ποιοτικά δεδομένα έχουν ως αποτέλεσμα μη ποιοτικές προβλέψεις. Η ποιότητα των δεδομένων βασίζεται σε αρκετούς παραμέτρους όπως:

- Ακρίβεια
- Ποσότητα, αν έχουμε λίγα δεδομένα δεν μπορούμε να κάνουμε ακριβής προβλέψεις, πάντα πρέπει να έχουμε έναν αριθμό αρκετό ώστε ο αλγόριθμος να μπορεί να κάνει μια ακριβή πρόβλεψη.
- Αξιοπιστία

2.4.3 Οι διασημότεροι ιστότοποι για δεδομένα

Οι 5 διασημότερες ιστοσελίδες για δεδομένα είναι:

- Google's Dataset Search
- Microsoft Research Open Data
- Amazon Datasets
- UCI Machine Learning Repository
- Goverments Datasets

Τα δεδομένα που χρησιμοποιήθηκαν είναι από το:
UCI Machine Learning Repository [9], minplib [10].

2.5 Σύνοψη Κεφαλαίου

Στο κεφάλαιο αυτό έγινε εισαγωγή στις βασικές έννοιες της μηχανικής μάθησης καθώς και σε κάποιους βασικούς αλγόριθμους. Είδαμε σε ποιες κατηγορίες χωρίζεται η μηχανική μάθηση καθώς και γιατί τα δεδομένα είναι ένα αναπόσπαστο κομμάτι της μηχανικής μάθησης.

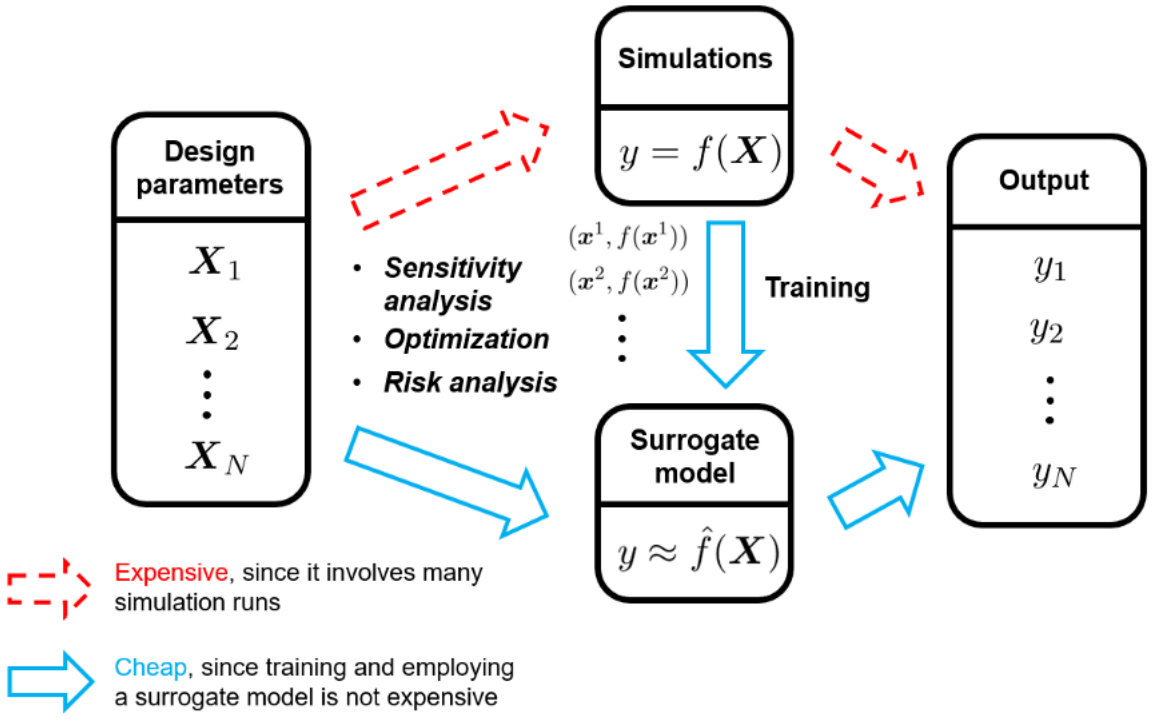
Κεφάλαιο 3

Υποκατάστατα Μοντέλα

Ένα υποκατάστατο μοντέλο (surrogate model) χρησιμοποιείται όταν ένα αποτέλεσμα ενδιαφέροντος δεν μπορεί εύκολα να μετρηθεί ή να υπολογιστεί, επομένως χρησιμοποιείται ένα μοντέλο του αποτελέσματος. Τα περισσότερα προβλήματα μηχανικού σχεδιασμού απαιτούν πειράματα και προσομοιώσεις για την αξιολόγηση των συναρτήσεων στόχου και περιορισμού σχεδιασμού ως συνάρτηση των μεταβλητών σχεδιασμού. Για παράδειγμα, προκειμένου να βρεθεί το βέλτιστο σχήμα αεροτομής για ένα φτερό αεροσκάφους, ένας μηχανικός προσομοιώνει τη ροή αέρα γύρω από το φτερό για διαφορετικές μεταβλητές σχήματος (μήκος, καμπυλότητα, υλικό κλπ). Για πολλά προβλήματα του πραγματικού κόσμου, ωστόσο, μια μεμονωμένη προσομοίωση μπορεί να διαρκέσει πολλά λεπτά, ώρες ή ακόμα και ημέρες για να ολοκληρωθεί. Ως αποτέλεσμα, εργασίες ρουτίνας όπως η βελτιστοποίηση σχεδιασμού, η εξερεύνηση του χώρου σχεδιασμού, η ανάλυση ευαισθησίας και η ανάλυση what-if καθίστανται αδύνατες, καθώς απαιτούν χιλιάδες ή και εκατομμύρια αξιολογήσεις προσομοίωσης. Ένας τρόπος μετριασμού αυτού του προβλήματος είναι η κατασκευή μοντέλων προσέγγισης, γνωστών ως υποκατάστατα μοντέλα (surrogate models), μεταμοντέλα (metamodels) ή εξομοιωτές (emulators), που μιμούνται τη συμπεριφορά του μοντέλου προσομοίωσης όσο το δυνατόν πλησιέστερα, ενώ είναι υπολογιστικά φθηνή αξιολόγηση. Τα υποκατάστατα μοντέλα κατασκευάζονται χρησιμοποιώντας μια προσέγγιση βάση των δεδομένων, από κάτω προς τα πάνω (bottom-up). Η ακριβής, εσωτερική λειτουργία του κώδικα προσομοίωσης δεν θεωρείται ότι είναι γνωστή (ή ακόμη και κατανοητή), μόνο η συμπεριφορά εισόδου-εξόδου είναι σημαντική. Ένα μοντέλο κατασκευάζεται με βάση τη μοντελοποίηση της απόκρισης του προσομοιωτή σε έναν περιορισμένο αριθμό έξυπνα επιλεγμέ-

νων σημείων δεδομένων. Αυτή η προσέγγιση είναι επίσης γνωστή ως μοντελοποίηση συμπεριφοράς ή μοντελοποίηση μαύρου κουτιού (black box modeling) [11].

Αν και η χρήση υποκατάστατων μοντέλων αντί πειραμάτων και προσομοιώσεων στον μηχανικό σχεδιασμό είναι πιο συνηθισμένη, τα υποκατάστατα μοντέλα μπορεί να χρησιμοποιηθούν σε πολλούς άλλους τομείς της επιστήμης όπου υπάρχουν ακριβά πειράματα ή αξιολογήσεις λειτουργιών [12].



Σχήμα 3.1: Υποκατάστατα μοντέλα και προσομοιώσεις

Στο Σχήμα 3.1 φαίνεται γιατί η χρήση υποκατάστατων μοντέλων προτιμάται έναντι των προσομοιώσεων.

Ένα υποκατάστατο μοντέλο, εκπαιδεύεται χρησιμοποιώντας μια προσέγγιση βάσει δεδομένων. Τα δεδομένα εκπαίδευσής του λαμβάνονται μέσω ανίχνευσης των εξόδων προσομοίωσης σε πολλές έξυπνα επιλεγμένες θέσεις στον χώρο παραμέτρων σχεδιασμού. Σε καθεμία από αυτές τις θέσεις, διεξάγεται μια πλήρης προσομοίωση για τον υπολογισμό της αντίστοιχης εξόδου της προσομοίωσης. Συγκεντρώνοντας τα ζεύγη εισόδων (παράμετροι σχεδίασης) και τις αντίστοιχες εξόδους τους σε ένα σύνολο δεδομένων εκπαίδευσης, μπορούμε να δημιουργήσουμε ένα στατιστικό μοντέλο με βάση το σύνολο δεδομένων που λήφθηκαν. Τα υποκατάστατα μοντέλα είναι μια ειδική περίπτωση επιβλεπόμενης μηχανικής μάθησης που εφαρμόζεται στον τομέα του μηχανικού σχεδιασμού. Αυτές οι δημοφιλείς τεχνικές μηχανικής μάθησης,

όπως η πολυωνυμική παλινδρόμηση, μηχανές υποστήριξης διανυσμάτων, νευρωνικά δίκτυα κ.λ.π., υιοθετούνται επίσης ευρέως ως υποκατάστατα μοντέλα για την επιτάχυνση των διαδικασιών σχεδιασμού και ανάλυσης προϊόντος [13].

Δειγματοληψία

Ξεκινάμε δημιουργώντας δεδομένα αρχικής εκπαίδευσης. Για να το πετύχουμε αυτό, επιλέγουμε προσεκτικά δείγματα των παραμέτρων σχεδιασμού από τον χώρο παραμέτρων τους. Αυτή η πρακτική είναι επίσης γνωστή ως Σχεδιασμός Πειραμάτων. Σε αυτό το στάδιο, είναι προτιμότερο να υπάρχουν δείγματα που είναι ομοιόμορφα κατανεμημένα σε όλο το χώρο των παραμέτρων. Αυτό είναι θετικό αφού μπορούμε να έχουμε αντιπροσωπευτικές τιμές της προσεγγιστικής σχέσης εισόδου-εξόδου από όλες τις περιοχές του χώρου παραμέτρων που ερευνήθηκε. Συνήθως, τα σχήματα δειγματοληψίας πλήρωσης χώρου χρησιμοποιούνται για τη δημιουργία δειγμάτων με την προαναφερθείσα ιδιότητα. Η διασημότερη μέθοδος είναι ο Latin Hypercube [14].

Latin Hypercube (LHS), είναι μια στρωματοποιημένη τυχαία μέθοδος δειγματοληψίας που αναπτύχθηκε αρχικά για αποτελεσματική αξιολόγηση αβεβαιότητας. Ο LHS χωρίζει τον χώρο των παραμέτρων σε κάδους ίσης πιθανότητας με στόχο να επιτύχει ομοιόμορφη κατανομή των σημείων δειγμάτων στον χώρο παραμέτρων που θα ήταν δυνατή με καθαρή τυχαία δειγματοληψία.

Κατασκευή Υποκατάστατου Μοντέλου

Υπάρχουν παραμετρικές (π.χ. Kriging) και μη παραμετρικές (π.χ. συναρτήσεις ακτινικής βάσης) εναλλακτικές για την κατασκευή υποκατάστατων μοντέλων. Η παραμετρική προσέγγιση προϋποθέτει τη συνολική λειτουργική μορφή της σχέσης μεταξύ της μεταβλητής απόκρισης και της μεταβλητής σχεδιασμού να είναι γνωστές, ενώ οι μη παραμετρικές χρησιμοποιούν διαφορετικούς τύπους, τοπικών μοντέλων σε διαφορετικές περιοχές των δεδομένων για τη δημιουργία ενός συνόλικού μοντέλου [15].

Θετικά

- Ευελιξία στην επιλογή οποιουδήποτε στατιστικού πλαισίου που λειτουργεί ανάλογα το πρόβλημα που θέλουμε να λύσουμε.

- Τα υποκατάστατα μοντέλα είναι απλά και εύκολα στην υλοποίηση τους.
- Τα υποκατάστατα μοντέλα είναι φθηνά και γρήγορα ενώ οι προσομοιώσεις διαρκούν πολύ περισσότερα και απαιτούν πολύ περισσότερα λεφτά.

Αρνητικά

- Κατά την κατασκευή ενός υποκατάστατου, ο χρήστης θα πρέπει να γνωρίζει ότι αυτό είναι ένα μοντέλο εκπαιδευμένο μέσα σε άλλο μοντέλο. Το υποκατάστατο μοντέλο δεν έχει πρόσβαση στις πραγματικές παρατηρήσεις και μπορεί να είναι τόσο καλό όσο το μοντέλο του μαύρου κουτιού. Είναι πιθανό να γίνουν κακές προβλέψεις. Αυτός είναι ο λόγος για τον οποίο ένα ξεχωριστό σύνολο δεδομένων δοκιμών είναι ζωτικής σημασίας, διότι επικυρώνει εάν οι απώλειες είναι μεγάλες.
- Δεν υπάρχει σαφής ορισμός ενός καλού υποκατάστατου μοντέλου. Η μέτρηση της υποκατάστατης απόδοσης απαιτεί κρίση και δεν υπάρχει μέτρηση που να ταιριάζει σε όλους. Μπορεί να είναι δύσκολο να προσδιοριστεί εάν το υποκατάστατο μοντέλο προσεγγίζει με ακρίβεια το μοντέλο του μαύρου κουτιού.
- Το υποκατάστατο μοντέλο μπορεί να έχει διαφορετικά επίπεδα απόδοσης για διαφορετικά υποσύνολα δεδομένων. Δηλαδή, μπορεί να προσεγγίσει τα αποτελέσματα για ένα υποσύνολο δεδομένων, αλλά να είναι ανακριβές για ένα άλλο υποσύνολο δεδομένων. Αυτό θα κάνει το υποκατάστατο να είναι μόνο εν μέρει ερμηνεύσιμο [16].

3.1 Συνάρτηση Ακτινικής Βάσης (Radial Basis Function)

Το υποκατάστατο μοντέλο συνάρτησης ακτινικής βάσης (RBF) αντιπροσωπεύει τη συνάρτηση παρεμβολής ως γραμμικό συνδυασμό συναρτήσεων βάσης, μία για κάθε σημείο εκπαίδευσης. Ο RBF ονομάζεται έτσι επειδή οι συναρτήσεις βάσης εξαρτώνται μόνο από την απόσταση από το σημείο πρόβλεψης έως το σημείο εκπαίδευσης για τη συνάρτηση βάσης. Οι συντελεστές των συναρτήσεων βάσης υπολογίζονται κατά το στάδιο της εκπαίδευσης. Η εξίσωση πρόβλεψης για τον RBF είναι:

$$\boldsymbol{y} = p(\boldsymbol{x})\boldsymbol{w}_p + \sum_{i}^{nt} \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{xt}_i)\boldsymbol{w}_r \tag{3.1}$$

, όπου $x \in \mathbb{R}^{nx}$ είναι το διάνυσμα εισόδου πρόβλεψης, $y \in \mathbb{R}^{nx}$ είναι η πρόβλεψη εξόδου, $xt_i \in \mathbb{R}^{nx}$ είναι το διάνυσμα εισόδου για το σημείο εκπαίδευσης i, $p(x) \in \mathbb{R}^{np}$ είναι το διάνυσμα που αντιστοιχίζει τους πολυωνυμικούς συντελεστές στην έξοδο πρόβλεψης, $\phi(x, xt_i) \in \mathbb{R}^{nt}$ είναι το διάνυσμα που αντιστοιχίζει τους συντελεστές συνάρτησης ακτινικής βάσης στην έξοδο πρόβλεψης, $w_p \in \mathbb{R}^{np}$, είναι το διάνυσμα των πολυωνυμικών συντελεστών και $w_r \in \mathbb{R}^{nt}$ είναι το διάνυσμα των συντελεστών συνάρτησης ακτινικής βάσης. Οι συντελεστές w_p και w_r υπολογίζονται λύνοντας το ακόλουθο γραμμικό σύστημα:

$$\begin{bmatrix} \phi(\mathbf{xt}_1, \mathbf{xt}_1) & \dots & \phi(\mathbf{xt}_1, \mathbf{xt}_{nt}) & \mathbf{p}(\mathbf{xt}_1)^T \\ \vdots & \ddots & \vdots & \vdots \\ \phi(\mathbf{xt}_{nt}, \mathbf{xt}_1) & \dots & \phi(\mathbf{xt}_{nt}, \mathbf{xt}_{nt}) & \mathbf{p}(\mathbf{xt}_{nt})^T \\ \mathbf{p}(\mathbf{xt}_1) & \dots & \mathbf{p}(\mathbf{xt}_{nt}) & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{w_r}_1 \\ \vdots \\ \mathbf{w_r}_{nt} \\ \mathbf{w_p} \end{bmatrix} = \begin{bmatrix} yt_1 \\ \vdots \\ yt_{nt} \\ 0 \end{bmatrix} \tag{3.2}$$

Ένα ιδιαίτερο χαρακτηριστικό της προσέγγισης με το μοντέλο του RBF είναι ότι μπορούν να χρησιμοποιηθούν διαφορετικοί τύποι του RBF (gaussian, linear, cubic). Το SMT χρησιμοποιεί μόνο τις συναρτήσεις βάσης Gauss προς το παρών. Η μαθηματική περιγραφή του φαίνεται στο (3.3)

$$\phi(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(\frac{||\mathbf{x}_i - \mathbf{x}_j||_2^2}{d0^2}\right) \tag{3.3}$$

<u>Θετικά</u>

- Έχει μόνο μία παράμετρο συντονισμού.
- Γρήγορη εκπαίδευση για μικρό αριθμό σημείων εκπαίδευσης.

<u>Αρνητικά</u>

- Επιρρεπείς σε ταλαντώσεις.
- Αριθμητικά προβλήματα όταν τα σημεία είναι πολύ κοντά το ένα στο άλλο.

3.2 Kriging

Ο Kriging είναι ένα μοντέλο παρεμβολής, είναι ένας γραμμικός συνδυασμός μιας γνωστής συνάρτησης $f_i(x)$ που προστίθεται σε μια πραγματοποίηση μιας στοχαστικής διαδικασίας $Z(x)$

$$\hat{y} = \sum_{i=1}^{k} \beta_i f_i(x) + \boldsymbol{Z}(x) \tag{3.4}$$

$\boldsymbol{Z}(x)$ είναι μια πραγματοποίηση μιας στοχαστικής διαδικασίας με μέσο μηδέν και συνάρτηση χωρικής συν-διακύμανσης που δίνεται από:

$$cov[\boldsymbol{Z}(x^{(i)}), \boldsymbol{Z}(x^{(j)})] = \sigma^2 R(x^{(i)}, x^{(j)}) \tag{3.5}$$

, όπου σ^2 είναι η διακύμανση της διαδικασίας και R είναι ο συσχετισμός που ελέγχει την ομαλότητα του μοντέλου Kriging και την επίδραση των κοντινών σημείων. Η Gaussian συνάρτηση είναι η συχνότερη σε χρήση SCF (spatial correlation function) που χρησιμοποιείται συνήθως καθώς παρέχει μια σχετικά ομαλή και άπειρα διαφοροποιήσιμη επιφάνεια [17]:

$$R(x_i, x_j) = R(|x_i, x_j|) = e^{-\frac{|x_i - x_j|^2}{\theta}} \tag{3.6}$$

Τέσσερις τύποι συναρτήσεων συσχέτισης είναι διαθέσιμοι στο SMT.
Συνάρτηση εκθετικής συσχέτισης:

$$\prod_{l=1}^{nx} \exp\left(-\theta_l \left|x_l^{(i)} - x_l^{(j)}\right|\right), \quad \forall\ \theta_l \in \mathbb{R}^+ \tag{3.7}$$

Συνάρτηση τετράγωνης εκθετικής (Gaussian) συσχέτισης:

$$\prod_{l=1}^{nx} \exp\left(-\theta_l \left(x_l^{(i)} - x_l^{(j)}\right)^2\right), \quad \forall\ \theta_l \in \mathbb{R}^+ \tag{3.8}$$

Συνάρτηση συσχέτισης Matérn 5/2:

$$\prod_{l=1}^{nx} \left(1 + \sqrt{5}\theta_l \left|x_l^{(i)} - x_l^{(j)}\right| + \frac{5}{3}\theta_l^2 \left(x_l^{(i)} - x_l^{(j)}\right)^2\right) \exp\left(-\sqrt{5}\theta_l \left|x_l^{(i)} - x_l^{(j)}\right|\right), \quad \forall\ \theta_l \in \mathbb{R}^+ \tag{3.9}$$

Συνάρτηση συσχέτισης Matérn 3/2:

$$\prod_{l=1}^{nx} \left(1+\sqrt{3}\theta_l \left|x_l^{(i)} - x_l^{(j)}\right|\right) \exp\left(-\sqrt{3}\theta_l \left|x_l^{(i)} - x_l^{(j)}\right|\right), \quad \forall\ \theta_l \in \mathbb{R}^+ \tag{3.10}$$

Θετικά

- Ευέλικτη διακύμανση πρόβλεψης.

Αρνητικά

- Απαιτεί πολύ χρόνο εάν ο αριθμός των σημείων εκπαίδευσης είναι μεγάλος.
- Αριθμητικά προβλήματα όταν τα σημεία είναι πολύ κοντά το ένα στο άλλο.

3.3 KPLS (Kriging Partial Least Squares)

Είναι ένα μοντέλο Kriging που χρησιμοποιεί τη μέθοδο των μερικών ελαχίστων τετραγώνων. Ο KPLS είναι ταχύτερος από το Kriging λόγω του μικρού αριθμού υπερπαραμέτρων που πρέπει να εκτιμηθούν διατηρώντας παράλληλα μια καλή ακρίβεια. Αυτό το μοντέλο είναι κατάλληλο για προβλήματα υψηλών διαστάσεων λόγω του πυρήνα (kernel) που κατασκευάστηκε μέσω της μεθόδου PLS(μερικά ελάχιστα τετράγωνα). Η μέθοδος των μερικών ελαχίστων τετραγώνων είναι ένα πολύ γνωστό εργαλείο για προβλήματα υψηλών διαστάσεων που αναζητά την κατεύθυνση που μεγιστοποιεί τη διακύμανση μεταξύ των μεταβλητών εισόδου και εξόδου. Αυτό γίνεται με μια προβολή σε ένα μικρότερο χώρο που εκτείνεται από τα λεγόμενα κύρια στοιχεία. Οι πληροφορίες PLS ενσωματώνονται στον πίνακα συσχέτισης Kriging για να κλιμακωθεί ο αριθμός των εισόδων μειώνοντας τον αριθμό των υπερπαραμέτρων. Ο αριθμός των κύριων στοιχείων h , που αντιστοιχεί στον αριθμό των υπερπαραμέτρων για το KPLS και πολύ μικρότερος από (αριθμός διάστασης του προβλήματος), συνήθως δεν υπερβαίνει τα 4 στοιχεία:

$$\prod_{k=1}^{h} \prod_{l=1}^{nx} e(-\theta_k (w_{*l}^{(k)} x_l^{(i)} - w_{*l}^{(k)} x_l^{(j)})^2) \tag{3.11}$$

Θετικά

- Διακύμανση πρόβλεψης, γρήγορη κατασκευή.
- Κατάλληλο για προβλήματα πολλών διαστάσεων.

Αρνητικά

- Αριθμητικά προβλήματα όταν τα σημεία είναι πολύ κοντά το ένα στο άλλο.

3.4 KPLSK(Kriging Partial Least Squares In Two Steps)

Ο KPLSK είναι ένα μοντέλο που βασίζεται στον KPLS και είναι βασικά κατασκευασμένο σε δύο βήματα. Το πρώτο βήμα συνίσταται στην εκτέλεση του KPLS και στην εκτίμηση υπερπαραμέτρων που εκφράζονται στον μειωμένο χώρο με έναν αριθμό διαστάσεων ίσο με h. Το δεύτερο βήμα συνίσταται στην έκφραση των υπερπαραμέτρων εκτίμησης στον αρχικό χώρο με έναν αριθμό διαστάσεων ίσο με nx. Στη συνέχεια, το χρησιμοποιεί ως σημείο εκκίνησης για την τοπική βελτιστοποίηση της συνάρτησης πιθανότητας ενός τυπικού Kriging μοντέλου. Η ιδέα εδώ είναι να μαντέψουμε μια «καλή» αρχική υπερπαράμετρο και να εφαρμόσουμε μια βελτιστοποίηση που βασίζεται σε κλίση χρησιμοποιώντας έναν κλασικό πυρήνα(kernel) Kriging. Η «καλή» εικασία θα παρέχεται από το KPLS: οι λύσεις $(\theta_1^*, ..., \theta_h^*)$ και οι PLS-συντελεστές $(w_1^k, ..., w_{nx}^k)$ για $k = 1, ..., h$. Με αλλαγή μεταβλητών $\eta\iota = \sum_{k=1}^{h} \theta_k^* w_l^{(k)^2}$, για $l = 1, ..., nx$ μπορούμε να εκφράσουμε το αρχικό σημείο υπερπαραμέτρων στον αρχικό χώρο.

Θετικά

- Διακύμανση πρόβλεψης, γρήγορη κατασκευή.
- Κατάλληλο για προβλήματα πολλών διαστάσεων.

Αρνητικά

- Απαιτεί πολύ χρόνο εάν ο αριθμός των σημείων εκπαίδευσης είναι μεγάλος.
- Αριθμητικά προβλήματα όταν τα σημεία είναι πολύ κοντά το ένα στο άλλο.

3.5 Προσέγγιση Ελάχιστων Τετραγώνων

Η μέθοδος των Ελαχίστων τετραγώνων ταιριάζει σε ένα γραμμικό μοντέλο με συντελεστές $\beta = (\beta_0, \beta_1, ..., \beta_{nx})$ για να ελαχιστοποιηθεί το υπολειπόμενο άθροισμα

των τετραγώνων μεταξύ των παρατηρούμενων αποκρίσεων στο σύνολο δεδομένων και των αποκρίσεων που προβλέπονται από τη γραμμική προσέγγιση. Μαθηματικά λύνει ένα πρόβλημα της μορφής: $min_\beta \parallel \mathbf{X}\beta - \mathbf{y} \parallel_2^2$, όπου $\mathbf{X} = (1, x^{(1)^T}, ..., x^{(nt)^T})^T$ με διαστάσεις $nt \times nx + 1$ [18].

Θετικά

- Απλή και γρήγορη κατασκευή.

Αρνητικά

- Κάνει ακριβής προβλέψεις μόνο για γραμμικά προβλήματα.

3.6 Στάθμιση αντίστροφης απόστασης

Το μοντέλο στάθμισης αντίστροφης απόστασης (IDW) είναι μια μέθοδος παρεμβολής και τα άγνωστα σημεία υπολογίζονται με σταθμισμένο μέσο όρο των σημείων δειγματοληψίας. Η εξίσωση πρόβλεψης για το IDW είναι [19]:

$$y = \begin{cases} \frac{\sum_i^{nt} \beta(\mathbf{x},\mathbf{xt}_i) yt_i}{\sum_i^{nt} \beta(\mathbf{x},\mathbf{xt}_i)}, & \text{για} \quad \mathbf{x} \neq \mathbf{xt}_i \quad \forall i \\ yt_i & \text{για} \quad \mathbf{x} = \mathbf{xt}_i \quad \text{για κάποιο } i \end{cases}, \tag{3.12}$$

όπου $x \in \mathbb{R}$ είναι το διάνυσμα εισόδου πρόβλεψης, $y \in \mathbb{R}$ είναι η έξοδος πρόβλεψης, $xt_i \in mathbbR$ είναι το διάνυσμα εισόδου για το i-οστό σημείο εκπαίδευσης και τέλος το $yt_i \in \mathbb{R}$ είναι η τιμή εξόδου για το i-οστό σημείο εκπαίδευσης. Η συνάρτηση στάθμισης β ορίζεται ως:

$$\beta(\mathbf{x}_i, \mathbf{x}_j) = \parallel \mathbf{x}_i - \mathbf{x}j \parallel_2^{-p}, \tag{3.13}$$

όπου p είναι θετικός πραγματικός αριθμός, ονομάζεται παράμετρος ισχύος. Αυτή η παράμετρος πρέπει να είναι αυστηρά μεγαλύτερη από 1 για να είναι συνεχείς οι παράγωγοι.

Θετικά

- Δεν απαιτείται εκπαίδευση.

Αρνητικά

- Δεν έχει καλή ακρίβεια.

3.7 Πολυωνυμική προσέγγιση δεύτερης τάξης

Το πολυωνυμικό μοντέλο δεύτερης τάξης (QP) μπορεί να εκφραστεί με

$$\mathbf{y} = \mathbf{X}\beta + \epsilon, \tag{3.14}$$

όπου ϵ είναι ένα διάνυσμα τυχαίων σφαλμάτων.Το διάνυσμα των εκτιμώμενων συντελεστών πολυωνυμικής παλινδρόμησης με χρήση της συνήθους εκτίμησης του ελάχιστου τετραγώνου είναι:

$$\beta = \mathbf{X^T X}^{-1}\mathbf{X^T y} \tag{3.15}$$

και τέλος :

$$\mathbf{X} = \begin{bmatrix} 1 & \dots x_1^{(1)} & \dots & x_{nx}^{(1)} & x_1^{(1)}x_2^{(1)} & \dots & x_{nx-1}^{(1)}x_{nx}^{(1)} & x_1^{(1)^2} & \dots & x_{nx}^{(1)^2} \\ \vdots & \vdots & \dots & \vdots & \vdots & \dots & \vdots & \vdots & \dots & \vdots \\ 1 & \dots x_1^{(nt)} & \dots & x_{nx}^{(nt)} & x_1^{(nt)}x_2^{(nt)} & \dots & x_{nx-1}^{(nt)}x_{nx}^{(nt)} & x_1^{(nt)^2} & \dots & x_{nx}^{(nt)^2} \end{bmatrix} \tag{3.16}$$

Θετικά

- Απλή και γρήγορη κατασκευή

Αρνητικά

- Απαιτείται μεγάλος αριθμός σημείων για μεγάλο αριθμό εισόδων.

3.8 Υποστήριξη Διανυσματικών Μηχανών (SVM)

Τα SVM λύνουν προβλήματα δυαδικής ταξινόμησης διατυπώνοντας τα ως κυρτά προβλήματα βελτιστοποίησης [20]. Το πρόβλημα βελτιστοποίησης συνεπάγεται στην εύρεση του μέγιστου περιθωρίου που χωρίζει το υπερπλάνο (hyperplane), ενώ ταξινομεί σωστά όσο το δυνατόν περισσότερα σημεία εκπαίδευσης. Στον Αλγόριθμο 1 δίνεται μια απλή μορφή ενός SVM αλγορίθμου [21]. Ο αλγόριθμος σταματά όταν όλα τα σημεία ταξινομούνται σε ένα όριο σφάλματος που ορίζουμε, για παράδειγμα $y_i f(x_i) > 1 - \epsilon$.

while *there are violating points* **do**
  Find a violator $\leftarrow violator$
  $candidateSV = candidateSV \cup violator$
  if $a_p < 0$ **then**
    $candidateSV = candidateSV/p$
    repeat till all such points are pruned
end

Αλγόριθμος 1: Απλή μορφή αλγορίθμου SVM

Θετικά

- Είναι αποτελεσματικό σε προβλήματα πολλών διαστάσεων.

Αρνητικά

- Δεν είναι κατάλληλο για προβλήματα με πολλά δεδομένα.
- Κίνδυνος υπερπροσαρμογής.

3.8.1 Υποστήριξη Διανυσματικής Κατηγοριοποίησης (SVC)

Ο αλγόριθμος SVC παράγει μια μη βαθμονομημένη τιμή που δεν είναι πιθανότητα. Έστω η μη κατωφλημένη έξοδος ενός SVC.

$$f(x) = h(x) + b \tag{3.17}$$

, όπου:

$$h(x) = \sum_i y_i \alpha_i k(x_i, x) \tag{3.18}$$

Όταν ο SVC γίνεται εκπαίδευση ελαχιστοποιεί μια συνάρτηση σφάλματος που προσθέτει ποινές σε μια προσέγγιση του ποσοστού εσφαλμένης ταξινόμησης εκπαίδευσης.

$$C \sum_i (1 - y_i fi)_+ + \frac{1}{2} \parallel h \parallel_F \tag{3.19}$$

Η ελαχιστοποίηση αυτής της συνάρτησης σφάλματος θα ελαχιστοποιήσει επίσης ένα όριο στην εσφαλμένη ταξινόμηση της δοκιμής, το οποίο είναι ένας επιθυμητός στόχος [22].

3.8.2 Υποστήριξη Διανυσματικής Παλινδρόμησης (SVR)

Το SVM μπορεί να χρησιμοποιηθεί και ως μέθοδος παλινδρόμησης, διατηρώντας όλα τα κύρια χαρακτηριστικά που χαρακτηρίζουν τον αλγόριθμο (μέγιστο περιθώριο). Το Support Vector Regression (SVR) χρησιμοποιεί τις ίδιες αρχές με το SVM για ταξινόμηση, με λίγες μόνο μικρές διαφορές. Πρώτα απ 'όλα, επειδή η έξοδος είναι ένας πραγματικός αριθμός, γίνεται πολύ δύσκολο να προβλέψουμε τις διαθέσιμες πληροφορίες, οι οποίες έχουν πολλές περισσότερες δυνατότητες. Στην περίπτωση παλινδρόμησης, ορίζεται ένα περιθώριο ανοχής (ϵ). Ωστόσο, η κύρια ιδέα είναι πάντα η ίδια: ελαχιστοποίηση του σφάλματος, εξατομίκευση του υπερεπιπέδου που μεγιστοποιεί το περιθώριο, έχοντας κατά νου ότι μέρος του σφάλματος είναι ανεκτό.

Υποστήριξη Διανυσματικής Παλινδρόμησης Ελάχιστων Τετραγώνων (LS-SVR)

Είναι μια έκδοση ελαχίστων τετραγώνων των μηχανών διανυσμάτων υποστήριξης (SVM). Στα μοντέλα LS-SVR, η λύση βρίσκεται λύνοντας ένα σύνολο γραμμικών εξισώσεων αντί ενός προβλήματος κυρτού τετραγωνικού προγραμματισμού (QP). Οι ταξινομητές SVM ελάχιστων τετραγώνων προτάθηκαν από τους Johan Suykens και Joos Vandewalle [23]. Τα LS-SVR είναι μια κατηγορία μεθόδων μάθησης που βασίζονται στον πυρήνα (kernel) [24]. Παρακάτω ακολουθεί εν συντομία πως υλοποιείται ο αλγόριθμος 2.

1.Με γνωστά δεδομένα εκπαίδευσης $xk, yk_{k=1}^{N}$, βρίσκει έναν βέλτιστο συνδυασμό (γ, σ) (π.χ. με 10πλάσια όρια διασταυρούμενης επικύρωσης ή γενίκευσης). Για τον βέλτιστο συνδυασμό (γ, σ) υπολογίζει το $e_k = \frac{a_k}{\gamma}$ από το σύστημα 3.20.
2. Υπολογίζει το $\hat{s}$ από την κατανομή e_k.
3. Προσδιορίζει τα βάρη v_k με βάση τα $e_k, \hat{s}$.
4. Λύνοντας το σταθμισμένο LS-SVR, δίνοντας το μοντέλο $y(x) = \sum_{k=1}^{N} a_k^{\star} K(x, x_k) + b^{\star}$ 3.22.

Αλγόριθμος 2: Αλγόριθμος LS-SVR

$$\left[\begin{array}{c|c} 0 & 1_{\nu}^{T} \\ \hline 1 & \Omega + V_{\gamma} \end{array}\right] \left[\begin{array}{c} b^{\star} \\ \hline a^{\star} \end{array}\right] = \left[\begin{array}{c} 0 \\ \hline y \end{array}\right] \tag{3.20}$$

όπου ο διαγώνιος πίνακας V_γ δίνεται από:

$$V_\gamma = diag\left\{\frac{1}{\gamma\nu_1}, \ldots, \frac{1}{\gamma\nu_N}\right\} \tag{3.21}$$

Η επιλογή των βαρών V_k καθορίζεται με βάση τις μεταβλητές σφάλματος $e_k = \frac{a_k}{\gamma}$.

Το προκύπτον μοντέλο LS-SVR για την εκτιμώμενη συνάρτηση γίνεται:

$$y(x) = \sum_{k=1}^{N} a_k K(x, x_k) + b \tag{3.22}$$

όπου a, b είναι οι λύσεις του 3.20.

3.9 Τυχαία Δάση

Τα τυχαία δάση είναι μια μέθοδος εκμάθησης συνόλου για ταξινόμηση και παλινδρόμηση, που κατά την εκπαίδευση κατασκευάζει ένα πλήθος δέντρων αποφάσεων κατά το χρόνο εκπαίδευσης. Για εργασίες ταξινόμησης, η έξοδος του τυχαίου δάσους είναι η κλάση που επιλέγεται από τα περισσότερα δέντρα. Για εργασίες παλινδρόμησης, επιστρέφεται ο μέσος όρος ή ο μέσος όρος πρόβλεψης των μεμονωμένων δέντρων.

function RandomForest(S, F)
  $H \leftarrow \emptyset$
  for $i \in 1, ..., B$ **do**
    $S^{(i)} \leftarrow$ A bootstrap sample from S
    $h_{(i)} \leftarrow DecisionTree(S^{(i)}, F)$
    $H \leftarrow \cup\{h_i\}$
  end for
  return H
end function

Αλγόριθμος 3: Αλγόριθμος Random Forest

function DecisionTree(S, F)
  At each node:
    $f \leftarrow$ very small subset of F
    Split on best feature in f
  return The learned Tree
end function

Αλγόριθμος 4: Αλγόριθμος Δέντρου Απόφασης

Στον Αλγόριθμο 3 φαίνεται μια απλή υλοποίηση του Random Forest.

3.9.1 Εκπαίδευση του Random Forest

Ο Random Forest εκπαιδεύεται μέσω της μεθόδου του bagging. Το Bagging ή Bootstrap Aggregating, αποτελείται από την τυχαία δειγματοληψία υποσυνόλων των δεδομένων εκπαίδευσης, την προσαρμογή ενός μοντέλου σε αυτά τα μικρότερα σύνολα δεδομένων και τη συγκέντρωση των προβλέψεων. Αυτή η μέθοδος επιτρέπει την επανειλημμένη χρήση πολλών περιπτώσεων για το στάδιο της εκπαίδευσης δεδομένου ότι κάνουμε δειγματοληψία με αντικατάσταση. Η συσσώρευση δέντρων αποτελείται από τη δειγματοληψία υποσυνόλων του συνόλου εκπαίδευσης, την τοποθέτηση ενός Δέντρου απόφασης σε καθένα και τη συγκέντρωση των αποτελεσμάτων τους [25].

3.9.2 Εντροπία στα Δέντρα Απόφασης

Η εντροπία είναι ένα μέτρο της αταξίας ή της αβεβαιότητας και ο στόχος των μοντέλων μηχανικής μάθησης γενικά είναι η μείωση της αβεβαιότητας. Χρησιμοποιεί την πιθανότητα ενός συγκεκριμένου αποτελέσματος προκειμένου να λάβει μια απόφαση σχετικά με τον τρόπο διακλάδωσης του κόμβου.

$$Entropy = \sum_{i=1}^{C} -p_i log_2(p_i) \tag{3.23}$$

```
def entropy(y):
    hist = y[np.array(y).nonzero()[0]]
    ps = hist / len(y)
    # Caclulate entropy
    return -np.sum([p * np.log2(p)for p in ps if p >0])
```

Αλγόριθμος 5: Συνάρτηση υπολογισμού της εντροπίας.

Στο Αλγόριθμο 5 φαίνεται η υλοποίηση που χρησιμοποίησα για να υπολογίζω την εντροπία.

Θετικά

- Μπορεί να χρησιμοποιηθεί για την επίλυση προβλημάτων τόσο ταξινόμησης όσο και παλινδρόμησης.

- Μπορεί να χειριστεί αυτόματα τιμές που λείπουν.

Αρνητικά

- Αποτυγχάνει να προσδιορίσει τη σημαντικότητα κάθε μεταβλητής.
- Ένας μεγάλος αριθμός δέντρων μπορεί να κάνει τον αλγόριθμο πολύ αργό και αναποτελεσματικό.

3.10 SMT (Surrogate Modeling Toolbox)

Το SMT είναι ένα πακέτο σε γλώσσα προγραμματισμού Python ανοιχτού κώδικα που αποτελείται από βιβλιοθήκες υποκατάστατων μεθόδων μοντελοποίησης (π.χ. rbf, Kriging), μεθόδους δειγματοληψίας και προβλήματα συγκριτικής αξιολόγησης. Το SMT έχει σχεδιαστεί για να διευκολύνει τους προγραμματιστές να εφαρμόσουν νέα υποκατάστατα μοντέλα σε μια καλά δοκιμασμένη, καλά τεκμηριωμένη πλατφόρμα και να έχουν μια βιβλιοθήκη υποκατάστατων μεθόδων μοντελοποίησης με την οποία να χρησιμοποιούν και για να συγκρίνουν μεθόδους [26].

Το SMT είναι διαφορετικό από τις υπάρχων βιβλιοθήκες υποκατάστατων μοντελοποίησης λόγω της έμφασης που δίνει σε παράγωγους, συμπεριλαμβανομένης της εκπαίδευσης των παράγωγων που χρησιμοποιούνται για μοντελοποίηση. Περιλαμβάνει επίσης μοναδικά υποκατάστατα μοντέλα: Kriging ελαχίστων τετραγώνων (KPLS), η οποία κλιμακώνεται καλά με τον αριθμό των εισόδων και την ενεργειακή ελαχιστοποίηση της παρεμβολής spline, η οποία κλιμακώνεται καλά με τον αριθμό των σημείων εκπαίδευσης.

Το SMT περιέχει μια βιβλιοθήκη μεθόδων δειγματοληψίας που χρησιμοποιούνται για τη δημιουργία συνόλων σημείων στο χώρο εισόδου, είτε για εκπαίδευση είτε για πρόβλεψη.

Τυχαία δειγματοληψία (Random Sampling): αυτή η κλάση δημιουργεί τυχαία δείγματα από ομοιόμορφη κατανομή πάνω από το χώρο του σχεδιασμού.

Πλήρης παραγοντική δειγματοληψία (Full-factorial Sampling): αυτή είναι μια κοινή μέθοδος δειγματοληψίας όπου όλες οι μεταβλητές εισόδου ορίζονται σε δύο επίπεδα το καθένα. Αυτά τα επίπεδα συνήθως συμβολίζονται με +1 και -1 για υψηλά και χαμηλά επίπεδα, αντίστοιχα. Η πλήρης παραγοντική δειγματοληψία υπολογίζει

όλα τα δυνατά συνδυασμούς αυτών των επιπέδων σε όλες αυτές τις μεταβλητές εισόδου.

Latin-Hypercube: αυτή είναι μια στατιστική μέθοδος για τη δημιουργία ενός ημί-τυχαίας δειγματοληψίας. Είναι από τις πιο δημοφιλείς τεχνικές δειγματοληψίας χάρη στην απλότητα και τις ιδιότητες προβολής του σε προβλήματα υψηλών διαστάσεων.

3.11 TensorFlow Neural Network

Το TensorFlow είναι μια βιβλιοθήκη, σχεδιασμένο από την ομάδα της Google για την εφαρμογή εννοιών μηχανικής μάθησης και βαθιάς μάθησης με τον πιο εύκολο τρόπο. Συνδυάζει την υπολογιστική άλγεβρα των τεχνικών βελτιστοποίησης για τον εύκολο υπολογισμό πολλών μαθηματικών παραστάσεων [27].

3.11.1 Σημαντικά χαρακτηριστικά του TensorFlow

- Περιλαμβάνει ένα χαρακτηριστικό που ορίζει, βελτιστοποιεί και υπολογίζει μαθηματικές εκφράσεις εύκολα με τη βοήθεια πολυδιάστατων πινάκων που ονομάζονται τανυστές (Tensors).

- Περιλαμβάνει μια υψηλή κλιμακούμενη δυνατότητα υπολογισμού με διάφορα σύνολα δεδομένων.

- Το TensorFlow χρησιμοποιεί υπολογιστές GPU, αυτοματοποιώντας τη διαχείριση. Περιλαμβάνει επίσης μια μοναδική δυνατότητα βελτιστοποίησης της ίδιας μνήμης και των δεδομένων που χρησιμοποιούνται.

- Μετρήσεις (Metrics), προκειμένου να αξιολογηθεί η απόδοση των μοντέλων μηχανικής εκμάθησης, το TensorFlow παρέχει στο API πρόσβαση σε μετρήσεις που χρησιμοποιούνται συνήθως. Παραδείγματα περιλαμβάνουν διάφορες μετρήσεις ακρίβειας (binary, categorical, sparse categorical) μαζί με άλλες μετρήσεις όπως η Precision, Recall, and Intersection-over-Union (IoU).

- Βελτιστοποιητές (Optimizers), προσφέρει ένα σύνολο βελτιστοποιητών για εκπαίδευση νευρωνικών δικτύων, συμπεριλαμβανομένων των ADAM, ADAGRAD και Stochastic Gradient Descent (SGD). Κατά την εκπαίδευση ενός μοντέλου,

διαφορετικοί βελτιστοποιητές προσφέρουν διαφορετικούς τρόπους ρύθμισης παραμέτρων, επηρεάζοντας συχνά τη σύγκλιση και την απόδοση ενός μοντέλου.

3.11.2 Συναρτήσεις Ενεργοποίησης

Μια συνάρτηση ενεργοποίησης αποφασίζει εάν ένας νευρώνας πρέπει να ενεργοποιηθεί ή όχι. Αυτό σημαίνει ότι θα αποφασίσει εάν η είσοδος του νευρώνα στο δίκτυο είναι σημαντική ή όχι στη διαδικασία πρόβλεψης χρησιμοποιώντας απλούστερες μαθηματικές πράξεις. Ο ρόλος της συνάρτησης ενεργοποίησης είναι να εξάγει έξοδο από ένα σύνολο τιμών εισόδου που τροφοδοτούνται σε έναν κόμβο (ή ένα επίπεδο).

- **Συνάρτηση Γραμμικής Ενεργοποίησης**, επίσης γνωστή ως ''χωρίς ενεργοποίηση'' ή ''συνάρτηση ταυτότητας'', είναι όπου η ενεργοποίηση είναι ανάλογη με την είσοδο. Η συνάρτηση δεν κάνει τίποτα στο σταθμισμένο άθροισμα της εισόδου, απλώς επιστρέφει την τιμή που της δόθηκε.

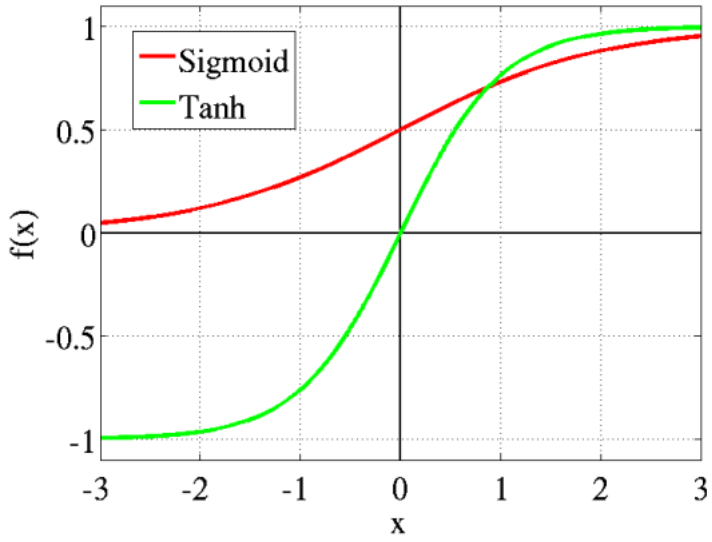
$$f(x) = x \tag{3.24}$$

- **Ενεργοποίηση Sigmoid**: η συνάρτηση σιγμοειδούς μας δίνει αποτελέσματα στο (0, 1), την καθιστά ιδανική για προβλήματα δυαδικής ταξινόμησης όπου πρέπει να βρούμε την πιθανότητα τα δεδομένα να ανήκουν σε μια συγκεκριμένη κλάση.

$$S(x) = \frac{1}{1 + e^{-x}} \tag{3.25}$$

  Στα Σχήματα 3.2 και 3.3 φαίνεται γραφικά η σιγμοειδή συνάρτηση.

- **Συνάρτηση Ενεργοποίησης Υπερβολικής Εφαπτομένης (tanh)**, είναι παρόμοια με τη σιγμοειδή (sigmoid) συνάρτηση και έχει ακόμη και το ίδιο σχήμα S με διαφορά στο εύρος εξόδου από (-1, 1). Στη συνάρτηση ενεργοποίησης υπερβολικής εφαπτομένης, όσο μεγαλύτερη είναι η είσοδος, τόσο πιο κοντά η τιμή εξόδου θα είναι στο 1.0, ενώ Όσο μικρότερη είναι η είσοδος (πιο αρνητική), τόσο πιο κοντά θα είναι η έξοδος στο -1.0.



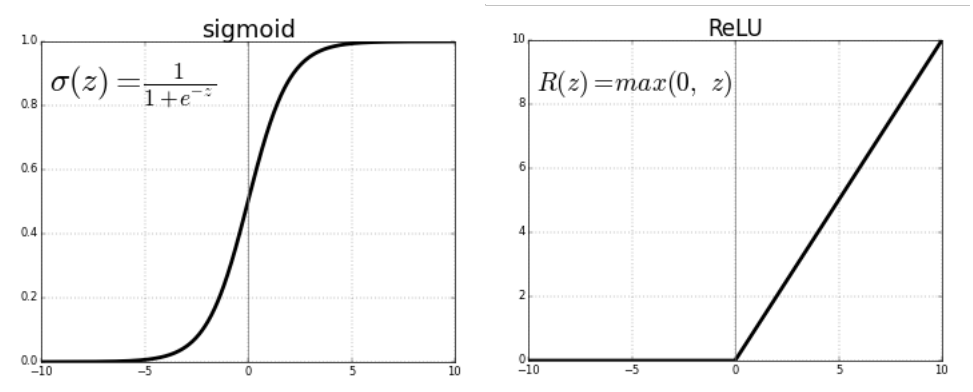
Σχήμα 3.2: tanh vs sigmoid

Στο Σχήμα 3.2 φαίνεται γραφικά η ομοιότητα των δύο συναρτήσεων ενεργοποίησης.

- **Ενεργοποίηση ReLU**: Η Διορθωμένη Γραμμική Μονάδα (Rectified Linear Unit) είναι η πιο συχνή χρησιμοποιούμενη λειτουργία ενεργοποίησης στη βαθιά μάθηση. Η συνάρτηση επιστρέφει 0 εάν η είσοδος είναι αρνητική, αλλά για οποιαδήποτε θετική είσοδο, επιστρέφει αυτήν την τιμή πίσω.

$$R(z) = max(0, z) \tag{3.26}$$

  Στο Σχήμα 3.3 φαίνεται ότι η συνάρτηση ενεργοποίησης ReLU είναι κατά το ήμισυ ανορθωμένη (από κάτω). Το R(z) είναι μηδέν όταν το z είναι μικρότερο από μηδέν και το R(z) είναι ίσο με z όταν το z είναι πάνω ή ίσο με το μηδέν. Επίσης φαίνεται γραφικά η διαφορά της με τη σιγμοειδή συνάρτηση.



Σχήμα 3.3: ReLU vs sigmoid

- **Ενεργοποίηση Softmax**, η Softmax χρησιμοποιείται συχνά ως ενεργοποίηση για το τελευταίο επίπεδο ενός δικτύου ταξινόμησης επειδή το αποτέλεσμα θα μπορούσε να ερμηνευτεί ως κατανομή πιθανότητας.

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{3.27}$$

3.11.3 Γιατί το Tensorflow είναι τόσο διάσημο

Το TensorFlow είναι καλά δημοσιευμένο (documented) και περιλαμβάνει πολλές βιβλιοθήκες μηχανικής μάθησης. Περιλαμβάνει ποικιλία αλγορίθμων μηχανικής μάθησης και βαθιάς μάθησης (deep learning). Το TensorFlow μπορεί να εκπαιδεύσει και να εκτελέσει βαθιά νευρωνικά δίκτυα για αναγνώριση εικόνας, ενσωμάτωση λέξεων και δημιουργία διαφόρων μοντέλων ακολουθιών.

3.11.4 Δομή δεδομένων τανυστή

Οι τανυστές χρησιμοποιούνται ως βασικές δομές δεδομένων στο TensorFlow. Οι τανυστές αντιπροσωπεύουν τα συνδετικά άκρα σε οποιοδήποτε διάγραμμα ροής που ονομάζεται Γράφημα ροής δεδομένων. Οι τανυστές ορίζονται ως πολυδιάστατος πίνακας ή λίστα. Οι τανυστές αναγνωρίζονται από τις ακόλουθες τρεις παραμέτρους:

- **Τάξη (Rank)**, η μονάδα διαστάσεων που περιγράφεται μέσα στον τανυστή ονομάζεται τάξη. Προσδιορίζει τον αριθμό των διαστάσεων του τανυστή. Μια τάξη ενός τανυστή μπορεί να περιγραφεί ως η σειρά ή οι n-διαστάσεις ενός τανυστή που ορίζεται.

- **Σχήμα (Shape)**, ο αριθμός των σειρών και των στηλών μαζί καθορίζει το σχήμα του τανυστή.
- **Τύπος (Type)**, περιγράφει τον τύπο δεδομένων που έχει εκχωρηθεί στα στοιχεία του τανυστή.

3.12 Alamo

Το ALAMO είναι ένα ισχυρό λογισμικό μηχανικής εκμάθησης που χρησιμοποιείται από περισσότερους από 500 μηχανικούς και επιστήμονες δεδομένων σε βιομηχανίες όπως η αυτοματοποίηση, τα φαρμακευτικά προϊόντα, το λογισμικό υπολογιστών από την κυκλοφορία του το 2018 [28].

Το ALAMO χρησιμοποιεί μεθόδους μηχανικής μάθησης για να δημιουργήσει γρήγορα ακριβή, ερμηνεύσιμα μοντέλα προσομοίωσης ή πειραματικού συστήματος μαύρου κουτιού(black-box).

3.12.1 IDAES Alamopy

Ο σκοπός του ALAMOPY (Automatic Learning of Algebraic MOdels PYthon wrapper) είναι καλείτε το λογισμικό ALAMO μέσω της python το οποίο δημιουργεί αλγεβρικά υποκατάστατα μοντέλα συστημάτων μαύρου κουτιού για τα οποία είναι διαθέσιμος προσομοιωτής ή πειραματική ρύθμιση. Θεωρήστε ένα σύστημα για το οποίο οι έξοδοι z είναι μια άγνωστη συνάρτηση f των εισόδων του συστήματος x. Το λογισμικό προσδιορίζει μια συνάρτηση f, δηλαδή μια σχέση μεταξύ των εισόδων και των εξόδων του συστήματος, που ταιριάζει καλύτερα με δεδομένα που συλλέγονται μέσω προσομοίωσης ή πειραματισμού [29].

Κεφάλαιο 4

Υπολογιστική Μελέτη

Σε αυτήν την ενότητα θα παρουσιαστούν τα αποτελέσματα των αλγορίθμων που αναλύθηκαν στο Κεφάλαιο 3 και θα σχολιαστούν τα αποτελέσματα τους.

Η ανάπτυξη του κώδικα και η εκτέλεση του έγινε σε υπολογιστή με λογισμικό Windows 11 Home και επεξεργαστή Intel Core i3-10100 CPU @ 3.60GHz με 8GB RAM. Για τη δημιουργία του κώδικα χρησιμοποιήθηκε το IDE PyCharm Community Edition 2021.2.3 της JetBrains με Python Interpreter έκδοσης Python 3.9. Δέκα αλγόριθμοι για υποκατάστατα αλγόριθμοι υλοποιήθηκαν οι οποίοι αναλύθηκαν στο Κεφάλαιο 3. Επίσης χρησιμοποιήθηκε το λογισμικό Alamo για να προσδιοριστεί η γενικευμένη συνάρτηση. Συγκεκριμένα:

1. Συνάρτηση Ακτινικής Βάσης (RBF)
2. Kriging
3. Kriging Μερικών Ελάχιστων Τετραγώνων (KPLS)
4. Kriging Μερικών Ελάχιστων Τετραγώνων σε δύο βήματα (KPLSK)
5. Προσέγγιση Ελάχιστων Τετραγώνων (LS)
6. Στάθμιση αντίστροφης απόστασης (IDW)
7. Πολυωνυμική προσέγγιση δεύτερης τάξης (QP)
8. Υποστήριξη Διανυσματικών Μηχανών (SVR - SVC)
9. Τυχαία Δάση (Random Forest)
10. Νευρωνικά Δίκτυα με χρήση του Tensorflow (Neural Network)

11. Alamo

Τα δεδομένα από τον Πίνακα 4.1 είναι από το UCI Machine Learning Repository [9]. Επίσης χρησιμοποιήθηκαν προβλήματα παλινδρόμησης από το minlplib [10], για την ακρίβεια 255 προβλήματα τα οποία έχουν όλα περίπου 2500 γραμμές ενώ οι στήλες διαφέρουν από πρόβλημα σε πρόβλημα. Για κάθε σύνολο δεδομένων εφαρμόστηκε τεχνική αντιστοίχησης (mapping) για τις τιμές που δεν ήταν σε μορφή αριθμών.

Για τον υπολογισμό της απόδοσης κάθε αλγορίθμου χρησιμοποιήθηκαν μετρικές τιμές από το sklearn [30]. Για να υπολογιστούν αυτές οι τιμές τα δεδομένα έγιναν split και κράτησε το 20% για test δεδομένα με σκοπό να τα συγκρίνει κατά πόσο η πρόβλεψη είναι κοντά στην πραγματική τιμή. Κάθε αλγόριθμος έτρεξε 10 φορές για κάθε πρόβλημα και τα αποτελέσματα τα οποία κρατήθηκαν είναι ο μέσος όρος για κάθε μετρική τιμή και ο μέσος όρος του χρόνου για κάθε αλγόριθμο. Αυτό έγινε διότι υπάρχει η περίπτωση σε κάποιον αλγόριθμο να υπάρξει καλό/ κακό split και να βγάλει πολύ καλό/ κακό αποτέλεσμα μία φορά αλλά να μην είναι η γενική περίπτωση.

Τέλος ανάλογα των τύπο του προβλήματος (παλινδρόμηση ή κατηγοριοποίηση) έγιναν οι απαραίτητες αλλαγές σε κάθε αλγόριθμο. Για παράδειγμα στα SVM χρησιμοποιήθηκε ο SVR για παλινδρόμηση ενώ για κατηγοριοποίηση χρησιμοποιήθηκε ο SVC. Οι αλγόριθμοι RBF, Kriging. KPLS, KPLSK, LS, IDW, QP είναι κώδικες του SMT. Ο RandomForest και οι αλγόριθμοι SVM υλοποιήθηκαν με τη βοήθεια βιβλιοθηκών όπως το sklearn. Επίσης για να γίνουν οι γραφικές απεικονίσεις των αποτελεσμάτων χρησιμοποιήθηκαν τα boxplot (παράσταση πλαισίου) μέσω της βιβλιοθήκης Matplotlib. Για να υλοποιηθούν τα νευρωνικά δίκτυα χρησιμοποιήθηκε η βιβλιοθήκη TensorFlow.
Μετρικές τιμές για Παλινδρόμηση:

- R2 score (καλύτερη τιμή είναι η μονάδα)
- Mean Squarred Error (καλύτερη τιμή είναι το μηδέν)

Μετρικές τιμές για Κατηγοριοποίηση:

- Accuracy score (καλύτερη τιμή είναι η μονάδα)

- F1 score (καλύτερη τιμή είναι η μονάδα και χειρότερη το μηδέν)
- Precision score (καλύτερη τιμή είναι η μονάδα και χειρότερη το μηδέν)
- Recall score (καλύτερη τιμή είναι η μονάδα και χειρότερη το μηδέν)

Παρακάτω στον Πίνακα 4.1 φαίνονται τα προβλήματα που λύθηκαν από το UCI.

Πίνακας 4.1: Προβλήματα UCI

Όνομα	**Τύπος**	**Διαστάσεις**
abalone	Κατηγοριοποίηση	4178x9
balanceScale	Κατηγοριοποίηση	626x5
bestSellerBooks	Κατηγοριοποίηση	551x7
bloodTransfusion	Κατηγοριοποίηση	749x5
cars	Κατηγοριοποίηση	1729x7
drugTypes	Κατηγοριοποίηση	201x6
haberman	Κατηγοριοποίηση	307x4
hayesRoth	Κατηγοριοποίηση	133x6
lenses	Κατηγοριοποίηση	25x5
ticTacToe	Κατηγοριοποίηση	959x10
advertising	Παλινδρόμηση	201x4
aquaticToxicity	Παλινδρόμηση	547x9
carPriceAssignment	Παλινδρόμηση	206x26
diabetes	Παλινδρόμηση	769x9
drivePoints	Παλινδρόμηση	607x19
insurance	Παλινδρόμηση	1339x7
realEstate	Παλινδρόμηση	415x8
slumpTest	Παλινδρόμηση	104x11
southGermanCredit	Παλινδρόμηση	1001x21
winequalityRed	Παλινδρόμηση	1600x12

4.1 Αποτελέσματα στα Δεδομένα Παλινδρόμησης του UCI

Σε αυτήν την ενότητα θα σχολιάσουμε τα αποτελέσματα των αλγορίθμων που αναφέρθηκαν στην αρχή του κεφαλαίου 4 και θα γίνουν παρατηρήσεις πάνω στα δεδομένα, τις μετρικές τιμές και τους χρόνους που χρειάστηκαν οι αλγόριθμοι για να τρέξουν.

Στον Πίνακα 4.2 βλέπουμε τα αποτελέσματα για τα προβλήματα παλινδρόμησης από το UCI για όλους τους αλγορίθμους που αναφέραμε στην αρχή του Κεφαλαίου 4.

Πίνακας 4.2: Αποτελέσματα για δεδομένα Παλινδρόμησης

Advertising				**aquaticToxicity**			
Αλγόριθμος	**Χρόνος (sec)**	**R2**	**MSE**	**Αλγόριθμος**	**Χρόνος (sec)**	**R2**	**MSE**
RBF	0.005	0.933	1.602	RBF	0.064	-5.247	16.179
Kriging	0.725	0.917	2.120	Kriging	10.799	-0.055	3.010
KPLS	0.225	0.914	2.206	KPLS	1.005	0.417	1.680
KPLSK	0.860	0.914	2.19	KPLSK	11.818	-0.045	2.980
LS	0.001	0.888	2.880	LS	0.001	0.456	1.550
IDW	0.000	0.858	3.658	IDW	0.007	0.365	1.820
QP	0.001	0.915	2.173	QP	0.002	0.277	2.063
SVR	0.008	0.469	13.767	SVR	0.026	0.406	1.703
RandomForest	0.055	0.925	2.012	RandomForest	0.175	0.244	2.109
TensorFlow	2.660	0.826	4.545	TensorFlow	3.708	0.423	1.647
Alamo	0.313	0.938	1.782	Alamo	0.203	0.494	1.409
carPriceAssignment				**diabete**			
RBF	0.0286	-13.313	990,510,619.4	RBF	0.149	-30.735	7.492
Kriging	9.880	0.906	6,314,387.3	Kriging	21.597	0.049	0.209
KPLS	0.1953	0.880	8,221,319.5	KPLS	2.043	0.102	0.197
KPLSK	10.110	0.909	6118561.9	KPLSK	23.547	0.059	0.207
LS	0.001	0.840	10331421.4	LS	0.003	0.261	0.163
IDW	0.004	0.750	17653083.8	IDW	0.020	0.163	0.180
QP	-	-	-	QP	0.001	0.195	0.177
SVR	0.008	0.094	63,124,485.8	SVR	0.026	0.051	0.209
RandomForest	0.128	0.892	7,073,597.3	RandomForest	0.208	-0.055	0.238
TensorFlow	2.639	0.746	17,310,289.72	TensorFlow	4.307	0.06	0.212
Alamo	0.824	0.948	3,672,000.0	Alamo	0.219	0.344	0.152
drivePoint				**insurance**			
RBF	0.181	0.988	19.915	RBF	0.431	0.852	21,477,697.5
Kriging	42.265	0.996	6.650	Kriging	27.519	0.738	38,518,999.43
KPLS	1.170	0.998	2.640	KPLS	3.491	0.501	72,284,100.5
KPLSK	44.280	0.996	6.436	KPLSK	34.398	0.743	37,923,671.8
LS	0.003	0.998	2.447	LS	0.001	0.751	36,984,623.9
IDW	0.029	0.966	56.050	IDW	0.045	0.140	128,512,998.6
QP	0.011	-39,967.8	65,992,492.8	QP	-	-	-
SVR	0.026	0.553	742.420	SVR	0.184	0.641	53,544,581.3
RandomForest	0.329	0.997	4.251	RandomForest	0.268	0.702	43,537,188.3
TensorFlow	3.602	0.997	4.275	TensorFlow	5.053	0.748	37,237,702.8
Alamo	0.670	0.999	2.416	Alamo	0.645	0.749	36,770,000.0
slumpTest				**outhGermanCredit**			
RBF	0.004	0.947	2.732	RBF	0.493	-10.330	2.344
Kriging	1.200	0.996	0.159	Kriging	133.399	0.032	0.196
KPLS	0.091	0.984	0.694	KPLS	3.273	0.034	0.196
KPLSK	1.169	0.996	0.159	KPLSK	137.983	0.002	0.203
LS	0.003	0.813	8.114	LS	0.007	0.192	0.164
IDW	0.000	0.545	22.070	IDW	0.064	-0.145	0.233
QP	0.001	0.953	2.158	QP	0.009	-47.890	10.316
SVR	0.004	0.008	48.330	SVR	0.060	-0.055	0.214
RandomForest	0.062	0.772	12.080	RandomForest	0.292	-0.177	0.242
TensorFlow	2.440	0.755	13.718	TensorFlow	4.395	-5.420	1.390
Alamo	0.158	0.946	3.940	Alamo	1.470	0.234	0.163
realEstate				**winequalityRed**			
RBF	0.039	-5,562.05	870,158.64	RBF	0.778	0.371	0.392
Kriging	5.391	0.258	152.770	Kriging	129.440	0.328	0.421
KPLS	0.558	0.571	91.247	KPLS	5.767	0.303	0.440
KPLSK	5.811	0.260	151.008	KPLSK	134.122	0.331	0.420
LS	0.000	0.518	102.775	LS	0.004	0.329	0.423
IDW	0.009	0.555	94.772	IDW	0.148	0.269	0.462
QP	0.000	-69.322	8712.400	QP	0.004	0.320	0.428
SVR	0.012	0.038	200.409	SVR	0.333	0.292	0.447
RandomForest	0.136	0.457	98.484	RandomForest	0.646	0.341	0.435
TensorFlow	3.183	0.579	75.080	TensorFlow	5.562	0.311	0.457
Alamo	0.394	0.693	56.300	Alamo	0.619	0.391	0.402

advertising: όπως παρατηρείται όλοι οι αλγόριθμοι βρίσκουν λύσεις αρκετά γρήγορα και οι δύο μετρικές τιμές $(R2, MSE)$ είναι καλές σε όλους τους αλγορίθμους εκτός του SVR, όπου έχει το χαμηλότερο $R2 = 0.46$ και το υψηλότερο $MSE = 13.76$. Οι αλγόριθμοι είναι λογικό να εκτελούνται γρήγορα καθώς οι διαστάσεις του συγκεκριμένου προβλήματος είναι $[201x4]$. Οι αλγόριθμοι με το καλύτερο $R2 = 0.93$ είναι ο RBF και το Alamo, όσον αφορά το MSE ο RBF έχει το καλύτερο αποτέλεσμα με 1.6.

aquaticToxicity: σε όλους τους αλγορίθμους παρατηρείται μείωση του $R2$ και αύξηση του MSE, το οποίο δεν είναι επιθυμητό. Επίσης το υψηλότερο $R2 = 0.49$ είναι του Alamo με το TensorFlow και τον KPLSK να είναι πολύ κοντά. Χειρότερο $R2 = -5.24$ έχει ο RBF επίσης έχει και το υψηλότερο $MSE = 16.17$. Τέλος οι χρόνοι είναι μεγαλύτεροι διότι το πρόβλημα μας έχει περισσότερες διαστάσεις $[547x9]$.

carPriceAssignment: Οι χρόνοι είναι παρόμοιοι με το προηγούμενο πρόβλημα. Παρατηρούμαι ότι όλοι οι αλγόριθμοι έχουν πολύ υψηλότερο MSE από πριν, το $R2$ σε όλους τους αλγόριθμους είναι κοντά στη μονάδα εκτός του RBF με $R2 = -13.31$ και του SVR με $R2 = 0.09$. Να σημειωθεί ότι ο αλγόριθμος QP δεν μπορούσε να τρέξει στο συγκεκριμένο πρόβλημα λογώ του σφάλματος "Singular Matrix".

diabete: Στο συγκεκριμένο πρόβλημα φαίνεται ότι οι χρόνοι είναι μεγαλύτερη, αφού οι διαστάσεις είναι $[769x9]$. Το $R2$ είναι μικρότερο από τα προηγούμενα προβλήματα είναι εμφανής μιας και το καλύτερο $R2 = 0.344$ από το Alamo και αυτή την φορά το χειρότερο $R2$ το έχει ο RBF. Τέλος το MSE είναι μικρότερο σε όλους τους αλγορίθμους με καλύτερο το Alamo και πολύ κοντά να είναι οι KPLS, LS, QP.

drivePoint: Οι χρόνοι είναι αυξημένοι για ακόμα μια φορά διότι οι διαστάσεις του είναι $[607x19]$, ο πιο χρονοβόρος αλγόριθμος είναι ο KPLSK με $44.28sec$, επίσης παρατηρείται ότι στους περισσότερους αλγορίθμους έχουμε καλό $R2, MSE$. Οι αλγόριθμοι οι οποίοι δεν απέδωσαν είναι οι: QP, SVR. Οι αλγόριθμοι με το καλύτερο $R2$ είναι οι Alamo, TensorFlow, RandomForest, LS, KPLSK, KPLS, Kriging με 0.99. Όσον αφορά το MSE το Alamo έχει το καλύτερο με 2.146, πολύ κοντά είναι οι LS με 2.44 και ο KPLS με 2.64.

insurance: Οι χρόνοι είναι παρόμοιοι με το πρόβλημα diabete με διαστάσεις $[1339x7]$, ο πιο χρονοβόρος αλγόριθμος είναι ο KPLSK με $34.39sec$. Το $R2$ είναι καλό ενώ το MSE είναι πολύ υψηλό. Ο αλγόριθμος με τα καλύτερα αποτελέσματα είναι

ο RBF με $R2 = 0.85, MSE = 21477697.5$. Τέλος ο QP έχει το ίδιο σφάλμα "Singular Matrix" με το πρόβλημα carPriceAssignment.

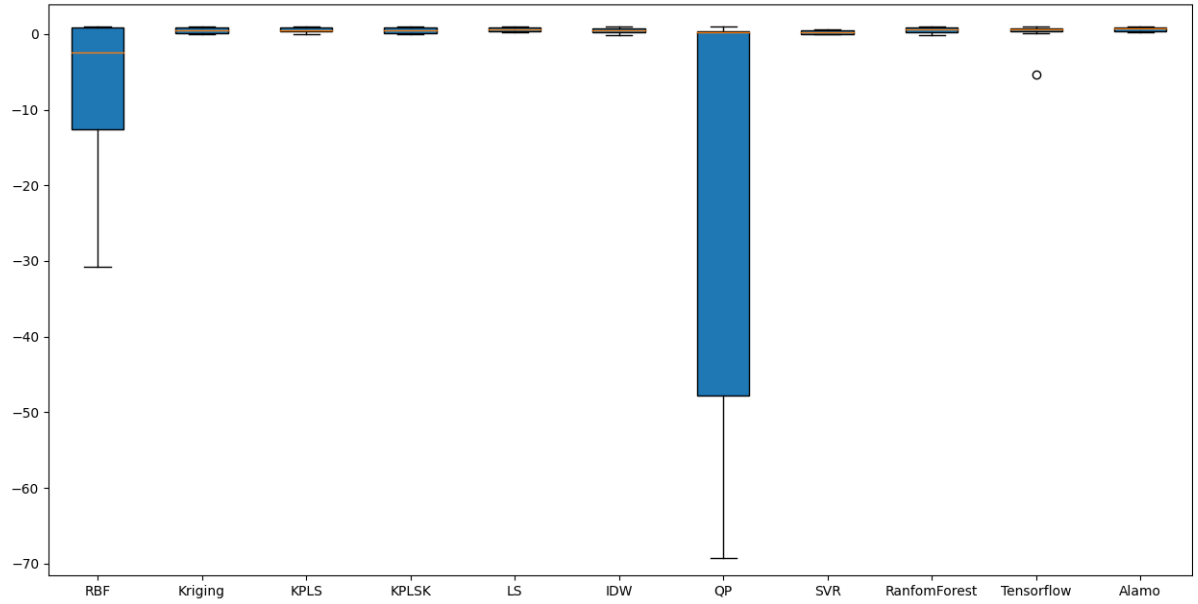
slumpTest: Οι χρόνοι είναι αρκετά μικροί, λογικό διότι οι διαστάσεις είναι $[104x11]$ με τον αγλόριθμο που θέλει τον περισσότερο χρόνο να είναι το TensorFlow με $2.44sec$. Οι μετρικές τιμές είναι καλύτερες από πριν, με χειρότερο $R2 = 0.00822$ έχει ο SVR. Ο Kriging έχει το καλύτερο $R2 = 0.99$ και το καλύτερο $MSE = 0.159$.

outhGermanCredit: Οι αλγόριθμοι χρειάστηκαν περισσότερο χρόνο από κάθε προηγούμενο πρόβλημα. Ο πιο χρονοβόρος αλγόριθμος είναι ο KPLSK με $137.98sec$ και ακολουθεί ο Kriging με $133.39sec$. Οι διαστάσεις του προβλήματος είναι $[1001x21]$. Στο συγκεκριμένο πρόβλημα παρατηρείται ότι το $R2$ είναι κοντά στο μηδέν και σε πολλούς αλγορίθμους (TensorFlow, RandomForest, SVR, QP, RBF) αρνητικό. Δηλαδή χειρότερο από ότι συνήθως αλλά το MSE είναι μικρό και σε πολλούς αλγορίθμους κοντά στο μηδέν. Οι αλγόριθμοι με το καλύτερο $MSE = 0.16$ είναι οι Alamo και LS.

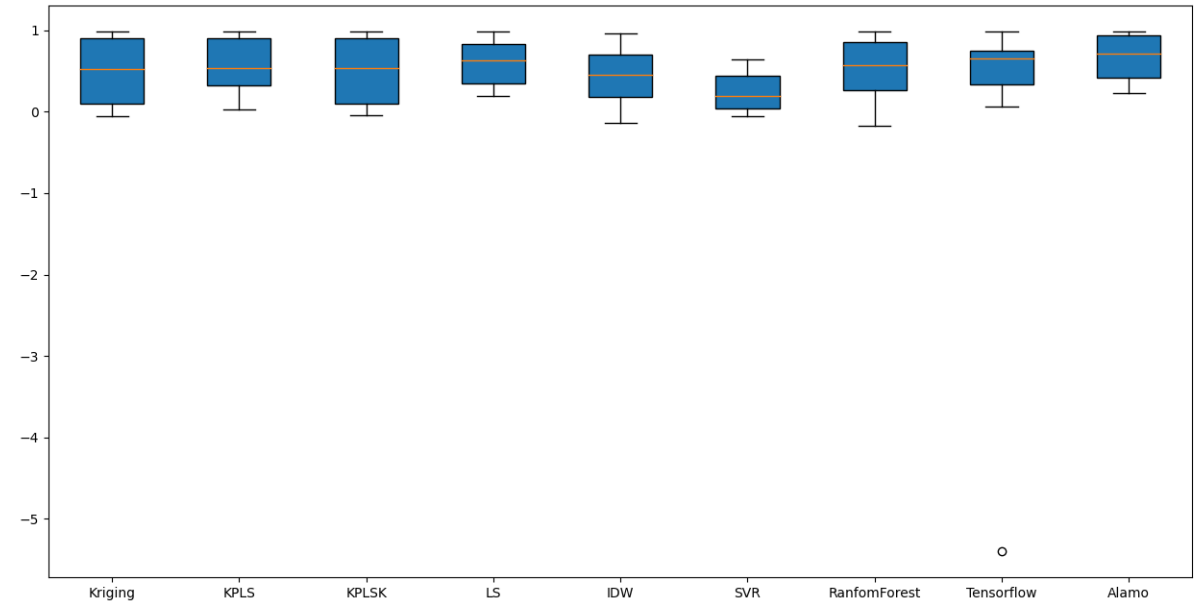
realEstate: Ο αλγόριθμος που χρειάστηκε τον περισσότερο χρόνο είναι ο KPLSK με $5.81sec$ ενώ γενικά το συγκεκριμένο πρόβλημα δεν ήταν χρονοβόρο διότι οι διαστάσεις του δεν είναι πολλές $[415x8]$. Ο RBF στη συγκεκριμένη περίπτωση έχει πολύ κακό $R2, MSE$. Ενώ γενικότερα όλοι οι αλγόριθμοι δεν έκαναν καλές προβλέψεις. Καλύτερο $R2 = 0.69$ και $MSE = 56.3$ έχει το Alamo.

winequalityRed: Το συγκεκριμένο πρόβλημα πήρε περισσότερο χρόνο, λογικό μίας και έχει $[1600x12]$ και είναι το πρόβλημα από το UCI Με τις περισσότερες γραμμές. Ο KPLSK απαιτεί $134.12sec$ και ακολουθεί ο Kriging με $129.4sec$. Το MSE είναι σε όλους τους αγλορίθμους κοντά στο μηδέν με καλύτερο $MSE = 0.39$ έχει ο RBF. Ενώ όσον αφορά το $R2$ όλοι οι αλγόριθμοι είναι κοντά στο 0.3, ο αλγόριθμος με το καλύτερο $R2 = 0.3912$ είναι το Alamo.

Παρακάτω στο Σχήμα 4.1 φαίνονται τα αποτελέσματα των έντεκα αλγορίθμων με βάση το R2 score τους. Το Σχήμα 4.2 είναι το ίδιο με το Σχήμα 4.1 αλλά δεν έχουν συμπεριληφθεί οι αλγόριθμοι RBF, QP, για να έχουμε μια καλύτερη γραφική παράσταση μεταξύ των άλλων εννέα αλγορίθμων.



Σχήμα 4.1: Boxplot για το R2 score στα δέκα προβλήματα παλινδρόμησης



Σχήμα 4.2: Boxplot για το R2 score στα δέκα προβλήματα παλινδρόμησης χωρίς τους QP, RBF

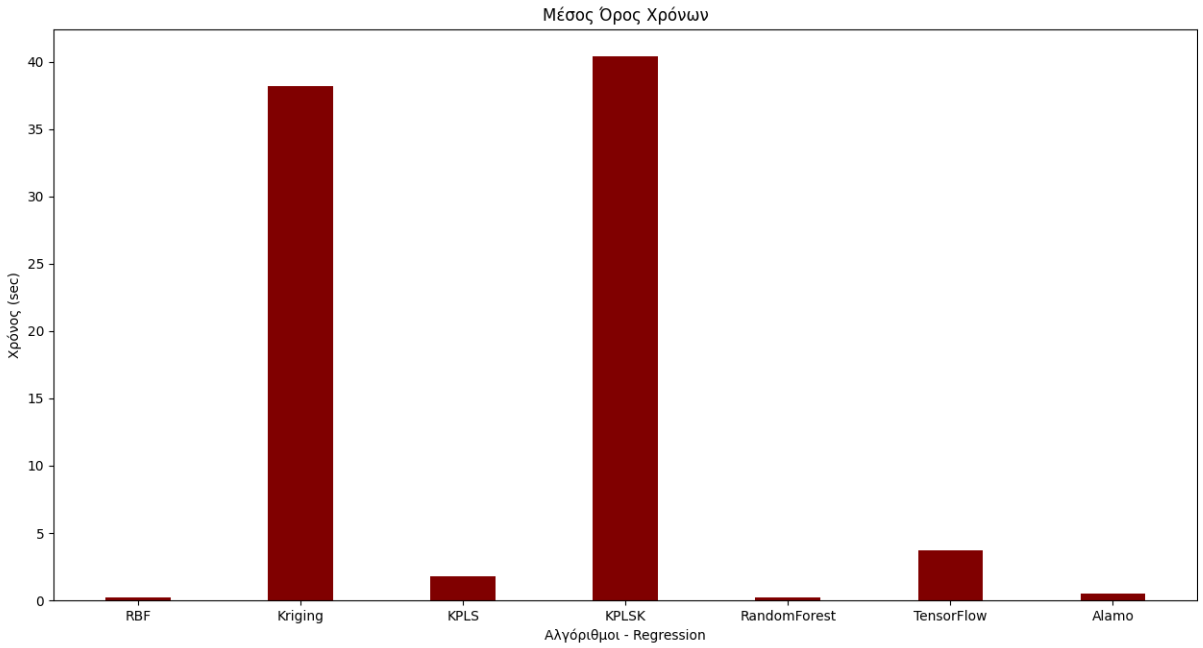
Γενικά σχόλια στα δέκα προβλήματα παλινδρόμησης: Παρατηρούμαι ότι όλοι οι αλγόριθμοι επηρεάζονται περίπου το ίδιο ανά πρόβλημα εκτός μερικών περιπτώσεων όπου έχουμε ακραίες αλλαγές όπως την περίπτωση του RBF στο πρόβλημα

realEstate. Οι πιο χρονοβόροι αλγόριθμοι με μεγάλη διαφορά απ' τους υπόλοιπους είναι οι: Kriging (συνήθως λίγο πιο γρήγορος), KPLSK. Λογικό μιας και στο SMT αναφέρει ότι οι συγκεκριμένοι αλγόριθμοι είναι χρονοβόροι. Συνήθως τα καλύτερα αποτελέσματα μας τα δίνει το Alamo αλλα και ο RandomForest είναι αρκετά κοντά στις περισσότερες περιπτώσεις. Οι αλγόριθμοι LS, IDW, QP είναι εξαιρετικά γρήγοροι και αυτό φαίνεται στον πίνακα διότι ο πιο χρονοβόρος σε αυτά τα δέκα προβλήματα ήταν ο LS με $0.1486sec$ και αν το συγκρίνουμε με τον KPLSK ο οποίος στο ίδιο πρόβλημα έχει $137.98sec$ καταλαβαίνουμε πόσο γρήγοροι είναι. Τέλος το TensorFlow, LS και ο RandomForest μας έδωσαν σχεδόν σε όλα τα προβλήματα καλά αποτελέσματα αλλά σε κανένα δεν μας έδωσε το καλύτερο αποτέλεσμα. Επίσης όπως φαίνεται και στο Σχήμα 4.1, 4.2 οι αλγόριθμοι που δεν μας έδωσαν καλά αποτελέσματα είναι οι: RBF, QP. Σύμφωνα με το SMT ο QP για να αποδώσει πρέπει να έχουμε πολλά δεδομένα, αλλά τα συγκεκριμένα προβλήματα δεν είχαν πολλές διαστάσεις, όσον αφορά τον RBF στα προβλήματα όπου δεν έχει καλά αποτελέσματα είναι σε δεδομένα όπου τα νούμερα είναι κοντά το ένα στο άλλο και όπως το SMT αναφέρει ότι σε δεδομένα όπου τα σημεία είναι κοντά στο ένα με το άλλο δεν κάνει καλές προβλέψεις. Επίσης ο SVR δεν έδωσε σε κανένα πρόβλημα καλό αποτέλεσμα εκτός του προβλήματος insurance που μας έδωσε $R2 = 0.64$ στα υπόλοιπα προβλήματα οι τιμές του ήταν κυρίως κοντά στο 0 αλλά και αρνητικές όπως στο πρόβλημα outhGermanCredit. Όταν το κουτί σε ένα boxplot είναι μεγάλο αυτό σημαίνει ότι οι τιμές διαφέρουν και στη δικιά μας περίπτωση ο QP, RBF φαίνεται να έχουν αρκετές διαφορετικές αρνητικές τιμές. Η πορτοκαλί γραμμή συμβολίζει τη διάμεσο (median) οι μισές βαθμολογίες είναι μεγαλύτερες ή ίσες με αυτήν την τιμή και οι μισές είναι μικρότερες. Όταν ένα κουτί είναι μικρό στην περίπτωση μας έχουμε τον SVR αυτό σημαίνει ότι οι τιμές δεν έχουν μεγάλη διακύμανση. Τέλος ο κύκλος ο οποίος φαίνεται στο TenserFlow στο -5.42 σημαίνει ότι έχουμε ακραία τιμή σε εκείνο το σημείο, το οποίο επαληθεύεται από τον Πίνακα 4.2 στο πρόβλημα outhGermanCredit. Τέλος όπως ήταν και λογικό οι αλγόριθμοι που είναι βασισμένοι στον Krigingδηλαδή οι Kriging, KPLS, KPLSK μας έδωσαν πολύ παρόμοια αποτελέσματα και τα boxplot τους είναι παρόμοια. Οι αλγόριθμοι με τη καλύτερη διάμεσο βάση των Σχημάτων 4.1, 4.2 είναι:

1. Alamo ≈ 0.7

2. TensorFlow ≈ 0.65
3. LS ≈ 0.63
4. RandomForest ≈ 0.58
5. Kriging, KPLS, KPLSK ≈ 0.52
6. IDW ≈ 0.450
7. QP, SVR ≈ 0.19
8. RBF ≈ -2.4

Παρακάτω στο Σχήμα 4.3 φαίνεται ο μέσος όρος για τους χρόνους για κάθε αλγόριθμο, στο σχήμα δεν συμπεριέλαβα τους αλγόριθμους SVR, QP, LS, IDW διότι οι χρόνοι τους ήταν πάρα πολύ κοντά στο μηδέν.



Σχήμα 4.3: Μέσος όρος χρόνων στα Προβλήματα παλινδρόμησης του UCI

4.2 Αποτελέσματα στα δεδομένα Κατηγοριοποίησης του UCI

Παρακάτω στον Πίνακα 4.3 βλέπουμε τα αποτελέσματα για τα προβλήματα κατηγοριοποίησης από το UCI για όλους τους αλγορίθμους που αναφέραμε στην αρχή του Κεφαλαίου 4.

Πίνακας 4.3: Αποτελέσματα για δεδομένα Κατηγοριοποίησης

abalone						**balanceScale**					
Αλγόριθμος	Χρόνος (sec)	accuracy	f1	precision	recall	Αλγόριθμος	Χρόνος (sec)	accuracy	f1	precision	recall
RBF	4.452	0.111	0.236	0.238	0.236	RBF	0.052	0.239	0.281	0.355	0.281
Kriging	274.909	0.093	0.195	0.197	0.195	Kriging	6.190	0.251	0.300	0.413	0.300
KPLS	24.840	0.028	0.107	0.147	0.1072	KPLS	1.299	0.245	0.296	0.400	0.290
KPLSK	351.360	0.092	0.196	0.196	0.196	KPLSK	7.506	0.247	0.298	0.410	0.298
LS	0.043	0.109	0.219	0.223	0.219	LS	0.001	0.169	0.240	0.250	0.240
IDW	0.734	0.103	0.236	0.233	0.236	IDW	0.008	0.008	0.011	0.006	0.011
QP	0.006	0.143	0.251	0.260	0.251	QP	0.008	0.272	0.340	0.427	0.3407
SVC	0.750	0.086	0.252	0.089	0.097	SVC	0.0168	0.151	0.189	0.140	0.205
RandomForest	2.346	0.138	0.239	0.152	0.136	RandomForest	0.109	0.057	0.070	0.07	0.066
TensorFlow	7.603	0.000	1.0	1.0	1.0	TensorFlow	1.83	0.203	1.0	1.0	1.0
Alamo	1.013	0.148	0.246	0.257	0.246	Alamo	25.828	0.315	0.319	0.466	0.317
bestSellersBook						**bloodTransfusion**					
RBF	0.064	0.120	0.601	0.848	0.601	RBF	0.099	0.152	0.744	0.757	0.744
Kriging	8.046	0.801	0.817	0.835	0.817	Kriging	4.998	0.500	0.500	0.738	0.714
KPLS	1.012	0.606	0.806	0.815	0.806	KPLS	1.875	0.477	0.738	0.730	0.738
KPLSK	9.0943	0.801	0.817	0.835	0.817	KPLSK	7.148	0.500	0.714	0.738	0.714
LS	0.001	0.604	0.659	0.660	0.659	LS	0.002	0.395	0.759	0.757	0.759
IDW	0.010	0.776	0.788	0.792	0.788	IDW	0.0146	0.586	0.720	0.701	0.722
QP	0.0012	0.574	0.699	0.702	0.699	QP	-	-	-	-	-
SVC	0.008	0.591	0.620	0.609	0.595	SVC	0.013	0.450	0.756	0.612	0.508
RandomForest	0.182	0.837	0.843	0.848	0.834	RandomForest	0.174	0.593	0.742	0.609	0.588
TensorFlow	1.786	0.560	0.718	0.560	1.0	TensorFlow	2.268	0.242	0.389	0.242	1.0
Alamo	453.406	0.679	0.738	0.688	0.677	Alamo	0.512	0.473	0.754	0.630	0.516
cars						**drugType**					
RBF	0.534	0.253	0.563	0.710	0.563	RBF	0.007	0.309	0.454	0.760	0.454
Kriging	46.570	0.516	0.850	0.934	0.850	Kriging	1.212	0.668	0.837	0.902	0.837
KPLS	3.748	0.2402	0.683	0.856	0.683	KPLS	0.179	0.503	0.690	0.831	0.690
KPLSK	55.382	0.529	0.881	0.942	0.880	KPLSK	1.355	0.660	0.835	0.903	0.835
LS	0.006	0.209	0.491	0.548	0.491	LS	0.000	0.189	0.272	0.773	0.272
IDW	0.073	0.267	0.673	0.536	0.673	IDW	0.001	0.091	0.1	0.332	0.1
QP	0.001	0.256	0.548	0.816	0.548	QP	-	-	-	-	-
SVC	0.062	0.784	0.915	0.88	0.760	SVC	0.0024	0.3366	0.722	0.300	0.406
RandomForest	0.202	0.962	0.978	0.964	0.961	RandomForest	0.096	0.996	0.994	0.996	0.996
TensorFlow	3.971	0.039	0.877	0.781	1.0	TensorFlow	1.032	0.116	0.707	0.547	1.0
Alamo	0.813	0.095	0.235	0.058	0.250	Alamo	347.628	0.130	0.465	0.096	0.205
haberman						**hayesRoth**					
RBF	0.078	0.575	0.750	0.730	0.750	RBF	0.007	0.586	0.600	0.702	0.629
Kriging	1.354	0.611	0.745	0.730	0.745	Kriging	0.821	0.431	0.529	0.576	0.529
KPLS	0.371	0.575	0.760	0.765	0.761	KPLS	0.147	0.584	0.618	0.672	0.618
KPLSK	1.571	0.591	0.743	0.730	0.743	KPLSK	0.913	0.435	0.525	0.604	0.525
LS	0.000	0.502	0.741	0.759	0.741	LS	0.000	0.664	0.655	0.712	0.655
IDW	0.002	0.467	0.732	0.639	0.732	IDW	0.001	0.300	0.45	0.405	0.451
QP	0.000	0.588	0.761	0.765	0.761	QP	0.001	0.586	0.600	0.656	0.600
SVC	0.002	0.431	0.720	0.390	0.500	SVC	0.001	0.250	0.362	0.240	0.336
RandomForest	0.099	0.511	0.633	0.526	0.521607	RandomForest	0.093	0.844	0.818	0.847	0.846
TensorFlow	1.567	0.742	1.0	1.0	1.0	TensorFlow	1.365	0.391	1.0	1.0	1.0
Alamo	0.272	0.660	0.780	0.755	0.648	Alamo	0.207	0.224	0.362	0.242	0.297
lense						**ticTacToe**					
RBF	0.001	0.420	0.610	0.766	0.619	RBF	0.223	0.637	0.850	0.864	0.855
Kriging	0.143	0.540	0.740	0.744	0.740	Kriging	38.520	0.651	0.976	0.984	0.9760
KPLS	0.042	0.510	0.720	0.773	0.720	KPLS	3.201	0.912	0.955	0.953	0.955
KPLSK	0.209	0.544	0.740	0.744	0.740	KPLSK	43.801	0.700	0.976	0.983	0.976
LS	0.001	0.540	0.790	0.833	0.739	LS	0.003	0.565	0.690	0.684	0.690
IDW	0.000	0.334	0.500	0.691	0.500	IDW	0.030	0.396	0.658	0.433	0.658
QP	0.000	0.400	0.290	0.600	0.400	QP	0.003	0.983	0.985	0.985	0.985
SVC	0.001	0.432	0.720	0.403	0.4816	SVC	0.027	0.862	0.886	0.916	0.839
RandomForest	0.076	0.677	0.739	0.690	0.705	RandomForest	0.194	0.957	0.962	0.971	0.947
TensorFlow	1.292	0.142	1.0	1.0	1.0	TensorFlow	2.453	0.654	0.791	0.654	1.0
Alamo	-	-	-	-	-	Alamo	0.385	0.983	0.985	0.989	0.970

Με βάση τον Πίνακα 4.3 θα σχολιάσουμε τα αποτελέσματα του κάθε αλγορίθμου για τα δεδομένα κατηγοριοποίησης από το UCI και θα γίνουν παρατηρήσεις πάνω στα δεδομένα, τις μετρικές τιμές και τους χρόνους που χρειάστηκαν οι αλγόριθμοι για να τρέξουν.

abalone: όπως και στα προβλήματα παλινδρόμησης οι αλγόριθμοι που χρειάστηκαν τον περισσότερο χρόνο είναι οι: Kriging $274sec$, KPLSK $351sec$. Οι αλγόριθμοι με το καλύτερο accuracy score είναι οι: Alamo και ο QP με $accuracy = 0.14$. Επίσης ο RandomForest είναι πολύ κοντά με $accuracy = 0.13$. Όμως τα καλύτερα f1 score, precision score, recall score τα έχει το Tensorflow με το καλύτερο score, δηλαδή τη μονάδα ενώ το accuracy score του είναι κοντά στο μηδέν, δηλαδή δεν είναι καλό.

balanceScale: το συγκεκριμένο πρόβλημα δεν ήταν χρονοβόρο, αφού οι διαστάσεις του είναι $[5x626]$ και ο αλγόριθμος που χρειάστηκε τον περισσότερο χρόνο είναι το Alamo με $25.82sec$ και δεύτερος είναι ο KPLSK με $7.5sec$. Ο αλγόριθμος με το καλύτερο accuracy score είναι το Alamo με 0.31 και το δεύτερο καλύτερο accuracy score το έχει ο QP με 0.27, ενώ δυο αλγόριθμοι: RandomForest και ο IDW έχουν accuracy κοντά στο μηδέν το οποίο δεν είναι επιθυμητό. Το TensorFlow έχει τα καλύτερα αποτελέσματα σε f1 score, precision score, recall score με μονάδα, δηλαδή το καλύτερο δυνατό αποτέλεσμα.

bestSellersBook: Στο συγκεκριμένο πρόβλημα όλοι οι αλγόριθμοι έτρεξαν σε λιγότερο από $10sec$ αλλά το Alamo χρειάστηκε $453.4sec$ το οποίο είναι πολύ μεγαλύτερο συγκριτικά με τους άλλους αλγορίθμους και αν έχουμε υπόψιν μας ότι το συγκεκριμένο πρόβλημα έχει μικρές διαστάσεις $[7x551]$ ο χρόνος είναι πολύ μεγάλος. Στο συγκεκριμένο πρόβλημα όλοι οι αλγόριθμοι είχαν υψηλότερο accuracy score με το υψηλότερο να το έχει ο RandomForest με 0.83 και μετά να ακολουθουν οι Kriging, KPLSK με 0.8. Μόνο ο RBF δεν είχε καλό accuracy συγκεκριμένα έχει το χειρότερο με 0.12. Για το recall score το TensorFlow είχε το καλύτερο δυνατό score με μονάδα και ο αλγόριθμος με το δεύτερο καλύτερο recall score είναι ο RandomForest με 0.83. Στο συγκεκριμένο πρόβλημα ενώ το TensorFlow έχει καλύτερο accuracy score οι τιμές του στο f1 score και Precision score είναι χαμηλότερες για την ακρίβεια έχει: $f1score = 0.71, precisionscore = 0.56$. Επίσης το precision score του είναι το χαμηλότερο απ' όλους τους άλλους αλγόριθμους. Το υψηλότερο precision score το έχουν οι RBF, RandomForest με 0.84. Το καλύτερο f1 score το έχει ο RandomForest

με 0.84 αλλά και οι αλγόριθμοι Kriging, KPLS, KPLSK είναι κοντά σε αυτό το score με ≈ 0.8.

bloodTransfusion: στο συγκεκριμένο πρόβλημα όλοι οι αλγόριθμοι βρήκαν αποτελέσματα σε λιγότερο από $10sec$ για την ακρίβεια ο πιο χρονοβόρος αλγόριθμος ήταν ο KPLSK με $7.14sec$. Στο συγκεκριμένο πρόβλημα ο QP δεν βρήκε λύση λόγο του σφάλματος "Singular Matrix" όπως είδαμε και σε κάποια προβλήματα παλινδρόμησης συνέβη το ίδιο. Το καλύτερο accuracy score έχει ο RandomForest με 0.59. Ενώ ο IDW είναι πάρα πολύ κοντά με 0.58 και οι Kriging, KPLSK έχουν 0.5. Επίσης και για αυτό το πρόβλημα ο RBF έχει το χαμηλότερο accuracy score. Ο TensorFlow έχει το χαμηλότερο f1 score με 0.380 ενώ όλοι οι άλλοι αλγόριθμοι έχουν ≈ 0.7. Το ίδιο ισχύει και για το precision score με το TensorFlow να έχει το χειρότερο με 0.24 ενώ όλοι οι άλλοι αλγόριθμοι έχουν από 0.6 εως 0.75 με τον LS να έχει το καλύτερο score. Για ακόμα φορά το καλύτερο recall score έχει το Tensorflow με μονάδα.

cars: λόγω ότι το συγκεκριμένο πρόβλημα έχει περισσότερες διαστάσεις $[1729x7]$ παρατηρήθηκε ότι οι χρόνοι αυξήθηκαν και για την ακρίβεια ο αλγόριθμος που ήταν πιο χρονοβόρος είναι ο KPLSK με $55.38sec$. Ο RandomForest έχει το καλύτερο accuracy score με 0.96 ενώ και οι υπόλοιπες μετρικές του τιμές είναι ≈ 0.96 εκτός του f1 score που έχει 0.97. Το χαμηλότερο accuracy score σημειώθηκε από το Alamo με 0.03. Οι υπόλοιποι αλγόριθμοι είχαν υψηλότερα score με τους RBF, LS, IDW, QP να έχουν ≈ 0.2. Τις υψηλότερες τιμές σε όλα εκτός του recall score τις σημείωσε ο RandomForest με το TensorFlow να έχει το υψηλότερο recall score. Ο αλγόριθμος SVC είχε πολύ καλά αποτελέσματα στο συγκεκριμένο πρόβλημα με $accuracyscore = 0.78, f1score = 0.91, precisionscore = 0.88, recallscore = 0.76$.

drugType: ο QP και για αυτό το πρόβλημα είχε το ίδιο σφάλμα με αποτέλεσμα να μην έχουμε λύση για τον συγκεκριμένο αλγόριθμο. Στο συγκεκριμένο πρόβλημα λόγο των διαστάσεων του οι χρόνοι είναι μικροί εκτός του Alamo όπου χρειάστηκε $347.6sec$, χρόνος ο οποίος είναι πολύ μεγάλος για ένα πρόβλημα διαστάσεων $[201x6]$. Οι μετρικές τιμές για όλους τους αλγορίθμους είναι χαμηλά εκτός του RandomForest ο οποίος σημείωσε τα καλύτερα score με $accuracyscore = 0.99, f1score = 0.99$, $precisionscore = 0.99$, $recallscore = 0.99$. Μόνο το TensorFlow είχε καλύτερο recall score που έχει μονάδα. Το χειρότερο accuracy score το έχει ο IDW με σχεδόν μηδέν για την ακρίβεια έχει 0.09, το μηδέν είναι το χειρότερο αποτέλεσμα. Τέλος πολύ

κακά αποτελέσματα έχει και στις υπόλοιπες μετρικές τιμές του.

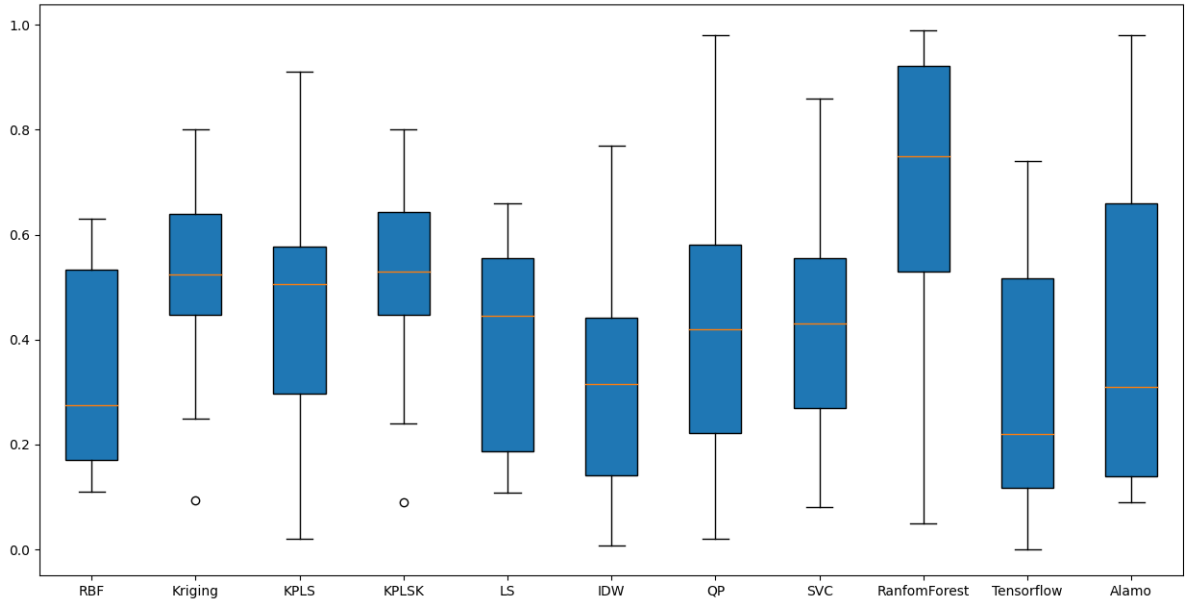
haberman: σε αυτό το πρόβλημα οι χρόνοι λόγο των μικρών διαστάσεων του προβλήματος ήταν μικροί, με τους αλγορίθμους που χρειάστηκαν παραπάνω χρόνο να είναι οι TensorFlow και ο KPLSK ≈ 1.5. Το accuracy score δεν ήταν ιδιαίτερα υψηλό με καλύτερη τιμή να την έχει το TensorFlow με 0.74 ενώ χειρότερη τιμή έχει ο αλγόριθμος SVC με 0.43. Το Tensorflow για τις υπόλοιπες τρεις τιμές έχει μονάδα που σημαίνει ότι έχει το καλύτερο δυνατό score και στο συγκεκριμένο πρόβλημα παρατειρείται ότι και οι άλλοι αλγόριθμοι είχαν τις υπόλοιπες μετρικές τιμές τους.

hayesRoth: όπως και το προηγούμενο πρόβλημα οι χρόνοι ήταν μικροί. Αυτό που αξίζει να σημειωθεί είναι ότι το TensorFlow έχει μονάδα σε τρεις μετρικές τιμές (f1 score, precision score, recall score) στο accuracy score δεν είχε καλό αποτέλεσμα όμως για την ακρίβεια σημείωσε ≈ 0.4. Όμως ο RandomForest σημείωσε το καλύτερο accuracy score με 0.84 ενώ και οι άλλες τρεις του τιμές είναι στο ≈ 0.8.

lense: το συγκεκριμένο πρόβλημα ήταν αρκετά μικρό $[25x5]$ με αποτέλεσμα το Alamo να μην μπορέσει να βρει αποτέλεσμα, το ίδιο πρόβλημα παρατηρήθηκε και σε ορισμένα προβλήματα με δύο στήλες (μια στήλη για τα x και μια για τα y). Λόγο των διαστάσεων του προβλήματος οι αλγόριθμοι έτρεξαν πάρα πολύ γρήγορα. Παρόμοιες τιμές έχουν όλοι οι αλγόριθμοι εκτός του TensorFlow όπου έχει το χαμηλότερο accuracy score με 0.14. Το καλύτερο accuracy score έχει ο RandomForest.

ticTacToe: στο παρών πρόβλημα οι αλγόριθμοι χρειάστηκαν λίγο παραπάνω χρόνο απ' ότι στα περισσότερα προβλήματα, ο αλγόριθμος που χρειάστηκε τον περισσότερο χρόνο ήταν ο KPLSK με $43.8sec$. Στο συγκεκριμένο πρόβλημα παρατηρείται γενική αύξηση στις μετρικές τιμές σε όλους τους αλγορίθμους με καλύτερο accuracy score να είναι του Alamo 0.98 όπως και στις υπόλοιπες τιμές το Alamo έχει ≈ 0.98 κάτι όπου δεν παρατηρήθηκε σε κανένα από τα άλλα προβλήματα κατηγοριοποίησης δηλαδή το Alamo να σημειώνει υψηλές τιμές σε όλες τις μετρικές τιμές. Επίσης ο RandomForest είχε υψηλά αποτελέσματα για την ακρίβεια και στις τέσσερις μετρικές τιμές έχει ≈ 0.95. Τέλος ο QP και στις τέσσερις μετρικές τιμές σημείωσε ≈ 0.98. Το οποίο σημαίνει ότι οι αλγόριθμοι και οι παράμετροι που χρησιμοποίησα ήταν καλή για το συγκεκριμένο πρόβλημα.

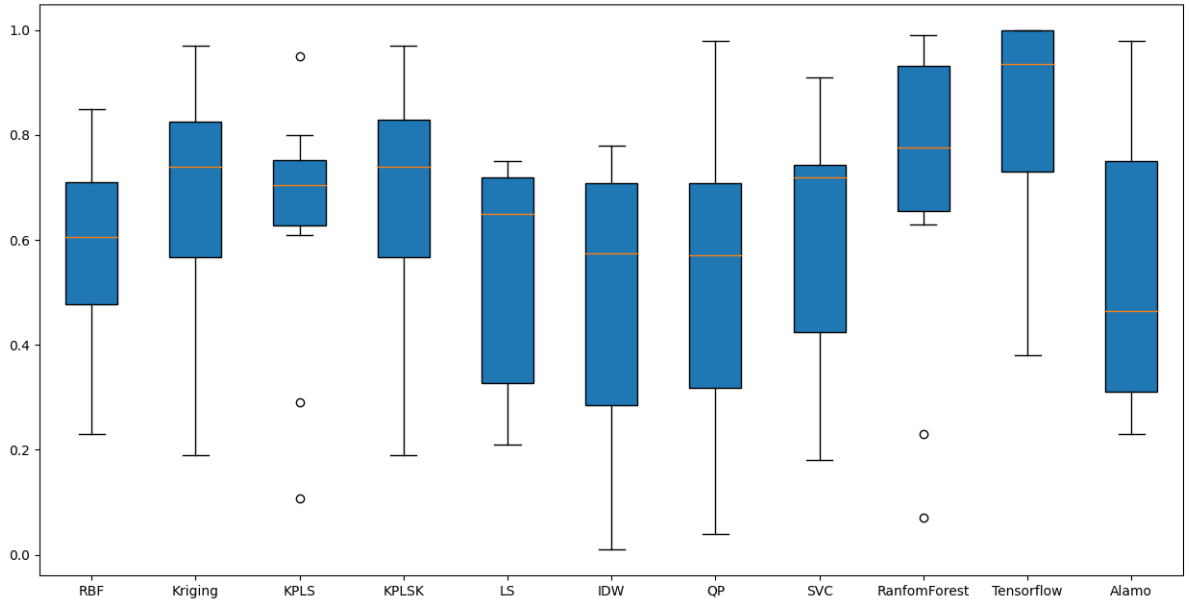
Στο Σχήμα 4.4 φαίνεται το accuracy score σε μορφή παράστασης πλαισίου για όλους τους αλγορίθμους.



Σχήμα 4.4: Accuracy score στα δεδομένα κατηγοριοποίησης UCI

Σχόλια στα δεδομένα κατηγοριοποίησης: όλοι οι αλγόριθμοι επηρεάζονται περίπου το ίδιο ανά πρόβλημα εκτός του TensorFlow το οποίο έβρισκε καλές λύσεις όταν οι άλλοι αλγόριθμοι δεν ήταν κοντά στη λύση και ορισμένες φορές όταν οι άλλοι αλγόριθμοι είχαν καλή βαθμολογία, είχε αρκετά χαμηλότερη βαθμολογία. Επίσης το TensorFlow σε όλα τα προβλήματα έχει το καλύτερο δυνατό recall score δηλαδή τη μονάδα. Στο Σχήμα 4.4 μπορούμε να δούμε με τη μορφή παράστασης πλαισίου το accuracy score για κάθε αλγόριθμο. Εκ πρώτης όψεως φαίνεται ότι ο RandomForest έχει το καλύτερο score με τιμή διαμέσου 0.751. Ο Kriging και ο KPLSK έχουν μια ακράια τιμή στο ≈ 0.09. Επίσης φαίνεται ότι οι αλγόριθμοι RandomForest, Alamo, QP, KPLS έχουν μια υψηλή μέγιστη τιμή με τον RandomForest να είναι πολύ κοντά στη μονάδα. Αλλά οι RBF, Kriging, KPLSK, LS, IDW, SVC, TensorFlow δεν είχαν σε κανένα πρόβλημα πολύ υψηλό accuracy score.

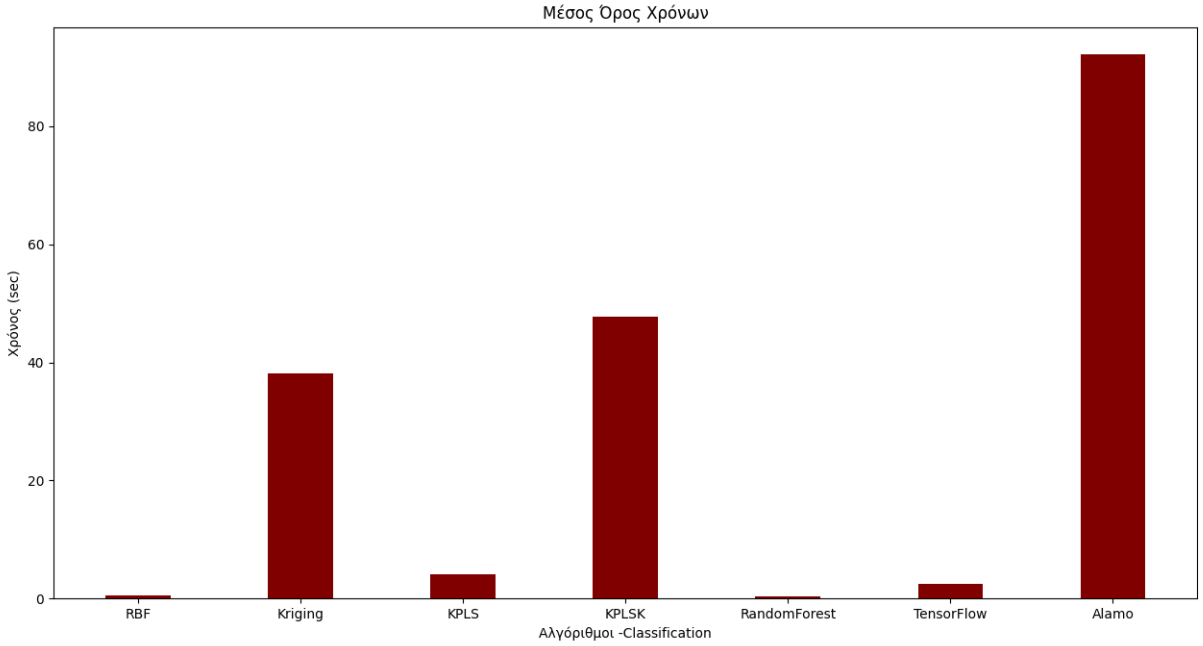
Παρακάτω στο Σχήμα 4.5 βλέπουμε το f1 score των έντεκα αλγόριθμων για τα δεδομένα κατηγοριοποίησης του UCI.



Σχήμα 4.5: F1 score στα δεδομένα κατηγοριοποίησης UCI

Από το Σχήμα 4.5 βλέπουμε ότι το Tensorflow είχε την καλύτερη μέγιστη τιμή με μονάδα και επίσης έχει την καλύτερη τιμή διαμέσου με 0.93 με ελάχιστη τιμή να είναι το 0.37. Ο RandomForest έρχεται δεύτερος με τιμή διαμέσου 0.77 αλλά έχει δύο ακραίες τιμές μια στο 0.2 και η δεύτερη κοντά στο μηδέν. Γενικότερα όλοι οι αλγόριθμοι είχαν διακυμάνσεις στις τιμές το οποίο φαίνεται από τα μπλε κουτιά τα οποία είναι αρκετά «ψηλά». Μόνο ο KPLS δεν έχει μεγάλη διακύμανση σε τιμές αλλά έχει τρεις ακραίες τιμές. Η μια ακραία τιμή είναι μέγιστη ενώ οι άλλες δύο είναι η μια κοντά στο 0.3 ενώ οι άλλη είναι στο 0.1

Στο παρακάτω Σχήμα 4.6 βλέπουμε το μέσο όρο των χρόνων για όλους τους αλγορίθμους εκτός των SVC, QP, LS, IDW διότι ο μέσος όρος τους είναι πολύ κοντά στο μηδέν.



Σχήμα 4.6: Μέσος όρος χρόνων σε sec για τα προβλήματα Κατηγοριοποίησης

Εάν το Σχήμα 4.6 το συγκρίνουμε με το Σχήμα 4.3 βλέπουμε ότι το Alamo είναι πολύ πιο γρήγορο στα δεδομένα παλινδρόμησης για την ακρίβεια το Alamo χρειάστηκε $0.5sec$ για τα δεδομένα παλινδρόμησης ενώ για τα δεδομένα κατηγοριοποίησης χρειάστηκε $92.2sec$ κατά μέσο όρο.

Στον παρακάτω Πίνακα 4.4 βλέπουμε τη σύγκριση των χρόνων για τα προβλήματα παλινδρόμησης, κατηγοριοποίησης από το UCI.

Πίνακας 4.4: Σύγκριση Χρόνων Παλινδρόμησης- Κατηγοριοποίησης

Όνομα	**Χρόνος Παλινδρόμησης**	**Χρόνος Κατηγοριοποίησης**
RBF	0.2	0.5
Kriging	38.2	38.2
KPLS	1.7	4.0
KPLSK	40.4	47.8
RandomForest	0.2	0.3
TensorFlow	3.7	2.5
Alamo	0.5	92.2

Όπως φαίνεται και στον Πίνακα 4.4 σε όλους τους αλγορίθμους παρουσιάζεται αύξηση του χρόνου, εκτός του Kriging που παραμένει ακριβώς ίδιος ο χρόνος. Αυτό συμβαίνει διότι τα περισσότερα προβλήματα κατηγοριοποίησης είναι μεγαλύτερα από αυτά της παλινδρόμησης και αυτό φαίνεται στον Πίνακα 4.1.

4.3 Αποτελέσματα Προβλημάτων minlplib

Σε αυτήν την ενότητα θα παρουσιαστούν τα αποτελέσματα που είχανε οι αλγόριθμοι στα 255 προβλήματα από το minliplib και θα συγκρίνουμε τον κάθε αλγόριθμο με βάση την ακρίβεια τους. Τέλος με την χρήση του λογισμικού Alamo υπολογίσαμε τη γενικευκεμένη συνάρτηση και τη συγκρίναμε με την πραγματική και πόσο κοντά έπεσε σε αυτήν.

Παρακάτω ακολουθεί ο Πίνακας 4.5 με τα αποτελέσματα του Alamo για τα 255 προβλήματα από το minlplib.

Πίνακας 4.5: Αποτελέσματα προβλημάτων minlplib με την χρήση του Alamo

Όνομα Προβλήματος	**Ίδια Συνάρτηση**	**R2**	**MSE**
chenery	Ναι	1.0	4.182e-08
eniplac	Ναι	1.0	2.64e-09
wastewater11m1	Σχεδόν	1.0	0.00308
catmix100	Σχεδόν	1.0	1.9459-09
fct	Σχεδόν	1.0	6.85799e-09
ghg_2veh	Σχεδόν	1.0	1.65199e-10

Πίνακας 4.5: Αποτελέσματα προβλημάτων minlplib με την χρήση του Alamo

Όνομα Προβλήματος	**Ίδια Συνάρτηση**	**R2**	**MSE**
tanksize	Όχι	0.73	9.034e+12
4stufen	Όχι	0.89	0.897e+05
dispatch	Όχι	1.0	7.154e-09
eq6_1	Όχι	0.61	62540000.0
waterno1_01	Όχι	1.0	1.18e-10
watersbp	Σχεδόν	1.0	5.902e-07
waters	Σχεδόν	1.0	5.934e-07
alkyl	Όχι	1.0	0.002421
alkylation	Όχι	0.99	516338.0
chakra	Όχι	1.0	0.0036
chance	Όχι	1.0	5.256e-07
clay0204h	Όχι	1.0	0.0003
clay0303h	Όχι	0.94	1.9676e+9
ex2_1_8	Όχι	1.0	0.001172
kport20	Σχεδόν	1.0	1.924e-09
batchs121208	Όχι	0.84	5.988e+10
bayes2_10	Όχι	1.0	4.0459e-08
trigx	Ναι	1.0	0.000509
abel	Όχι	1.0	1.056
alan	Ναι	0.99	4.02e+10
arki0002	Όχι	0.99	0.413e+12
arki0003	Όχι	1.00	0.481e-07
batch	Όχι	0.99	2.126e+7
batch0812	Όχι	0.96	1.898e+9
batch0812_nc	Όχι	0.99	1.956e+9
batchdes	Όχι	0.99	7.304e+6
batchs10106	Όχι	0.89	1.05e+10
bayes2_20	Όχι	1.0	4.172e-08
bayes2_30	Όχι	1.0	4.118e-08
bayes2_50	Όχι	1.0	4.2539e-08

Πίνακας 4.5: Αποτελέσματα προβλημάτων minlplib με την χρήση του Alamo

Όνομα Προβλήματος	**Ίδια Συνάρτηση**	**R2**	**MSE**
bearing	Ναι	1.0	1.916e-10
beuster	Όχι	-0.66	1.32156e+6
catmix200	Σχεδόν	1.0	1.945e-09
catmix400	Σχεδόν	1.0	1.938e-09
catmix800	Σχεδόν	1.0	1.928e-09
chem	Όχι	1.0	4.426e+4
chp_partload	Όχι	1.0	4.222e-13
clay0203h	Όχι	0.95	1.892e+9
clay0203hfsg	Όχι	0.94	2.0876e+9
clay0203m	Όχι	0.93	2.672e+9
clay0204hfsg	Όχι	1.0	0.0003
clay0204m	Όχι	1.0	0.00031
clay0205h	Όχι	1.0	0.00063
clay0205hfsg	Όχι	1.0	0.0006
clay0205m	Όχι	1.0	0.0006
clay0303hfsg	Όχι	0.93	2.6e+9
clay0303m	Όχι	0.90	3.796e+9
clay0304h	Όχι	1.0	0.0003
clay0304hfsg	Όχι	1.0	0.0003
clay0304m	Όχι	1.0	0.0003
clay0305h	Όχι	1.0	0.0006
clay0305hfsg	Όχι	1.0	0.0006
clay0305m	Όχι	1.0	0.0006
demo7	Όχι	1.0	0.002
etamac	Σχεδόν	1.0	1.274e-12
ex2_1_2	Όχι	1.0	1.522e-07
ex2_1_1	Σχεδόν	0.96	8.533
ex1223	Όχι	1.0	2.024e-12
ex1224	Όχι	0.98	0.0013
ex1225	Σχεδόν	1.0	1.3e-12

Πίνακας 4.5: Αποτελέσματα προβλημάτων minlplib με την χρήση του Alamo

Όνομα Προβλήματος	**Ίδια Συνάρτηση**	**R2**	**MSE**
ex1226	Όχι	1.0	7.738e-12
ex1252	Όχι	0.7	3.56e+9
ex2_1_3	Όχι	1.0	4.934e-09
ex2_1_4	Όχι	1.0	5.169e-09
ex2_1_5	Όχι	1.0	8.84e-11
ex2_1_6	Όχι	1.0	2.392e-10
ex2_1_7	Όχι	1.0	8.028
ex2_1_9	Όχι	0.99	2.48e+10
ex3_1_1	Όχι	1.0	5.468e-08
ex2_1_10	Όχι	1.0	74.12
ex3_1_2	Όχι	1.0	197.8
ex3_1_3	Όχι	1.0	1.242
ex3_1_4	Ναι	1.0	5.592e-11
ex4_1_4	Σχεδόν	1.0	5.46e-11
ex4_1_8	Όχι	1.0	2.054e-12
ex4_1_9	Όχι	1.0	4.9917e-30
ex5_2_2_case1	Όχι	0.97	2.936e+5
ex5_2_2_case2	Όχι	0.96	6.636e+5
ex5_2_2_case3	Όχι	0.94	7.008e5
ex5_2_4	Όχι	0.98	37520
ex5_3_2	Όχι	1.0	8.682e-14
ex5_3_3	Όχι	0.92	0.419
ex5_4_2	Σχεδόν	1.0	5.34e-08
ex5_4_3	Όχι	0.95	5.624e+4
ex5_4_4	Όχι	0.97	22320
ex6_1_1	Όχι	0.87	5020
ex6_1_2	Όχι	0.79	8738
ex6_1_3	Όχι	0.90	11120.0
ex6_1_4	Όχι	0.88	63340.0
ex6_2_10	Όχι	0.99	0.00042

Πίνακας 4.5: Αποτελέσματα προβλημάτων minlplib με την χρήση του Alamo

Όνομα Προβλήματος	**Ίδια Συνάρτηση**	**R2**	**MSE**
ex6_2_11	Όχι	0.80	0.009
ex6_2_12	Όχι	0.99	0.0001
ex6_2_13	Όχι	0.99	0.00012
ex6_2_14	Όχι	0.96	0.0094
ex6_2_5	Όχι	0.99	2.622
ex6_2_6	Όχι	0.93	0.0043
ex6_2_7	Όχι	0.97	0.00016
ex6_2_8	Όχι	0.81	0.00039
ex6_2_9	Όχι	0.97	0.00012
ex7_2_1	Όχι	0.98	6316.6
ex7_2_3	Ναι	1.0	5.222e-08
ex7_2_4	Όχι	0.94	0.636
ex8_1_1	Όχι	0.95	0.015
ex8_1_6	Όχι	0.41	2.278
ex8_1_7	Όχι	0.46	870800
ex8_3_13	Σχεδόν	1.0	9.986e-08
ex8_4_4	Όχι	1.0	1.63e-13
ex8_4_5	Όχι	1.0	8.204e-14
ex9_1_1	Ναι	1.0	4.012e-08
ex9_1_2	Ναι	1.0	8.262e-09
ex9_1_4	Όχι	1.0	8.576e-09
ex9_1_5	Όχι	1.0	9.438e-09
ex9_1_8	Ναι	1.0	2.896e-09
ex9_2_2	Ναι	1.0	0.000168
ex9_2_3	Σχεδόν	1.0	6.335e-08
ex9_2_4	Ναι	1.0	9.288e-05
ex9_2_5	Ναι	1.0	4.65e-05
ex9_2_6	Όχι	1.0	0.0005
ex9_2_7	Ναι	1.0	0.002
ex9_2_8	Όχι	0.89	29280.0

Πίνακας 4.5: Αποτελέσματα προβλημάτων minlplib με την χρήση του Alamo

Όνομα Προβλήματος	**Ίδια Συνάρτηση**	**R2**	**MSE**
feedtray	Όχι	1.0	7.802e-12
filter	Όχι	1.0	0.4907
fin2bb	Όχι	1.0	1.016e-06
flay02h	Όχι	1.0	8.186e-13
flay02m	Όχι	1.0	8.602e-13
flay03h	Όχι	1.0	8.618e-13
flay03m	Όχι	1.0	8.55e-13
flay04h	Όχι	1.0	8.654e-12
flay04m	Ναι	1.0	8.526e-12
flay05h	Όχι	1.0	8.5119e-13
flay05m	Όχι	1.0	8.35e-13
flay06h	Όχι	1.0	8.6e-13
flay06m	Όχι	1.0	8.456e-13
fo7	Όχι	1.0	3.044e-07
fo7_2	Όχι	1.0	3.243e-07
fo7_ar2_1	Όχι	1.0	3.268e-07
fo7_ar25_1	Όχι	1.0	3.028e-07
forest	Ναι	1.0	4.216e-08
fuel	Όχι	1.0	4.673e-07
gams01	Όχι	0.64	3188000.0
gams02	Όχι	0.90	44000
gasnet	Ναι	1.0	6.943e-09
gastrans	Ναι	1.0	2.7619e-12
gastransnlp	Όχι	1.0	2.786e-12
gear2	Όχι	0.32	30.84
gear3	Όχι	0.31	34.48
gear4	Ναι	1.0	1.724e-10
windfac	Σχεδόν	1.0	0.0005001
ghg_3veh	Ναι	1.0	3.838e-10
haverly	Ναι	1.0	4.564e-10

Πίνακας 4.5: Αποτελέσματα προβλημάτων minlplib με την χρήση του Alamo

Όνομα Προβλήματος	**Ίδια Συνάρτηση**	**R2**	**MSE**
heatexch_trigen	Σχεδόν	1.0	0.00000001
hhfair	Όχι	0.98	1.348e+14
himmel11	Όχι	1.0	173.8
himmel16	Σχεδόν	1.0	4.27e-08
house	Σχεδόν	1.0	7.139e-09
hs62	Όχι	0.85	19320000
hydro	Όχι	1.0	0.003144
immun	Όχι	1.0	197400
johnall	Όχι	1.0	0.014
kport40	Σχεδόν	1.0	1.928e-09
m3	Ναι	1.0	7.504e-07
m6	Όχι	1.0	2.144e-06
m7	Όχι	1.0	5.028e-06
mathopt1	Όχι	0.99	6.716e+8
mathopt3	Όχι	0.54	3.788e+12
mathopt4	Όχι	0.96	1.656e+8
mathopt6	Όχι	0.43	3.922
pindyck	Όχι	1.0	7.396e-08
powerflow0009p	Όχι	1.0	819.6
p_ball_10b_5p_2d_h	Όχι	1.0	2.654e-12
p_ball_10b_5p_2d_m	Όχι	1.0	2.662e-12
powerflow0009r	Όχι	1.0	824.6
powerflow0014p	Όχι	1.0	1614
powerflow0014r	Όχι	1.0	1626.0
powerflow0030p	Όχι	0.72	1.41e+16
powerflow0030r	Όχι	0.81	9.848e+15
powerflow0039p	Όχι	1.0	330.2
powerflow0039r	Όχι	1.0	334.8
primary	Όχι	0.97	24.54
prob07	Όχι	0.99	3068000

Πίνακας 4.5: Αποτελέσματα προβλημάτων minlplib με την χρήση του Alamo

Όνομα Προβλήματος	**Ίδια Συνάρτηση**	**R2**	**MSE**
prob09	Όχι	0.78	22000.0
ramsey	Όχι	1.0	5.788e-12
rbrock	Όχι	0.97	3.216e+8
risk2bpb	Όχι	1.0	6.6259e-06
sample	Ναι	1.0	6.684e-05
st_bpaf1a	Όχι	0.99	634.4
st_bpaf1b	Όχι	0.98	993.6
st_bpk1	Όχι	0.92	7.76e+9
st_bpv1	Όχι	0.79	862.2
st_bpv2	Όχι	0.97	1.09
st_bsj2	Όχι	1.0	0.0138
st_bsj3	Όχι	0.97	5960000
st_bsj4	Όχι	0.97	8530000
st_cqpf	Ναι	1.0	0.0342
st_cqpjk1	Σχεδόν	1.0	23.00
st_cqpjk2	Ναι	1.0	4.84e-13
st_qpk1	Σχεδόν	0.88	2.624e+10
st_qpk2	Όχι	0.81	9.15e+10
st_qpk3	Όχι	0.99	1.466e+10
st_rv1	Όχι	1.0	8.944e-06
st_rv2	Όχι	1.0	6.046e-07
st_rv3	Όχι	1.0	7.035e-07
st_rv7	Όχι	1.0	2.012e-06
st_rv8	Όχι	1.0	1.432e-06
st_rv9	Όχι	1.0	2.104e-06
st_z	Όχι	1.0	0.00959
t1000	Σχεδόν	1.0	7.882e-06
torsion100	Ναι	1.0	8.096e-14
torsion25	Ναι	1.0	8.368e-14
torsion50	Ναι	1.0	8.518e-14

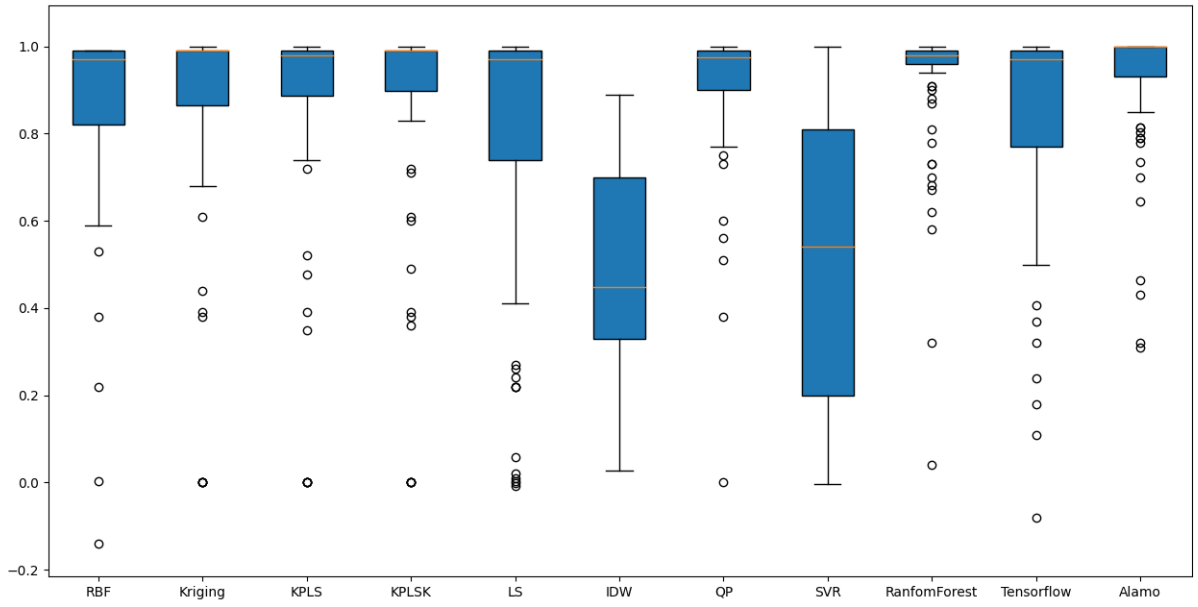
Πίνακας 4.5: Αποτελέσματα προβλημάτων minlplib με την χρήση του Alamo

Όνομα Προβλήματος	**Ίδια Συνάρτηση**	**R2**	**MSE**
torsion75	Ναι	1.0	8.397e-14
trigx	Ναι	1.0	0.0005
wager	Ναι	1.0	9.83e-14
waste	Ναι	1.0	6.91e-09
wastewater02m1	Ναι	1.0	0.0001928
wastewater02m2	Ναι	1.0	0.000192
wastewater04m1	Ναι	1.0	0.0001
wastewater04m2	Ναι	1.0	0.00019
wastewater05m1	Ναι	1.0	0.00037
wastewater05m2	Ναι	1.0	0.0003
wastewater11m1	Σχεδόν	1.0	0.003
wastewater11m2	Σχεδόν	1.0	0.0032
wastewater12m1	Ναι	1.0	0.005
wastewater12m2	Ναι	1.0	0.00522
wastewater13m1	Σχεδόν	1.0	0.0126
wastewater13m2	Σχεδόν	1.0	0.011
wastewater14m1	Ναι	1.0	0.0014
wastewater14m2	Ναι	1.0	0.00146
wastewater15m1	Ναι	1.0	0.00038
wastewater15m2	Ναι	1.0	0.00037
water3	Ναι	1.0	5.89e-07
water4	Ναι	1.0	5.962e-07
waterful2	Ναι	1.0	5.914e-07
waterno1_02	Όχι	1.0	0.150e-08
watersym1	Ναι	1.0	5.978e-07
watersym2	Σχεδόν	1.0	5.898e-07
watertreatnd_conc	Ναι	1.0	9.504e-10
watertreatnd_flow	Σχεδόν	1.0	1.0488e-09
waterund01	Σχεδόν	1.0	8.696e-06
waterund08	Σχεδόν	1.0	0.0002

Πίνακας 4.5: Αποτελέσματα προβλημάτων minlplib με την χρήση του Alamo

Όνομα Προβλήματος	Ίδια Συνάρτηση	R2	MSE
waterund11	Σχεδόν	1.0	0.0001
waterund14	Ναι	1.0	0.000139
waterund17	Σχεδόν	1.0	0.000205
waterund18	Σχεδόν	1.0	8.832e-05
waterx	Σχεδόν	1.0	8.633e-07
water	Ναι	1.0	5.986e-07
waterz	Σχεδόν	1.0	5.908e-07
worst	Όχι	1.0	29.28

Παρακάτω στο Σχήμα 4.7 βλέπουμε σε μορφή παράστασης πλαισίου το R2 score των έντεκα αλγορίθμων στα 255 προβλήματα του minplib.



Σχήμα 4.7: Παράσταση Πλαισίου για το R2 score στα προβλήματα minlplib

Με βάση το Σχήμα 4.7 παρατηρούμαι ότι όλοι οι αλγόριθμοι κυμαίνονται κοντά στο 0.9 εκτός του IDW όπου γνωρίζουμε από το Κεφάλαιο 3 ότι δεν έχει καλή ακρίβεια. Επίσης ο αλγόριθμος SVR ενώ έχει και αποτελέσματα κοντά στη μονάδα σε ορισμένα προβλήματα, τα αποτελέσματα που μας έδωσε κυμαίνονται σε μεγάλο

εύρος αυτό φαίνεται διότι το μπλε «κουτί» του είναι αρκετά ψηλό όπως είναι και του IDW. Η τιμή της διαμέσου του είναι η 0.54. Με μια πρώτη ματιά παρατηρούμαι ότι οι αλγόριθμοι όπου απέδωσαν καλύτερα από τους υπόλοιπους είναι οι Kriging, KPLSK, Alamo. Αν γίνει μεγέθυνση κοντά στη μονάδα βλέπουμε ότι το Alamo έχει το 50% των τιμών του στο εύρος $0.99 - 1$ ενώ ο Kriging με τον KPLSK δεν είχαν ποτέ μονάδα και δεν έχουν τόσο υψηλό ποσοστό στο να προσεγγίζουν τόσο κοντά το καλύτερο δυνατό αποτέλεσμα. Όμως ο Kriging είχε την πιο ακραία αρνητική τιμή με $-6.17e + 05$ το συγκεκριμένο αποτέλεσμα παρουσιάστηκε μεμονωμένα δηλαδή μόνο σε ένα πρόβλημα υπήρχε τόσο κακό R2. Επιπλέον ο KPLSK είχε δύο κακές περιπτώσεις με αρνητικό R2. Στη συνέχεια φαίνεται ότι ο RandomForest είναι λίγο καλύτερος από τους QP, KPLS αλλά όλοι τους φαίνεται να έχουν αρκετές ακραίες τιμές. Όμως ο RandomForest ήταν πιο σταθερός στα αποτελέσματα του και για αυτό το κουτί του φαίνεται να είναι πιο μικρό απ' των άλλων δύο. Στα προηγούμενα προβλήματα παλινδρόμησης του UCI ο QP δεν είχε αποδώσει καλά διότι τα περισσότερα προβλήματα ήταν μικρών διαστάσεων και όπως φαίνεται στα συγκεκριμένα προβλήματα όπου είναι αρκετά μεγαλύτερα ο QP μας έδωσε πολύ καλά αποτελέσματα. Έπειτα το TensorFlow με το LS φαίνεται να μοιάζουν γραφικά όπου και η τιμή της διαμέσου είναι στο 0.97 αλλά το κουτί του TensorFlow είναι πιο μικρό από ότι του LS. Αυτό σημαίνει ότι το TensorFlow είχε συνολικά R2 πιο ψηλά από το LS. Αλλά όπως φαίνεται και γραφικά και οι δύο αλγόριθμοι έχουν πολλές ακραίες τιμές κοντά στο μηδέν κάτι το οποίο δεν είναι επιθυμητό αποτέλεσμα. Τέλος φαίνεται ότι RBF έχει τιμή διαμέσου στο 0.97 δηλαδή περισσότερες από τις μισές του τιμές είναι πάνω από αυτό το νούμερο ενώ οι υπόλοιπες μισές είναι από κάτω. Επίσης το 25% των τιμών του κυμαίνονται από $0.59 - 0.80$. Το αρνητικό του συγκεκριμένου αλγορίθμου είναι ότι έχει πολλές ακραίες τιμές και σε ορισμένα προβλήματα ήταν και αρνητικές.

Παρατηρήσεις ανάμεσα στα προβλήματα παλινδρόμησης UCI-minlplib: όπως φαίνεται ξεκάθαρα και σχηματικά από τα Σχήματα 4.1, 4.2, 4.7, οι αλγόριθμοι μας είχαν καλύτερα αποτελέσματα στα προβλήματα από το minlplib. Αυτό μπορεί να οφείλεται στις διαστάσεις των προβλημάτων όπου τα περισσότερα προβλήματα του UCI ήταν αρκετά μικρότερα από αυτά του minlplib και οι αλγόριθμοι μας είχαν περισσότερα δεδομένα και σημεία εκπαίδευσης. Επίσης αλγόριθμοι όπως ο QP

χρειάζονται πολλές εισόδους όπως αναφέρθηκε και στο Κεφάλαιο 3. Μόνο ο αλγόριθμος IDW είχε παρόμοια αποτελέσματα όπου και στις δύο περιπτώσεις σημείωσε διάμεσο τιμή 0.45 αλλά το SMT αναφέρει ότι ο συγκεκριμένος αλγόριθμος δεν έχει καλή ακρίβεια, όλοι οι άλλοι αλγόριθμοι είχαν αρκετά υψηλότερη τιμή. Επιπλέον ο SVR δεν είχε πολύ υψηλή διάμεσο και η γραφική παράσταση πλαισίου του είναι αρκετά ψηλά συγκριτικά με τους άλλους αλγορίθμους. Τέλος, εκτός του καλύτερου διάμεσου οι αλγόριθμοι έχουν μικρότερα κουτιά όπου όπως έχει αναφερθεί και πιο πάνω σημαίνει ότι περισσότερες τιμές έπεσαν κοντά και ειδικά στους αλγορίθμους με υψηλή διάμεσο το κουτί είναι κοντά στη μονάδα κάτι το οποίο είναι πολύ καλό για τους αλγόριθμους.

Παρακάτω ακολουθεί ο Πίνακας 4.6 όπου φαίνονται οι διαφορές ανά αλγόριθμο με βάση τις τιμές των διαμέσων τους. Παρατηρούμαι ότι οι περισσότεροι αλγόριθμοι έχουν πολύ καλύτερη τιμή στα προβλήματα του minlplib.

Πίνακας 4.6: Σύγκριση Διάμεσων τιμών minlplib - UCI

Όνομα	**Διάμεσος Τιμή minlplib**	**Διάμεσος τιμή UCI**
RBF	0.97	–2.4
Kriging	0.99	0.52
KPLS	0.98	0.52
KPLSK	0.99	0.52
LS	0.97	0.63
IDW	0.45	0.45
QP	0.97	0.19
SVR	0.54	0.19
RandomForest	0.98	0.58
TensorFlow	0.97	0.65
Alamo	≈ 1	0.7

4.3.1 Σύγκριση Συναρτήσεων με την χρήση του Alamo

Παρακάτω στον Πίνακα 4.7 βλέπουμε σε μορφή ποσοστών τα αποτελέσματα του Alamo στο να βρίσκει την πραγματική συνάρτηση.

Πίνακας 4.7: Αποτελέσματα Συναρτήσεων Προβλημάτων minlplib

	Ίδια Συνάρτηση	**Συνάρτηση Κοντά στην Πραγματική**	**Διαφορετική Συνάρτηση**
Ποσοστό (%)	19.6	14.1	66.2

Σχολιασμός σε περίπτωση ίδιας συνάρτησης: Όπως φαίνεται και από τον Πίνακα 4.7 το ποσοστό το να βρίσκει την πραγματική συνάρτηση το λογισμικό Alamo στα προβλήματα μας είναι $\approx 19.6\%$. Συγκεκριμένα στο πρόβλημα chenery με διαστάσεις (2501, 5) η πραγματική συνάρτηση είναι:

$$z_{real} = -x9 - x10 - x11 - x12$$

, ενώ η πρόβλεψη του Alamo είναι:

$$z_{alamo} = -x1 - x2 - 0.99 * x3 - 0.99 * x4$$

Παρατηρούμαι ότι η τάξη των x είναι ίδια όσο και τα πρόσημα, οι μόνες αλλαγές είναι οι σταθεροί συντελεστές. Αυτή η διαφορά προκύπτει στο ότι οι τιμές που γίνονται πρόβλεψη είναι πολύ κοντά αλλά όχι ίδιες με τις πραγματικές τιμές. Οι μετρικές τιμές του συγκεκριμένου προβλήματος είναι: $R2 = 1.0, MSE = 4.182$. Για να δω πόσο ακριβή είναι η συνάρτηση που προέβλεψε το Alamo με την πραγματική τιμή τοποθέτησα τις τιμές του x_test ενώ ήξερα το y_test το αποτέλεσμα ήταν το εξής: $y\_test = -7.79e + 03, y_{alamo\_predict} = -7.79e + 03$. Όπως φαίνεται και από τον υπολογισμό της συνάρτησης η λύση είναι πάρα πολύ κοντά με την πραγματική τιμή. Τέλος και στα υπόλοιπα προβλήματα όπου οι συναρτήσεις ήταν «ίδιες» παρατήρησα παρόμοια αποτελέσματα. Για τον υπολογισμό χρησιμοποίησα το λογισμικό Matlab [31].

Περίπτωση συνάρτησης όπου είναι κοντά στην πραγματική αλλά όχι ίδια: Στη συγκεκριμένη κατηγορία έβαλα τα προβλήματα στα οποία η πρόβλεψη της συνάρτησης είναι πολύ κοντά στην πραγματική αλλά συνήθως προσθέτει κάποιο παραπάνω όρο, όπως $sin(x1)$ ή $ln(x1)$. Στον Πίνακα 4.5 αναφέρω την κατηγορία ως ''Σχεδόν''. Ένα τέτοιο παράδειγμα μπορούμε να εντοπίσουμε στο πρόβλημα

wastewater11m1 όπου η πραγματική συνάρτηση είναι:

$$z_{real} = -(-x112 - x113 - x114 - x115 - x116 - x117 - x118)$$

, ενώ η πρόβλεψη είναι:

$$z_{alamo} = x1 + x2 + 0.99 * x3 + 0.99 * x4 + x5 + x6 + x7 - sin(x6)$$

Όπως παρατηρείται ο επιπλέον όρος είναι στο τέλος το $sin(x6)$. Όσον αφορά τους σταθερούς συντελεστές υπάρχει η ίδια εξήγηση όπως και στην παραπάνω κατηγορία. Τα αποτελέσματα που μου έδωσε η συγκεκριμένη συνάρτηση είναι πάρα πολύ κοντά στην πραγματική τιμή, $y_{test} = 1.923452e + 05, y_{predicted} = 1.923452e + 05$. Ένα ακόμα παράδειγμα όπου το Alamo προβλέπει τη συνάρτηση κοντά στην πραγματική είναι στα προβλήματα catmix, συγκεκριμένα στο πρόβλημα catmix100 έχουμε:

$$z_{real} = -(-x202 - x303 + 1),$$

$$z_{alamo} = x1 + x2 + sin(x1) - 0.99$$

Όπως φαίνεται από τις δύο συναρτήσεις ο επιπλέον όρος είναι το $sin(x1)$. Τα αποτελέσματα της συνάρτησης φαίνονται στον Πίνακα 4.5.

Περίπτωση όπου η συνάρτηση δεν έχει ομοιότητες

Στις περισσότερες περιπτώσεις το Alamo δεν έπεσε κοντά στην πρόβλεψη της συνάρτησης. Αυτό συνέβη διότι βρήκε άλλη συνάρτηση που να βρίσκει τιμές κοντά στις πραγματικές, αλλά υπήρξαν και περιπτώσεις όπου η συνάρτηση που βρήκε δεν έβρισκε αποτελέσματα κοντά στις πραγματικές τιμές. Μία τέτοια κακιά περίπτωση εντοπίζεται στο πρόβλημα tanksize όπου έχουμε $R2 = 0.7352$ και $MSE = 9034000000000$. Το MSE είναι πάρα πολύ υψηλό καθώς και το $R2$ δεν είναι κοντά στη μονάδα. Η πραγματική συνάρτηση είναι:

$$z_{real} = -\frac{(-x34 * x37 - x35 - x36)}{24.87 * x37}$$

ενώ το Alamo προέβλεψε:

$$z_{alamo} = 10850*x1+19254.9*x2-38.58*x2^2-0.45e-002*x1^3+0.421e-001*x2^3-5146753.65$$

Όπως ξεκάθαρα φαίνεται το Alamo δεν είναι κοντά στην πραγματική συνάρτηση, καθώς έχει προσθέσει δυνάμεις στους συντελεστές όπου δεν υπάρχουν στην πραγματική συνάρτηση, όπως $x1^3, x2^2$. Για να δω κατά εάν η τιμές που προβλέπει η συνάρτηση έκανα το ίδιο με παραπάνω δηλαδή κράτησα το y_test και έλεγξα αν οι τιμές είναι κοντά, όμως παρατήρησα ότι δεν είναι. $y_{test} = 1.431269e+04, y_{predicited} = 3.748275e+06$. Βέβαια υπάρχουν περιπτώσεις όπως το 4stufen, όπου η συνάρτηση δεν είναι όμοια αλλά η συνάρτηση κάνει προβλέψεις κοντά στις πραγματικές τιμές:

$$z_{real} = -(-x145 - x146 - x147 - x148 - x149 - x150 - 3271.22)$$

$$z_{alamo} = 4.88 * x2 - 12.11 * sin(x1) + 13.01 * sin(x2) - 40.43 * sin(x4) - 16.3 * sin(x6)$$
$$-0.318e-002 * x2^2 + 0.2e-002 * x4^2 + 0.25e-005 * x5^3 + 3349.35$$

Παρότι η συνάρτηση δεν είναι η ίδια οι τιμές είναι πάρα πολύ κοντά: $y_{test} = 3.387669e+03, y_{predicted} = 3.385528e+03$, παρατηρούμαι ότι η συνάρτηση δεν είναι η ίδια ή παρόμοια οι τιμές όμως που προβλέπει το μοντέλο μας είναι ικανοποιητικές.

Κεφάλαιο 5

Συμπεράσματα

Στην παρούσα διπλωματική εργασία παρουσιάστηκαν έντεκα μοντέλα υποκατάστατων μοντέλων, όπου μέσα σε αυτά είναι το λογισμικό TensorFlow και Alamo. Η γλώσσα προγραμματισμού που χρησιμοποιήθηκε για την υλοποίηση και για την χρήση των λογισμικών είναι η Python. Για να τρέξει το Alamo μέσω της Python χρησιμοποιήθηκε το IDAES [29]. Οι αλγόριθμοι δοκιμάστηκαν σε δέκα προβλήματα παλινδρόμησης από το UCI και επίσης άλλα δέκα προβλήματα κατηγοριοποίησης επίσης από το UCI [9]. Τέλος χρησιμοποιήθηκαν και άλλα 255 προβλήματα από το minlplib τα οποία ήταν παλινδρόμησης. Για να ελέγξουμε την ακρίβεια κάθε αλγόριθμου χρησιμοποιήθηκαν οι μετρικές τιμές που αναφέρθηκαν στο Κεφάλαιο 4.

Τα βασικά συμπεράσματα τα οποία βγήκαν για τα αποτελέσματα ήταν:

1. Οι αλγόριθμοι κατά πλειοψηφία είχανε πολύ καλύτερα αποτελέσματα στα προβλήματα του minlplib απ' ότι είχανε στα προβλήματα από το UCI. Όπως αναλύθηκε και στο Κεφάλαιο 4 αυτό συμβαίνει λόγο ότι οι αλγόριθμοι μας έχουν περισσότερα σημεία εκπαίδευσης. Χαρακτηριστικό παράδειγμα είναι ο QP ο οποίος χρειάζεται πολλά σημεία εισόδου για να έχει καλή απόδοση ενώ στο UCI ήταν ο χειρότερος αλγόριθμος μαζί με τον RBF, στα προβλήματα του minplib ήταν ένας από τους καλύτερους με πολύ υψηλή ακρίβεια.

2. Ο IDW όπως ήταν αναμενόμενο δεν απέδωσε σε κανένα πρόβλημα καλά και αυτό φαίνεται από την θεωρία στο Κεφάλαιο 3 όπου περιμένουμε να μην έχει υψηλή ακρίβεια.

3. Το Alamo στα δεδομένα κατηγοριοποίησης χρειάστηκε πολύ παραπάνω χρόνο

απ' ότι οι υπόλοιποι αλγόριθμοι ενώ στα δεδομένα παλινδρόμησης ήταν αρκετά γρήγορο.

4. Οι αλγόριθμοι SVM είχανε σταθερή απόδοση και στα δεδομένα παλινδρόμησης και στα δεδομένα κατηγοριοποίησης αλλά δεν είχανε καλά αποτελέσματα όπως αναλύθηκε στο Κεφάλαιο 4.

5. Το TensorFlow είχε καλά αποτελέσματα αλλά σε ορισμένα προβλήματα δεν είχε καλά αποτελέσματα, αυτό μπορεί να προκλήθηκε διότι το δίκτυο να χρειαζόταν περισσότερη εκπαίδευση (όλα τα προβλήματα έτρεξαν με epochs = 100).

6. Το TensorFlow στα δεδομένα κατηγοριοποίησης είχε τα καλύτερα αποτελέσματα από τους άλλους αλγόριθμους, όπου υπήρξαν και περιπτώσεις που είχε το καλύτερο δυνατό αποτέλεσμα σε ορισμένες μετρικές, την μονάδα.

7. Ο Random Forest είχε την καλύτερη ακρίβεια στα προβλήματα κατηγοριοποίησης του UCI και ήταν ο δεύτερος καλύτερος αλγόριθμος σε f1 score μετά το TensorFlow.

8. Για τα προβλήματα παλινδρόμησης το καλύτερο μοντέλο που μπορεί να χρησιμοποιηθεί είναι το Alamo όπου έχουμε και περιπτώσεις που έχει R2 ίσο με την μονάδα και συνήθως το MSE είναι πολύ μικρό και πλησιάζει το μηδέν.

9. Τέλος, είδαμε ότι το Alamo δε βρίσκει συχνά την πραγματική συνάρτηση και ούτε πέφτει κοντά σε αυτήν αλλά η συνάρτηση που προβλέπει μιμείται την πραγματική συνάρτηση.

Βιβλιογραφία

[1] A. L. Samuel, "Machine learning," *The Technology Review*, vol. 62, no. 1, pp. 42–45, 1959.

[2] T. M. Mitchell and T. M. Mitchell, *Machine learning*, vol. 1. McGraw-hill New York, 1997.

[3] S. Badillo, B. Banfai, F. Birzele, I. I. Davydov, L. Hutchinson, T. Kam-Thong, J. Siebourg-Polster, B. Steiert, and J. D. Zhang, "An introduction to machine learning," *Clinical pharmacology & therapeutics*, vol. 107, no. 4, pp. 871–885, 2020.

[4] I. Rish *et al*., "An empirical study of the naive bayes classifier," in *IJCAI 2001 workshop on empirical methods in artificial intelligence*, vol. 3, pp. 41–46, 2001.

[5] A. Singh, N. Thakur, and A. Sharma, "A review of supervised machine learning algorithms," in *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, pp. 1310–1315, Ieee, 2016.

[6] A. Dongare, R. Kharde, A. D. Kachare, *et al*., "Introduction to artificial neural network," *International Journal of Engineering and Innovative Technology (IJEIT)*, vol. 2, no. 1, pp. 189–194, 2012.

[7] D. Berrar, "Cross-validation.," 2019.

[8] A. Jain, H. Patel, L. Nagalapatti, N. Gupta, S. Mehta, S. Guttula, S. Mujumdar, S. Afzal, R. Sharma Mittal, and V. Munigala, "Overview and importance of data quality for machine learning tasks," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 3561–3562, 2020.

[9] "Data for ml." `https://archive.ics.uci.edu/ml/index.php`.

[10] "problems." `https://www.minlplib.org/instances.html/`.

[11] A. Sobester, A. Forrester, and A. Keane, *Engineering design via surrogate modelling: a practical guide*. John Wiley & Sons, 2008.

[12] I. Kalogeris, "Advanced surrogate modeling and machine learning methods in computational stochastic mechanics," 2020.

[13] J. Persson, *Efficient optimization of complex products: a simulation and surrogate model based approach*. PhD thesis, Linköping University Electronic Press, 2015.

[14] A. Afzal, K.-Y. Kim, and J.-w. Seo, "Effects of latin hypercube sampling on surrogate modeling and optimization," *International Journal of Fluid Machinery and Systems*, vol. 10, no. 3, pp. 240–253, 2017.

[15] N. V. Queipo, R. T. Haftka, W. Shyy, T. Goel, R. Vaidyanathan, and P. K. Tucker, "Surrogate-based analysis and optimization," *Progress in aerospace sciences*, vol. 41, no. 1, pp. 1–28, 2005.

[16] "Surrogate model pros-cons." `https://www.soa.org/digital-publishing-platform/emerging-topics/surrogate-models/`.

[17] X. Wu, C. Wang, and T. Kozlowski, "Kriging-based surrogate models for uncertainty quantification and sensitivity analysis," in *Proceedings of the MC-2017, International Conference on Mathematics Computational Methods Applied to Nuclear Science Engineering*, 2017.

[18] "Least-squares approximation." `https://scikit-learn.org/stable/modules/linear_model.html`.

[19] D. Shepard, "A two-dimensional interpolation function for irregularly-spaced data," in *Proceedings of the 1968 23rd ACM national conference*, pp. 517–524, 1968.

[20] V. Vapnik, "The support vector method of function estimation," in *Nonlinear modeling*, pp. 55–85, Springer, 1998.

[21] S. Vishwanathan and M. N. Murty, "Ssvm: a simple svm algorithm," in *Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No. 02CH37290)*, vol. 3, pp. 2393–2398, IEEE, 2002.

[22] J. Platt *et al.*, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," *Advances in large margin classifiers*, vol. 10, no. 3, pp. 61–74, 1999.

[23] J. A. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural processing letters*, vol. 9, no. 3, pp. 293–300, 1999.

[24] J. A. Suykens, J. De Brabanter, L. Lukas, and J. Vandewalle, "Weighted least squares support vector machines: robustness and sparse approximation," *Neurocomputing*, vol. 48, no. 1-4, pp. 85–105, 2002.

[25] T.-H. Lee, A. Ullah, and R. Wang, "Bootstrap aggregating and random forest," in *Macroeconomic forecasting in the era of big data*, pp. 389–429, Springer, 2020.

[26] M. A. Bouhlel, J. T. Hwang, N. Bartoli, R. Lafage, J. Morlier, and J. R. R. A. Martins, "A python surrogate modeling framework with derivatives," *Advances in Engineering Software*, p. 102662, 2019.

[27] "Website of tensorflow." `https://www.tensorflow.org`.

[28] "Infos and website alamo." `https://minlp.com/alamo-modeling-tool`.

[29] "idaes doc." `https://readthedocs.org/projects/idaes-pse/downloads/pdf/1.4.0/`.

[30] “sklearn metrics.” https://scikit-learn.org/stable/modules/model_evaluation.html/.

[31] “Matlab site.” https://www.mathworks.com/.