ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΜΕ ΕΦΑΡΜΟΓΕΣ ΣΤΗ ΒΙΟΙΑΤΡΙΚΗ

Ταξινόμηση Γενετικών Δεδομένων

ΠΑΝΑΓΙΩΤΟΥ ΝΙΚΟΛΑΟΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Υπεύθυνος

Τασουλής Σωτήριος

Επίκουρος Καθηγητής

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΜΕ ΕΦΑΡΜΟΓΕΣ ΣΤΗ ΒΙΟΙΑΤΡΙΚΗ

Ταξινόμηση Γενετικών Δεδομένων

Παναγιώτου Νικόλαος

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Επιβλέπων

Τασουλής Σωτήριος

Επίκουρος Καθηγητής

Λαμία, 2021

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

1. *Δεν παραθέτω κομμάτια βιβλίων ή άρθρων ή εργασιών άλλων αυτολεξεί **χωρίς να τα περικλείω σε εισαγωγικά** και χωρίς να αναφέρω το συγγραφέα, τη χρονολογία, τη σελίδα. Η αυτολεξεί παράθεση χωρίς εισαγωγικά χωρίς αναφορά στην πηγή, είναι λογοκλοπή. Πέραν της αυτολεξεί παράθεσης, λογοκλοπή θεωρείται και η παράφραση εδαφίων από έργα άλλων, συμπεριλαμβανομένων και έργων συμφοιτητών μου, καθώς και η παράθεση στοιχείων που άλλοι συνέλεξαν ή επεξεργάσθηκαν, χωρίς αναφορά στην πηγή. Αναφέρω πάντοτε με πληρότητα την πηγή κάτω από τον πίνακα ή σχέδιο, όπως στα παραθέματα.*
2. *Δέχομαι ότι η αυτολεξεί **παράθεση χωρίς εισαγωγικά**, ακόμα κι αν συνοδεύεται από αναφορά στην πηγή σε κάποιο άλλο σημείο του κειμένου ή στο τέλος του, είναι αντιγραφή. Η αναφορά στην πηγή στο τέλος π.χ. μιας παραγράφου ή μιας σελίδας, δεν δικαιολογεί συρραφή εδαφίων έργου άλλου συγγραφέα, έστω και παραφρασμένων, και παρουσίασή τους ως δική μου εργασία.*
3. *Δέχομαι ότι υπάρχει επίσης περιορισμός στο μέγεθος και στη συχνότητα των παραθεμάτων που μπορώ να εντάξω στην εργασία μου εντός εισαγωγικών. Κάθε μεγάλο παράθεμα (π.χ. σε πίνακα ή πλαίσιο, κλπ), προϋποθέτει ειδικές ρυθμίσεις, και όταν δημοσιεύεται προϋποθέτει την άδεια του συγγραφέα ή του εκδότη. Το ίδιο και οι πίνακες και τα σχέδια*
4. *Δέχομαι όλες τις συνέπειες σε περίπτωση λογοκλοπής ή αντιγραφής.*

Ημερομηνία: ……/…../201

Ο – Η Δηλ.

Παναγιώτου Νικόλαος

(Υπογραφή)

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

Ταξινόμηση Γενετικών Δεδομένων

Παναγιώτου Νικόλαος

Τριμελής Επιτροπή:

Τασουλής Σωτήριος, Επίκουρος Καθηγητής (επιβλέπων)

Μπάγκος Παντελεήμων, Καθηγητής - Κοσμήτορας

Πλαγιανάκος Βασίλειος, Καθηγητής

Περιεχόμενα

Κατάλογος Εικόνων

Περίληψη

Η Τεχνητή Νοημοσύνη είναι η μελέτη της ευφυούς συμπεριφοράς. Ο Απώτερος σκοπός της είναι μια θεωρία ευφυΐας πού περιγράφει την συμπεριφορά που έχουν φυσικά νοήμονα όντα και που οδηγεί στην δημιουργία τεχνητών όντων με ευφυή συμπεριφορά. Τα τελευταία χρόνια η Τεχνητή Νοημοσύνη χρησιμοποιείται σε πληθώρα πληροφοριακών συστημάτων σε πολλούς εμπορικούς και επιστημονικούς τομείς όπως επίσης και στον τομέα της ιατρικής.

Η παρούσα πτυχιακή εργασία επικεντρώνεται στο τομέα της τεχνητής νοημοσύνης και ειδικότερα στους αλγόριθμους που χρησιμοποιούνται για την επίλυση προβλημάτων μηχανικής μάθησης. Στόχος είναι η λήψη απόφασης για το ποιος αλγόριθμος τεχνητής νοημοσύνης είναι πιο αποτελεσματικός σε γονιδιακά δεδομένα. Για την επίτευξη του στόχου αυτού χρησιμοποιείται η βάση δεδομένων ClinVar, από την ιστοσελίδα Kaggle, με πάνω από 60.000 δεδομένα. Η βάση δεδομένων αυτή περιέχει πληροφορίες για μεταλλάξεις στο ανθρώπινο γονιδίωμα. Αυτές οι μεταλλάξεις έχουν ταξινομηθεί από ιατρικά εργαστήρια στις εξής κατηγορίες: καλοήθεις, πιθανών καλοήθεις, αβέβαιη σημασία, πιθανών κακοήθεις και κακοήθεις (benign, likely benign, uncertain significance, likely pathogenic, and pathogenic.) Το γεγονός όμως ότι αυτές οι μεταλλάξεις κατατάσσονται χειροκίνητα μπορεί να οδηγήσει σε αντιφατικές ταξινομήσεις από εργαστήριο σε εργαστήριο άρα και σε προβλήματα ερμηνείας αν η μετάλλαξη έχει αντίκτυπο σε κάποιον ασθενή. Λόγω του μεγάλου όγκου δεδομένων της βάσης που χρησιμοποιήθηκε, ήταν απαραίτητο να αξιοποιηθούν τεχνικές εξόρυξης και ανάλυσης δεδομένων. Μέρος της πτυχιακής εργασίας αποτελεί και η παράθεση πηγαίου κώδικα με σκοπό να γίνει αντιληπτός ο τρόπος εξαγωγής των αποτελεσμάτων.

Λέξεις Κλειδιά

Τεχνητή νοημοσύνη, αλγόριθμοι, επεξεργασία δεδομένων, ταξινόμηση, γενετικά δεδομένα

Abstract

Artificial Intelligence is the study of intelligent behavior. Its ultimate goal is an intelligence theory that describes the behavior of naturally intelligent beings and that leads to the creation of artificial beings with intelligent behavior. In recent years, Artificial Intelligence has been used in a variety of information systems in many commercial and scientific fields as well as in the field of medicine.

The present dissertation focuses on the field of artificial intelligence and in particular on the algorithms used to solve machine learning problems. The goal is to decide which artificial intelligence algorithm is most effective in genetic data. To achieve this goal the ClinVar database is being used, from the Kaggle website, with over 60,000 data. This database contains information about mutations in the human genome. These mutations have been classified by medical laboratories into the following categories: benign, potentially benign, uncertain significance, potentially malignant, and malignant (benign, likely benign, uncertain significance, likely pathogenic, and pathogenic.) However, these mutations may be classified manually which may lead to contradictory classifications from laboratory to laboratory and therefore to problems of interpretation if the mutation has an impact on a patient. Due to the large volume of data, it was necessary to utilize data mining and analysis techniques. Part of the thesis defense is the citation of source code in order to understand how the results are extracted.

Keywords

Artificial intelligence, algorithms, data processing, classification, genetic data

Εισαγωγή

Ο όρος τεχνητή νοημοσύνη (AI) αναφέρεται στον κλάδο της πληροφορικής ο οποίος ασχολείται με τη σχεδίαση και την υλοποίηση υπολογιστικών συστημάτων που μιμούνται στοιχεία της ανθρώπινης συμπεριφοράς τα οποία υπονοούν έστω και στοιχειώδη ευφυΐα: μάθηση, προσαρμοστικότητα, εξαγωγή συμπερασμάτων, επίλυση προβλημάτων και πολλά άλλα. (wikipedia) Ο συγκεκριμένος όρος (AI) επινοήθηκε το 1956, αλλά το (AI) έχει γίνει πιο δημοφιλές σήμερα λόγω του αυξημένου όγκου δεδομένων, των προηγμένων αλγορίθμων και των βελτιώσεων στην ισχύ των υπολογιστών και την αποθήκευση των δεδομένων. Αρχικά, η έρευνα γύρω από το AI επικεντρώθηκε σε θέματα όπως η επίλυση προβλημάτων και οι συμβολικές μέθοδοι. Τη δεκαετία του ’60, το Υπουργείο Άμυνας των ΗΠΑ ενδιαφέρθηκε για αυτόν τον τύπο εργασίας και ξεκίνησε την εκπαίδευση των υπολογιστών στη μίμηση της βασικής ανθρώπινης συλλογιστικής. Για παράδειγμα, η Υπηρεσία Προηγμένων Ερευνητικών Προγραμμάτων Άμυνας (DARPA) ολοκλήρωσε τα προγράμματα χαρτογράφησης δρόμων τη δεκαετία του ’70. επίσης, η DARPA παρήγαγε ευφυείς προσωπικούς βοηθούς το 2003, πολύ πριν η Siri, Alexa και Cortana γίνουν πασίγνωστες. Αυτή η πρώτη εργασία, προετοίμασε το έδαφος για την αυτοματοποίηση και την τυπική συλλογιστική που βλέπουμε στους υπολογιστές σήμερα, συμπεριλαμβανομένων των συστημάτων υποστήριξης λήψης αποφάσεων και των έξυπνων συστημάτων αναζήτησης που μπορούν να σχεδιαστούν ώστε να συμπληρώνουν και να βελτιώνουν τις ανθρώπινες ικανότητες.

Κεφάλαιο 1: Τεχνητή Νοημοσύνη

Τεχνητή νοημοσύνη (ΤΝ) (Artificial Intelligence-AI) είναι η μελέτη και δημιουργία προγραμμάτων ηλεκτρονικών υπολογιστών, που σκοπό έχουν να συμπεριφέρονται " έξυπνα" όπως το ανθρώπινο μυαλό. Ο όρος έξυπνα δεν έχει σαφή ερμηνεία , αλλά για τον χώρο της τεχνητής νοημοσύνης μπορεί να ερμηνεύεται σαν ικανότητα να λύνουν προβλήματα, να μαθαίνουν από προηγούμενες εκτελέσεις, "να καταλαβαίνουν" δύσκολες καταστάσεις και δεδομένα και να αντιλαμβάνονται διαφορές και ομοιότητες μεταξύ καταστάσεων. Ένας γενικός ορισμός είναι ο εξής : ΤΝ είναι ο τομέας της επιστήμης υπολογιστών ο οποίος ασχολείται με τη σχεδίαση και την υλοποίηση υπολογιστικών συστημάτων τα οποία είναι ικανά να μιμηθούν στοιχεία της ανθρώπινης συμπεριφοράς τα οποία υπονοούν έστω και στοιχειώδη ευφυΐα: μάθηση, προσαρμοστικότητα, εξαγωγή συμπερασμάτων, κατανόηση από συμφραζόμενα, επίλυση προβλημάτων κλπ. Ο J. McCarthy όρισε τον τομέα αυτόν ως «επιστήμη και μεθοδολογία της δημιουργίας νοούντων μηχανών». [1]

Η Τεχνητή Νοημοσύνη αποτελεί σημείο τομής μεταξύ πολλών πεδίων όπως: Της επιστήμης υπολογιστών, της ψυχολογίας, της φιλοσοφίας, της νευρολογίας, της γλωσσολογίας και της επιστήμης μηχανικών, με στόχο τη σύνθεση ευφυούς συμπεριφοράς, με στοιχεία συλλογιστικής, μάθησης και προσαρμογής στο περιβάλλον, ενώ συνήθως εφαρμόζεται σε μηχανές ή υπολογιστές ειδικής κατασκευής. Διαιρείται στη συμβολική τεχνητή νοημοσύνη, η οποία επιχειρεί να εξομοιώσει την ανθρώπινη νοημοσύνη αλγοριθμικά χρησιμοποιώντας σύμβολα και λογικούς κανόνες υψηλού επιπέδου και στην υποσυμβολική τεχνητή νοημοσύνη, η οποία προσπαθεί να αναπαράγει την ανθρώπινη ευφυΐα χρησιμοποιώντας στοιχειώδη αριθμητικά μοντέλα που συνθέτουν επαγωγικά νοήμονες συμπεριφορές με τη διαδοχική αυτοοργάνωση απλούστερων δομικών συστατικών («συμπεριφορική τεχνητή νοημοσύνη»), προσομοιώνουν πραγματικές βιολογικές διαδικασίες όπως η εξέλιξη των ειδών και η λειτουργία του εγκεφάλου

«υπολογιστική νοημοσύνη», ή αποτελούν εφαρμογή στατιστικών μεθοδολογιών σε προβλήματα ΤΝ.

Οι Russel και Norvig [2] διακρίνουν τους ορισμούς της ΤΝ σε τέσσερις κατηγορίες, με βάση το αν χαρακτηρίζουν ένα σύστημα ως ευφυές με κριτήριο το :

• Αν σκέφτεται σαν άνθρωπος (Μηχανισμός, γνωστική επιστήμη).

• Αν ενεργεί σαν άνθρωπος (Συμπεριφορά,Turing test).

• Αν σκέφτεται ορθολογικά (Μηχανισμός, νόμοι ορθής σκέψης).

• Αν ενεργεί ορθολογικά (Συμπεριφορά, ορθολογικοί πράκτορες).

Ιστορική Αναδρομή

Οι συλλογισμοί του Αριστοτέλη (384-322 π.Χ) παρείχαν πρότυπα εκφράσεων που έδιναν πάντα σωστά συμπεράσματα από σωστές υποθέσεις. Κατά τη δεκαετία του 1940 εμφανίστηκε η πρώτη μαθηματική περιγραφή τεχνητού νευρωνικού δικτύου, με πολύ περιορισμένες δυνατότητες επίλυσης αριθμητικών προβλημάτων. Καθώς ήταν εμφανές ότι οι ηλεκτρονικές υπολογιστικές συσκευές που κατασκευάστηκαν μετά τον Β' Παγκόσμιο Πόλεμο ήταν ένα τελείως διαφορετικό είδος μηχανής από ότι προηγήθηκε, η συζήτηση για την πιθανότητα εμφάνισης μηχανών με νόηση ήταν στην ακμή της. Το 1950 ο μαθηματικός Alan Turing, πατέρας της θεωρίας υπολογισμού και προπάτορας της τεχνητής νοημοσύνης, πρότεινε τη δοκιμή Turing μία απλή δοκιμασία που θα μπορούσε να εξακριβώσει αν μία μηχανή διαθέτει ευφυΐα. Η τεχνητή νοημοσύνη θεμελιώθηκε τυπικά ως πεδίο στη συνάντηση ορισμένων επιφανών Αμερικανών επιστημόνων το 1956 (John McCarthy, Marvin Minsky, Claude Shannon κλπ.). Τη χρονιά αυτή παρουσιάστηκε για πρώτη φορά και το Logic Theorist, ένα πρόγραμμα το οποίο στηριζόταν σε συμπερασματικούς κανόνες τυπικής λογικής και σε ευρετικούς αλγορίθμους αναζήτησης για να αποδεικνύει μαθηματικά θεωρήματα. Επόμενοι σημαντικοί σταθμοί ήταν η ανάπτυξη της γλώσσας

προγραμματισμού LISP το 1958 από τον John McCarthy, δηλαδή της πρώτης γλώσσας συναρτησιακού προγραμματισμού η οποία έπαιξε πολύ σημαντικό ρόλο στη δημιουργία εφαρμογών ΤΝ κατά τις επόμενες δεκαετίες, η εμφάνιση των γενετικών αλγορίθμων την ίδια χρονιά από τον Friedberg και η παρουσίαση του βελτιωμένου νευρωνικού δικτύου perceptron το ’62 από τον Rosenblatt. Κατά τα τέλη της δεκαετίας του ’60 όμως άρχισε ο χειμώνας της ΤΝ, μία εποχή κριτικής, απογοήτευσης και υποχρηματοδότησης των ερευνητικών προγραμμάτων καθώς όλα τα μέχρι τότε εργαλεία του χώρου ήταν κατάλληλα μόνο για την επίλυση εξαιρετικά απλών προβλημάτων. Στα μέσα του 70 ωστόσο προέκυψε μία αναθέρμανση του ενδιαφέροντος για τον τομέα λόγω των εμπορικών εφαρμογών που απέκτησαν τα έμπειρα συστήματα, μηχανές ΤΝ με αποθηκευμένη γνώση για έναν εξειδικευμένο τομέα και δυνατότητα ταχείας εξαγωγής λογικών συμπερασμάτων, τα οποία συμπεριφέρονται όπως ένας άνθρωπος ειδικός στον αντίστοιχο τομέα. Παράλληλα έκανε την εμφάνισή της η γλώσσα λογικού προγραμματισμού Prolog η οποία έδωσε νέα ώθηση στη συμβολική ΤΝ, ενώ στις αρχές της δεκαετίας του ’80 άρχισαν να υλοποιούνται πολύ πιο ισχυρά και με περισσότερες εφαρμογές νευρωνικά δίκτυα, όπως τα πολυεπίπεδα perceptron και τα δίκτυα Hopfield. Ταυτόχρονα οι γενετικοί αλγόριθμοι και άλλες συναφείς μεθοδολογίες αναπτύσσονταν πλέον από κοινού, κάτω από την ομπρέλα του εξελικτικού υπολογισμού. [3] [4]

Η εξέλιξη της Τεχνητής Νοημοσύνης

Οι απαρχές της Τεχνητής Νοημοσύνης ανάγονται στους "συλλογισμούς" του Αριστοτέλη (384-322 π.Χ), οι οποίοι παρείχαν πρότυπα εκφράσεων που έδιναν πάντα σωστά συμπεράσματα από σωστές υποθέσεις (Αριστοτέλεια συλλογιστική). Οι σημαντικότερες στιγμές που ακολούθησαν είναι αρχικά το 1854 όταν ο George Boole έθεσε τις βάσεις της προτασιακής λογικής. Αργότερα, 1879 ο Gottlieb Frege πρότεινε ένα σύστημα αυτοματοποιημένης συλλογιστικής και έθεσε τις βάσεις του κατηγορηματικού λογισμού (predicate calculus). Σχεδόν ένα αιώνα αργότερα, 1943 ο McCulloch και ο Pitts πρότειναν ένα μοντέλο τεχνητών νευρώνων που είχε τη δυνατότητα να μαθαίνει και να υπολογίζει κάθε υπολογίσιμη συνάρτηση. Την δεκαετία του 50' ο Alan Turing(1913-1954), ο οποίος θεωρείται ο πατέρας της ΤΝ, εμπνεύστηκε ένα τεστ (Turing test), για την αναγνώριση ευφυών μηχανών. Για τη δοκιμασία αυτή απαιτούνταν τρία δωμάτια χωρίς άμεση επικοινωνία, δύο άνθρωποι (Α και Β) και ένας Ηλεκτρονικός Υπολογιστής (Η/Υ). Ο άνθρωπος Α βρίσκεται απομονωμένος στο πρώτο δωμάτιο και στέλνει ταυτόχρονα πληκτρολογημένες ερωτήσεις ευφυΐας, υποκειμενικές, συναισθηματικές, κρίσεως, εγκυκλοπαιδικών γνώσεων κλπ. ,στον άνθρωπο Β και στον Η/Υ που βρίσκονται στα άλλα δύο δωμάτια. Ο άνθρωπος Β και ο Η/Υ απαντούν γραπτώς στις ερωτήσεις του Α, ο οποίος προσπαθεί μέσα από τις απαντήσεις, να ανακαλύψει ποιος είναι ο Β και ποιος ο Η/Υ. Εάν ο Α δεν μπορεί να διαχωρίσει δύο, τότε μπορεί να θεωρηθεί ότι ο Η/Υ κατέχει ευφυΐα. Οπότε, για να περάσει τη δοκιμασία αυτή ένας υπολογιστής, πρέπει να έχει τις εξής ικανότητες :

- Επεξεργασία φυσικής γλώσσας (natural language processing)
- Αναπαράσταση γνώσης (knowledge representation)
- Αυτοματοποιημένη συλλογιστική (automated reasoning)
- Μηχανική μάθηση (machine learning)
- Μηχανική όραση (computer vision)

• Ρομποτική (robotics)

Το 1951 ο Minsky και ο Edmonts υλοποίησαν το πρώτο νευρωνικό δίκτυο, το SNARC, με 40 νευρώνες, το οποίο χρησιμοποιούσε 3.000 λυχνίες και πέντε χρόνια αργότερα, το 1956 διοργανώθηκε το συνεδρίου (workshop) καθοριστικού στη γέννηση της Τεχνητής Νοημοσύνης. Οργανώθηκε από τους McCarthy, MinsKy, Shannon και Rochester και αφορούσε τη θεωρία αυτομάτων, νευρωνικά δίκτυα και μελέτη της ευφυΐας. Το 1958 ο McCarthy όρισε τη συναρτησιακή γλώσσα LISP. Πρότεινε ένα υποθετικό σύστημα(τον advice taker), που χρησιμοποιούσε γνώση (όπως το LT) αλλά αφορούσε γενικά καθημερινά προβλήματα. Την ίδια χρονιά ο Friedberg πρότεινε μια τεχνική, τη μηχανική εξέλιξη (machine evolution) όπως ονομάζεται τώρα, γενετικοί αλγόριθμοι (genetic algorithms). Λίγα χρόνια αργότερα κατά την δεκαετία του ’60 στο Stanford υλοποιήθηκε το πρώτο robot, το Shakey robot. Την χρονιά 1962 πραγματοποιήθηκαν βελτιώσεις της μεθόδου μάθησης των νευρωνικών δικτύων του Hebb από τον Rosenblatt με τα perceptrons, ενώ 3 χρόνια αργότερα, το 1965 το πρόγραμμα ELIZA του Weizenbaum μπορούσε να κάνει συζήτηση για οποιοδήποτε θέμα, χρησιμοποιώντας και παραφράζοντας τις προτάσεις που έδινε σαν ερώτηση ο χρήστης. Το πρόγραμμα ANALOGY του Tom Evans έλυνε προβλήματα γεωμετρικής αναλογίας που χρησιμοποιούνταν σε τεστ ευφυΐας και αναπτύχθηκε το 1968.

Η δεκαετία του 70’ ήταν η εποχή της πεποίθησης ότι τα συστήματα ήταν κατάλληλα μόνο για παιχνίδια (toy problems). Το χαρακτηριστικό των συστημάτων της εποχής, ήταν ότι περιείχαν ελάχιστη ή καθόλου γνώση για το πεδίο του προβλήματος (weak methods). Το εύρος εφαρμογών των νευρωνικών δικτύων ήταν μικρό. Αναπτύχθηκαν συστήματα που περιείχαν την απαιτούμενη γνώση ώστε να συμπεριφέρονται όπως οι άνθρωποι ειδικοί σε διάφορα θέματα. Ονομάστηκαν Έμπειρα Συστήματα (Expert Systems) ή Συστήματα Γνώσης (Knowledge Systems). Ορισμένα σημαντικά προγράμματα Έμπειρων συστημάτων είναι τα παρακάτω:

DENDRAL: Εύρεση της μοριακής δομής οργανικών ενώσεων με δεδομένα από φασματογράφο μάζας.

MYCIN :Διάγνωση μολύνσεων του αίματος.

PROSPECTOR :Συμβουλές για τοποθεσίες γεώτρησης για το στοιχείο μολυβδένιο.

R1 : Για την εταιρεία Digital Equipments. Διαμόρφωση των παραγγελιών με βάση τις ανάγκες των πελατών.

SHRDLU και LUNAR: Κατανόηση φυσικής γλώσσας.

Στις αρχές της δεκαετίας του 70’ προτάθηκε η γλώσσα προγραμματισμού Prolog, ενώ το 1975 προτάθηκαν από τον Minsky τα πλαίσια (frames).

Αργότερα το 1981 Ιάπωνες μελετητές ανακοίνωσαν το πρόγραμμα της 5ης γενιάς, ένα δεκαετές πρόγραμμα για την κατασκευή υπολογιστών με γλώσσα προγραμματισμού την Prolog. Στόχος ήταν να κατασκευαστούν ευφυή συστήματα, τα οποία εκτός των άλλων, θα ήταν σε θέση να επικοινωνούν πλήρως με τον άνθρωπο σε φυσική γλώσσα. Στα μέσα της δεκαετίας του 80’ Επανεμφανίστηκαν τα νευρωνικά δίκτυα όπως επίσης και ο αλγόριθμος μάθησης με οπισθοδρόμηση (Back-propagation) και εφαρμόστηκε σε πολλά προβλήματα με μεγάλη επιτυχία.

Σήμερα η Τεχνητή Νοημοσύνη έχει εξελιχθεί ώστε να καλύπτει όχι μόνο συστήματα που βασίζονται σε κανόνες και έμπειρα συστήματα, αλλά και συστήματα που βασίζονται στην εξελικτική διαδικασία ή σε πράκτορες. Περιοχές έρευνας περιλαμβάνουν την ανάπτυξη της γνώσης για συλλογιστικά μοντέλα, όπως οντολογίες και εφαρμογές εξόρυξης δεδομένων για την αυτόματη απόκτηση γνώσης. Σχετικοί τομείς έρευνας και ανάπτυξης περιλαμβάνουν το ηλεκτρονικό εμπόριο και την ηλεκτρονική οικονομία. [3] [5]

Εφαρμογές Τεχνητής Νοημοσύνης

Τα τελευταία χρόνια οι απαιτήσεις των εφαρμογών γίνονται όλο και πιο περίπλοκες και δύσκολα εντοπίζονται μεμονωμένες κατηγορίες μεθόδων ή ερευνητικών αντικειμένων της ΤΝ. Μερικές από τις εφαρμογές της ΤΝ είναι:

- Έμπειρα συστήματα
- Ρομποτική
- Επίλυση προβλημάτων
- Σχεδιασμός νευρωνικών δικτύων
- Ευφυή συστήματα πρακτόρων
- Τεχνητή όραση
- Ανάλυση δεδομένων μεγάλου όγκου

Τεχνητή Νοημοσύνη και Γενετικά Δεδομένα

Η ανάλυση δεδομένων μεγάλου όγκου βρίσκει εφαρμογή και στα γενετικά δεδομένα. Η μίμηση της ανθρώπινης νοημοσύνης είναι η έμπνευση για αλγόριθμους AI, αλλά οι εφαρμογές AI σε κλινικά δεδομένα στοχεύουν σε εργασίες που δεν είναι πρακτικές στην εκτέλεση χρησιμοποιώντας ανθρώπινη νοημοσύνη και είναι επιρρεπείς σε σφάλματα όταν αντιμετωπίζονται με τυπικές στατιστικές προσεγγίσεις. Πολλές μέθοδοι τεχνητής νοημοσύνης έχουν προσαρμοστεί για να αντιμετωπίσουν διάφορα προβλήματα που προκύπτουν κατα την ανάλυση κλινικών δεδομένων. Παρακάτω αναφέρονται μερικά από τα προβλήματα που έχουν ανατεθεί στην AI στον τομέα αυτο.

Αρχικά ένα συχνό πρόβλημα είναι το variant calling, καθώς η κλινική ερμηνεία των γονιδιωμάτων είναι ευαίσθητη στην ταυτοποίηση μεμονωμένων γενετικών μεταλλάξεων μεταξύ των εκατομμυρίων που ζουν σε κάθε γονιδίωμα, απαιτώντας εξαιρετική ακρίβεια.

Ένα μείγμα στατιστικών τεχνικών, συμπεριλαμβανομένων χειροποίητων χαρακτηριστικών, χρησιμοποιούνται για την αντιμετώπιση αυτών των ζητημάτων, με αποτέλεσμα την υψηλή ακρίβεια αλλά και την εμφάνιση σφαλμάτων. Οι αλγόριθμοι AI μπορούν να μάθουν αυτά τα σφάλματα από ένα μόνο γονιδίωμα με τη χρήση ενός γνωστού πρότυπου αναφοράς και να παράγουν ανώτερες εναλλακτικές κλήσεις. Ειδικότερα δημιουργήθηκε το DeepVariant, ένα συνελικτικό νευρωνικό δίκτυο (CNN), που εκπαιδεύτηκε απευθείας σε ανάγνωση ευθυγραμμίσεων (read alignments) χωρίς καμία εξειδικευμένη γνώση σχετικά με τη γονιδιωματική ή τις πλατφόρμες αλληλουχίας και ξεπέρασε τα τυπικά εργαλεία σε ορισμένες εργασίες variant calling. Η βελτιωμένη ακρίβεια πιστεύεται ότι οφείλεται στην ικανότητα των CNN να εντοπίζουν πολύπλοκες εξαρτήσεις στην αλληλουχία δεδομένων

Ακόμα η ερμηνεία των δεδομένων του ανθρώπινου γονιδιώματος βασίζεται στην αναγνώριση σχετικών γενετικών παραλλαγών μέσω προηγούμενης γνώσης και συμπεράσματος του αντίκτυπου των γενετικών αυτών μεταλλάξεων στα λειτουργικά γονιδιωματικά στοιχεία. Οι αλγόριθμοι AI μπορούν να βελτιώσουν τη χρήση προηγούμενων γνώσεων, ενημερώνοντας τη χαρτογράφηση φαινοτύπου σε γονότυπο. Αυτό σχετίζεται τόσο με τον σχολιασμό του γονιδιώματος όσο και με την ταξινόμηση παραλλαγών, επειδή πολλοί από τους αλγόριθμους AI που χρησιμοποιούνται για την πρόβλεψη της παρουσίας ενός λειτουργικού στοιχείου από πρωτογενή δεδομένα αλληλουχίας DNA χρησιμοποιούνται επίσης για την πρόβλεψη της επίδρασης μιας γενετικής παραλλαγής σε αυτά τα λειτουργικά στοιχεία. Η υπολογιστική αναγνώριση και πρόβλεψη μιας παραλλαγής είναι μια ανοιχτή πρόκληση στην ανθρώπινη γονιδιωματική. Τα πρόσφατα ευρήματα υποδηλώνουν ότι οι αλγόριθμοι AI θα βελτιώσουν ουσιαστικά την ικανότητά μας να κατανοούμε γενετικές παραλλαγές.

Τέλος τα ανθρώπινα γονιδιώματα περιέχουν πολλές γενετικές μεταλλάξεις που είτε είχαν προηγουμένως περιγραφεί ως παθογόνες είτε προβλεπόμενες να είναι παθογόνες, ανεξάρτητα από την ατομική κατάσταση της υγείας. Επομένως, η μοριακή διάγνωση της νόσου απαιτεί συχνά τόσο τον προσδιορισμό των υποψηφίων παθογόνων παραλλαγών όσο και τον προσδιορισμό της αντιστοιχίας μεταξύ του φαινοτύπου του ασθενή και εκείνων που

αναμένεται να προκύψουν από κάθε υποψήφια παθογόνο παραλλαγή. Οι αλγόριθμοι AI μπορούν να βελτιώσουν σημαντικά τη χαρτογράφηση του φαινοτύπου στον γονότυπο, ειδικά μέσω της εξαγωγής διαγνωστικών εννοιών υψηλότερου επιπέδου που είναι ενσωματωμένες σε ιατρικές εικόνες και EHR(ηλεκτρονικό ιστορικό υγείας). [3]

Κεφάλαιο 2: Εξόρυξη Δεδομένων

Εξόρυξη δεδομένων είναι η διαδικασία που περιλαμβάνει τον εντοπισμό καίριων μοτίβων ή προτύπων, τα οποία παρουσιάζουν ενδιαφέρον, καθώς και την δημιουργία περιγραφικών, κατανοητών και προβλεπτικών μοντέλων μεγάλης κλίμακας. Αυτή η διαδικασία περιλαμβάνει τους βασικούς αλγορίθμους που μας επιτρέπουν να εξάγουμε θεμελιώδης πληροφορίες και γνώσεις από τεράστιες ποσότητες δεδομένων.

Η εξαγωγή προτύπων από δεδομένα συμβαίνει εδώ και αιώνες. Οι πρώτες μέθοδοι για τον προσδιορισμό προτύπων ήταν αυτές της θεωρίας Bayes και της ανάλυσης της παλινδρόμησης. Ο πολλαπλασιασμός, η ευρεία διαθεσιμότητα και η εξέλιξη της τεχνολογίας υπολογιστών έχουν αυξήσει τον όγκο των συγκεντρωμένων δεδομένων και την ζήτηση για αποδοτικούς και αποτελεσματικούς χειρισμούς. Καθώς οι συλλογές δεδομένων αυξήθηκαν τόσο σε όγκο όσο και σε πολυπλοκότητα, η χειρωνακτική ανάλυση τους έχει αντικατασταθεί από την αυτόματη επεξεργασία δεδομένων. Σε αυτό συνέβαλαν άλλες ανακαλύψεις της επιστήμης των υπολογιστών, όπως τα νευρωνικά δίκτυα, η συσταδοποίηση, οι γενετικοί αλγόριθμοι (1950), τα δέντρα απόφασης (1960) και η μηχανή υποστήριξης διανυσμάτων(1990). Η εξόρυξη δεδομένων είναι η διαδικασία εφαρμογής αυτών των μεθόδων στα δεδομένα με σκοπό την αποκάλυψη άγνωστων προτύπων σε μεγάλα σύνολα δεδομένων. Αυτό γεφυρώνει το χάσμα της εφαρμοσμένης στατιστικής και της τεχνητής νοημοσύνης (τα οποία συνήθως παρέχουν το μαθηματικό υπόβαθρο) με την διαχείριση βάσης δεδομένων κάνοντας χρήση του τρόπο με τον οποίο αποθηκεύονται και κατατάσσονται στη βάση δεδομένων για να εκτελέσουν την θεωρία και τους διαθέσιμους αλγορίθμους περισσότερο αποτελεσματικά, επιτρέποντας σε τέτοιες μεθόδους να εφαρμόζονται σε μεγάλα σύνολα δεδομένων.

Στην πράξη, οι δύο βασικοί στόχοι της εξόρυξης δεδομένων τείνουν να είναι η πρόβλεψη και η περιγραφή. Η πρόβλεψη περιλαμβάνει τη χρήση ορισμένων μεταβλητών ή πεδίων στο σύνολο δεδομένων για την πρόβλεψη άγνωστων ή μελλοντικών τιμών άλλων

μεταβλητών ενδιαφέροντος. Η περιγραφή, από την άλλη πλευρά, επικεντρώνεται στην εύρεση προτύπων που περιγράφουν τα δεδομένα που μπορούν να ερμηνευτούν από τον άνθρωπο. Επομένως, είναι δυνατό να τοποθετηθούν οι δραστηριότητες εξόρυξης δεδομένων σε μία από τις δύο κατηγορίες :

1. Πρόβλεψη εξόρυξης δεδομένων, η οποία παράγει το μοντέλο του συστήματος που περιγράφεται από το δεδομένο σύνολο δεδομένων, ή

2. Περιγραφική εξόρυξη δεδομένων, η οποία παράγει νέες, σημαντικές πληροφορίες βάσει του διαθέσιμου συνόλου δεδομένων.

[7]

Η Εξέλιξη της Εξόρυξης Δεδομένων

Η εξόρυξη δεδομένων θέτει τις βάσεις της το 1763 όταν το θεώρημα του Thomas Bayes δημοσιεύεται και σχετίζει την τρέχουσα πιθανότητα με την αρχική πιθανότητα. Είναι θεμελιώδους σημασίας για την εξόρυξη δεδομένων καθώς επιτρέπει την κατανόηση σύνθετων πραγματικοτήτων που βασίζονται σε εκτιμώμενες πιθανότητες. Σχεδόν μισό αιώνα αργότερα το 1805 ο Adrien-Marie Legendre και ο Carl Friedrich Gauss εφαρμόζουν μία τεχνική παλινδρόμησης για να καθορίσουν τις τροχιές των σωμάτων γύρω από τον Ήλιο (κομήτες και πλανήτες). Ο στόχος της ανάλυσης παλινδρόμησης είναι η εκτίμηση των σχέσεων μεταξύ των μεταβλητών και η συγκεκριμένη μέθοδος που χρησιμοποιούνται σε αυτή την περίπτωση είναι η μέθοδος των ελαχίστων τετραγώνων. Η παλινδρόμηση είναι ένα από τα βασικά εργαλεία στην εξόρυξη δεδομένων. Αξιοσημείωτη ήταν η χρονιά του 1936 η οποία αποτέλεσε σημαντικό κομμάτι στην εξέλιξη της εξόρυξης δεδομένων καθώς ήταν η αρχή της εποχής των υπολογιστών. Το γεγονός αυτό κατέστησε εφικτή τη συλλογή και επεξεργασία μεγάλων ποσοτήτων δεδομένων. Σε ένα χαρτί του 1936, με Αριθμούς Υπολογιστών, ο Alan Turing εισήγαγε την ιδέα μιας καθολικής μηχανής ικανής να εκτελεί υπολογισμούς όπως οι σύγχρονοι υπολογιστές μας. Ο σύγχρονος υπολογιστής βασίζεται στις ιδέες του Turing.

Λίγο αργότερα το 1943 ο Warren McCulloch και ο Walter Pitts ήταν οι πρώτοι που δημιούργησαν ένα εννοιολογικό μοντέλο ενός νευρικού δικτύου. Σε μία εργασία με τίτλο "A logical calculus of the ideas immanent in nervous activity", περιγράφουν την ιδέα ενός νευρώνα σε ένα δίκτυο. Κάθε ένας από αυτούς τους νευρώνες μπορεί να κάνει 3 πράγματα: να λαμβάνει εισόδους, να εισάγει διαδικασίες και να παράγει αποτελέσματα. Το 1965, ο Lawrence J. Fogel δημιούργησε μια νέα εταιρεία αποκαλούμενη Decision Science, Inc. για εφαρμογές εξελικτικού προγραμματισμού. Ήταν η πρώτη εταιρεία που εφάρμοσε τον εξελικτικό υπολογισμό για την επίλυση πραγματικών προβλημάτων. Πέντε χρόνια αργότερα έγινε δυνατή η αποθήκευση και η αναζήτηση μεγάλου όγκου δεδομένων (terabytes, petabytes), με εξελιγμένα συστήματα διαχείρισης βάσεων δεδομένων. Επιπλέον, οι αποθήκες δεδομένων επιτρέπουν στους χρήστες να μετακινούνται από έναν τρόπο σκέψης προσανατολισμένο στις συναλλαγές σε έναν πιο αναλυτικό τρόπο προβολής των δεδομένων. Ωστόσο, η εξόρυξη πολύπλοκων στοιχείων από αυτές τις αποθήκες δεδομένων των πολυδιάστατων μοντέλων είναι πολύ περιορισμένη. Το 1975 ο John Henry Holland έγραψε το βιβλίο με τίτλο "Adaptation in Natural and Artificial Systems", το πρωτοποριακό βιβλίο για τους γενετικούς αλγόριθμους. Είναι το βιβλίο που ξεκίνησε αυτόν τον τομέα σπουδών, παρουσιάζοντας τα θεωρητικά θεμέλια και διερευνώντας τις εφαρμογές. Στις αρχές της δεκαετίας του 80' η εταιρία HNC δημιουργεί τη φράση "database mining". Αυτός ο όρος προοριζόταν να προστατεύει ένα προϊόν που ονομάζεται Database Mining Workstation. Ήταν ένα εργαλείο γενικού σκοπού για την οικοδόμηση μοντέλων νευρωνικών δικτύων το οποίο δεν είναι πλέον διαθέσιμο. Κατά τη διάρκεια αυτής της περιόδου οι εξελιγμένοι αλγόριθμοι μπορούν να "μαθαίνουν" τις σχέσεις από τα δεδομένα που επιτρέπουν στους εμπειρογνώμονες του αντικειμένου να διασαφηνίσουν τι σημαίνουν οι σχέσεις. Το 1989 ο όρος "Knowledge Discovery in Databases" δημιουργείται από τον Gregory Piatetsky-Shapiro. Επίσης, την ίδια περίοδο ιδρύει το πρώτο workshop που ονομάζεται επίσης KDD. Η δεκαετία του 90' θεωρείται υψίστης σημασίας για την εξόρυξη δεδομένων καθώς, ο όρος "εξόρυξη δεδομένων" δημιουργείται. Οι εταιρείες λιανικής και η χρηματοοικονομική κοινότητα χρησιμοποιούν την εξόρυξη δεδομένων για να αναλύσουν δεδομένα και να αναγνωρίσουν τάσεις για την αύξηση της πελατειακής τους βάσης, να προβλέψουν διακυμάνσεις των επιτοκίων, των τιμών των μετοχών, ζήτηση των πελατών

κ.α. Το 1992 οι Bernhard E. Boser, Isabelle M. Guyon και Vladimir N. Vapnik πρότειναν μια βελτίωση στην αρχική Support Vector Machine που επιτρέπει τη δημιουργία μη γραμμικών ταξινομητών. Τα Support Vector Machines είναι μια προσέγγιση εποπτευόμενης μάθησης που αναλύει τα δεδομένα και αναγνωρίζει πρότυπα που χρησιμοποιούνται για ανάλυση ταξινόμησης και παλινδρόμησης. Ένα χρόνο αργότερα το 1993 ο Gregory Piatetsky-Shapiro ξεκινάει το Knowledge Discovery Nuggets (KDnuggets) newsletter. Αρχικά σχεδιάστηκε για να ενώσει την ομάδα ερευνητών που παρευρισκόντουσαν στο KDD Workshop. Ωστόσο, το KDnuggets.com φαίνεται να έχει ένα ευρύτερο κοινό τώρα.

Το 2001, αν και ο όρος “data science” υπήρχε από το 1960, ο William S. Cleveland την παρουσίασε ως ανεξάρτητο επιστημονικό κλάδο το 2001. Σύμφωνα με τις Build Data Science Teams, ο DJ Patil και ο Jeff Hammerbacher χρησιμοποίησαν τότε τον όρο για να περιγράψουν τους ρόλους τους στο LinkedIn και στο Facebook, ενώ το 2003 το βιβλίο Moneyball, από τον Michael Lewis, δημοσιεύεται και αλλάζει τον τρόπο με τον οποίο πολλά major league front offices δουλεύουν. Η Oakland Athletics χρησιμοποίησε μια στατιστική προσέγγιση για να βρει ωφέλιμα χαρακτηριστικά σε παίκτες που ήταν υποτιμημένοι και φθηνότεροι. Με αυτόν τον τρόπο, συγκέντρωσαν με επιτυχία μια ομάδα που τους έφερε στα playoffs του 2002 και του 2003 με το 1/3 της μισθοδοσίας. Σχεδόν μια δεκαετία αργότερα το τον Φεβρουάριο του 2015, ο DJ Patil έγινε ο πρώτος Chief Data Scientist του Λευκού Οίκου. Σήμερα, η εξόρυξη δεδομένων είναι ευρέως διαδεδομένη στις επιχειρήσεις, την επιστήμη, τη μηχανική, την ιατρική και σε πολλούς άλλους κλάδους. Η συλλογή δεδομένων γίνεται φθηνότερη και οι συσκευές συλλογής δεδομένων πολλαπλασιάζονται. Μια από τις πιο δραστήριες τεχνικές που διερευνούνται σήμερα είναι η Deep Learning. Με την δυνατότητα να καταγράφει εξαρτήσεις και σύνθετα σχέδια αποτελεσματικότερα από άλλες τεχνικές, αναζωπυρώνει μερικές από τις μεγαλύτερες προκλήσεις στον κόσμο της εξόρυξης δεδομένων, της επιστήμης των δεδομένων και της τεχνητής νοημοσύνης. [8]

Διαδικασία Εξόρυξης Δεδομένων

Η γενική πειραματική διαδικασία της εξόρυξης δεδομένων περιλαμβάνει τα ακόλουθα βήματα :

• **Συλλογή δεδομένων** : Αυτό το βήμα αφορά τον τρόπο με τον οποίο παράγονται και συλλέγονται τα δεδομένα. Σε γενικές γραμμές, υπάρχουν δύο διακριτές πιθανότητες. Η πρώτη είναι όταν η διαδικασία παραγωγής δεδομένων βρίσκεται υπό τον έλεγχο ενός ειδικού (modeler - σχεδιαστής). Αυτή η προσέγγιση είναι γνωστή ως σχεδιασμένο πείραμα (designed experiment). Η δεύτερη πιθανότητα είναι όταν ο ειδικός δεν μπορεί να επηρεάσει τη διαδικασία παραγωγής δεδομένων. Αυτή είναι γνωστή ως παρατηρητική προσέγγιση (observational approach).

• **Προ-επεξεργασία δεδομένων** : Η προεπεξεργασία είναι απαραίτητη για την ανάλυση πολυπαραγοντικών συνόλων δεδομένων πριν την εξόρυξη δεδομένων. Καθώς η εξόρυξη δεδομένων μπορεί να αποκαλύψει μόνο τα πρότυπα που πράγματι εμφανίζονται στα δεδομένα, το σύνολο δεδομένων που ερευνούμε, πρέπει να είναι αρκετά μεγάλο για να περιέχει αυτά τα πρότυπα. Σε ένα μεγάλο πλήθος δεδομένων είναι λογικό να εμφανίζονται παρατηρήσεις που περιέχουν θόρυβο ή παρατηρήσεις με ελλειπή ή ελλείποντα δεδομένα. Έτσι από το ερευνώμενο σύνολο κρίνεται απαραίτητο να αφαιρεθούν τα παραπάνω.

• **Μοντελοποίηση** : Κατά τη φάση της μοντελοποίησης, επιλέγονται συγκεκριμένοι αλγόριθμοι μοντελοποίησης και εκτελούνται στα δεδομένα. Η επιλογή των συγκεκριμένων αλγορίθμων που χρησιμοποιούνται στη διαδικασία εξόρυξης δεδομένων βασίζεται στη φύση της ερώτησης και στα επιθυμητά αποτελέσματα.

• **Αξιολόγηση**: Αυτό το βήμα καθορίζει τον βαθμό στον οποίο το προκύπτον μοντέλο πληροί τις απαιτήσεις του χρήστη. Η αξιολόγηση μπορεί να γίνει δοκιμάζοντας το μοντέλο σε πραγματικές εφαρμογές. Το μοντέλο ελέγχεται για τυχόν λάθη ή βήματα που πρέπει να επαναληφθούν.

• **Ερμηνεία και εξαγωγή συμπερασμάτων**: Στο βήμα αυτό τα μοντέλα που προέκυψαν ερμηνεύονται με σκοπό την εξόρυξη γνώσης και την εξαγωγή συμπερασμάτων. [9]

Εξόρυξη Δεδομένων ή Τεχνητή Νοημοσύνη

Η εξόρυξη δεδομένων αναφέρεται στην εξερεύνηση ενός υπάρχοντος μεγάλου συνόλου δεδομένων για να ανακαλυφθούν προηγούμενα άγνωστα μοτίβα, σχέσεις και ανωμαλίες που υπάρχουν στα δεδομένα. Μας δίνει τη δυνατότητα να βρούμε εντελώς νέες πληροφορίες.

Η τεχνητή νοημοσύνη (AI) αναφέρεται στους υπολογιστές που αναλύουν μεγάλα σύνολα δεδομένων και στη συνέχεια «μαθαίνουν» μοτίβα που θα τους βοηθήσουν να κάνουν προβλέψεις για νέα σύνολα δεδομένων. Εκτός από τον αρχικό προγραμματισμό και ίσως κάποια ρύθμιση, ο υπολογιστής δεν χρειάζεται ανθρώπινη αλληλεπίδραση για να μάθει από τα δεδομένα.

Στην πραγματικότητα, η τεχνητή νοημοσύνη μπορεί να χρησιμοποιήσει ορισμένες τεχνικές εξόρυξης δεδομένων για να δημιουργήσει μοντέλα και να βρει μοτίβα, έτσι ώστε να μπορεί να κάνει καλύτερες προβλέψεις και η εξόρυξη δεδομένων μπορεί μερικές φορές να χρησιμοποιήσει τεχνικές τεχνητής νοημοσύνης για να παράγει ακριβέστερη ανάλυση. Αυτό οδηγεί στο συμπέρασμα ότι η αλληλεπίδραση τους θα αποτελέσει ένα από τα καλύτερα εργαλεία για ανάλυση δεδομένων.

Κεφάλαιο 3: Αλγόριθμοι Τεχνητής Νοημοσύνης

Άτυπα, με τον όρο «αλγόριθμος» χαρακτηρίζεται κάθε καλώς ορισμένη υπολογιστική διαδικασία, η οποία καλείται να επιλύσει ένα συγκεκριμένο πρόβλημα, εντός πεπερασμένου χρόνου. Τυπικότερα, αποτελεί μία ακολουθία, η οποία απεικονίζει την είσοδο του προβλήματος, δηλαδή τα δεδομένα του και την έξοδο, δηλαδή την λύση του προβλήματος. Διαφορετικοί αλγόριθμοι Τεχνητής Νοημοσύνης μπορούν να χρησιμοποιηθούν για την επίλυση μιας κατηγορίας προβλημάτων. Παρακάτω παρουσιάζονται οι διαφορετικοί τύποι αλγορίθμων που εμπίπτουν στα προβλήματα παλινδρόμησης, ομαδοποίησης και ταξινόμησης. [10]

Αλγόριθμοι Παλινδρόμησης

Οι αλγόριθμοι παλινδρόμησης είναι αλγόριθμοι εποπτευόμενης (supervised) μηχανικής μάθησης. Οι αλγόριθμοι αυτοί μπορούν να προβλέψουν τις τιμές εξόδου βάσει των σημείων δεδομένων εισόδου που τροφοδοτούνται στο σύστημα εκμάθησης. Οι πιο συνηθισμένοι αλγόριθμοι σε αυτήν την κατηγορία είναι

α) Γραμμική παλινδρόμηση/Linear regression

Χρησιμοποιείται για τη μέτρηση γνήσιων ιδιοτήτων λαμβάνοντας υπόψη τις συνεπείς μεταβλητές. Είναι ο απλούστερος από όλους τους αλγορίθμους παλινδρόμησης αλλά μπορεί να εφαρμοστεί μόνο σε περιπτώσεις γραμμικής σχέσης ή ενός γραμμικά διαχωρίσιμου προβλήματος. Ο αλγόριθμος σχεδιάζει μια ευθεία γραμμή μεταξύ σημείων δεδομένων που ονομάζεται best-fit γραμμή ή γραμμή παλινδρόμησης και χρησιμοποιείται για την πρόβλεψη νέων τιμών. [11]

β) Παλινδρόμηση Λάσου/Lasso Regression

Ο αλγόριθμος παλινδρόμησης Lasso λειτουργεί αποκτώντας το υποσύνολο των προβλέψεων που ελαχιστοποιούν το σφάλμα πρόβλεψης για μια μεταβλητή. Αυτό

επιτυγχάνεται περιορίζοντας τα σημειακά δεδομένα και επιτρέποντας σε ορισμένα από αυτά να συρρικνωθούν σε μηδενική τιμή. [12]

γ) Λογιστική παλινδρόμηση/Logistic Regression

Η λογιστική παλινδρόμηση χρησιμοποιείται κυρίως για δυαδική ταξινόμηση. Αυτή η μέθοδος επιτρέπει την ανάλυση ενός συνόλου μεταβλητών και την πρόβλεψη ενός κατηγορηματικού αποτελέσματος. [13]

Αλγόριθμοι Ομαδοποίησης

Η ομαδοποίηση είναι η διαδικασία διαχωρισμού και οργάνωσης των σημείακών δεδομένων σε ομάδες με βάση τις ομοιότητες των μελών της ομάδας. Αυτό είναι μέρος της μη εποπτευόμενης (unsupervised) μάθησης. Ο κύριος στόχος είναι η ομαδοποίηση παρόμοιων αντικειμένων. Ακολουθούν οι πιο συνηθισμένοι αλγόριθμοι ομαδοποίησης.

α) Ομαδοποίηση K-Means

Είναι ο απλούστερος αλγόριθμος χωρίς επίβλεψη. Ο αλγόριθμος συγκεντρώνει παρόμοια σημειακά δεδομένων και στη συνέχεια τα συνδέει σε ένα σύμπλεγμα. Η ομαδοποίηση γίνεται με τον υπολογισμό του κέντρου της ομάδας σημειακών δεδομένων και στη συνέχεια αξιολογώντας την απόσταση κάθε σημείου από το κέντρο του συμπλέγματος. Με βάση την απόσταση, το σημείο που αναλύεται στη συνέχεια αντιστοιχείται στο πλησιέστερο σύμπλεγμα. Το «K» στο K-means σημαίνει τον αριθμό συστάδων στα οποία ομαδοποιούνται τα σημειακά δεδομένα. [14]

β) Μέση μετατόπιση/Mean Shift

Αυτός ο αλγόριθμος λειτουργεί ενημερώνοντας τα υποψήφια κέντρα ώστε να είναι ο μέσος όρος των σημείων σε μια δεδομένη περιοχή. Αυτοί οι υποψήφιοι στη συνέχεια φιλτράρονται σε ένα στάδιο μετα-επεξεργασίας για να εξαλείψουν τυχόν διπλότυπα και να σχηματίσουν το τελικό σύνολο κέντρων. [15]

γ) Αλγόριθμος Ιεραρχικής Ομαδοποίησης/ Hierarchical clustering

Αυτοί οι αλγόριθμοι ταξινομούν τις συστάδες με ιεραρχική σειρά αφού μάθουν τα σημειακά δεδομένα και κάνουν παρατηρήσεις ομοιότητας. Μπορεί να είναι διαχωριστικής ομαδοποίησης, για προσέγγιση από κάτω προς τα κάτω ή συγκεντρωτικής ομαδοποίησης, για προσέγγιση από κάτω προς τα πάνω. [16]

Αλγόριθμοι Ταξινόμησης

Οι αλγόριθμοι ταξινόμησης αποτελούν μέρος της εποπτευόμενης μάθησης. Αυτοί οι αλγόριθμοι χρησιμοποιούνται για να διαιρέσουν την υποκείμενη μεταβλητή σε διαφορετικές κατηγορίες και στη συνέχεια να προβλέψουν την κλάση για μια δεδομένη είσοδο. Η παρούσα εργασία εστιάζει στους αλγόριθμους αυτούς καθώς ο τελικός σκοπός της είναι η ταξινόμηση γενετικών δεδομένων. Για την επίτευξη του στόχου οι αλγόριθμοι ταξινόμησης που παρουσιάζονται παρακάτω εφαρμόζονται στην βάση δεδομένων που επιλέχθηκε.

α) Dummy Classifier

Ο Dummy Classifier είναι ένας ταξινομητής που κάνει προβλέψεις χρησιμοποιώντας απλούς κανόνες και αποτελεί ένα απλό αλγόριθμο που χρησιμοποιείται ως μέτρο σύγκρισης για άλλους αλγορίθμους. Ο ταξινομητής αυτός δίνει ένα μέτρο απόδοσης “βασικής γραμμής” (baseline performance), δηλαδή το ποσοστό επιτυχίας που πρέπει να περιμένει κανείς, έστω και αν απλά μαντέψει.

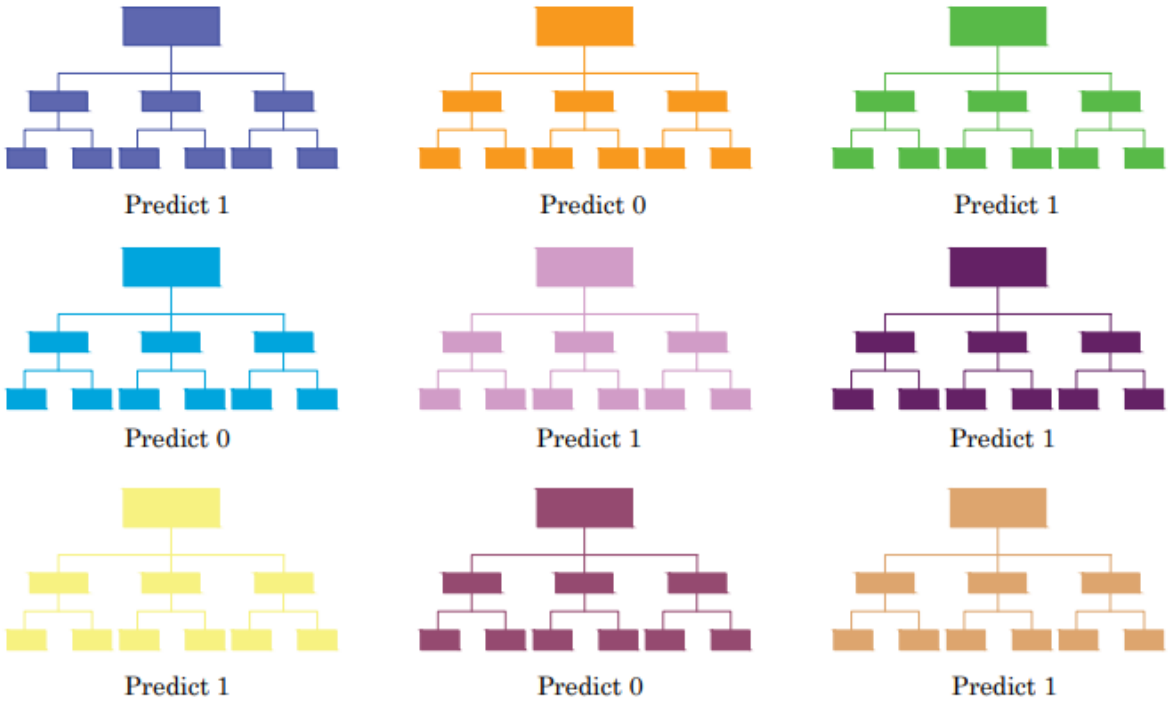
β) Τυχαίο δάσος/Random Forest

Στον αλγόριθμο Random Forest δημιουργείται ένα μεγάλο πλήθος δέντρων τα οποία λειτουργούν ως σύνολο που ονομάζεται δάσος. Όσο περισσότερα δέντρα τόσο πιο ισχυρό το σύνολο που μελετάται. Ο αλγόριθμος Random Forest δημιουργεί δέντρα αποφάσεων σε τυχαία επιλεγμένα δεδομένα. Κάθε δέντρο που δημιουργείται προβλέπει μια κατηγορία. Στη

συνέχεια η κατηγορία που έχει τις περισσότερους «ψήφους» από τα δέντρα δηλαδή που έχει προβλεφθεί περισσότερες φορές αποτελεί και το αποτέλεσμα του αλγορίθμου.

Η θεμελιώδης έννοια πίσω από τον Random Forest είναι μια απλή αλλά ισχυρή, η σοφία του πλήθους.

Στην παρακάτω εικόνα φαίνεται ένα απλό παράδειγμα λειτουργείας του αλγορίθμου. Έχουν δημιουργηθεί 9 decision trees εκ των οποίων τα 6 προβλέπουν την κατηγορία 1 ενώ τα 3 την κατηγορία 0. Επομένως το αποτέλεσμα είναι η κατηγορία 1. [17]



Εικόνα 1: 6 Prediction 1 και 3 Prediction 0

γ) Κ κοντινότερος γείτονας/K Nearest Neighbors Classifier

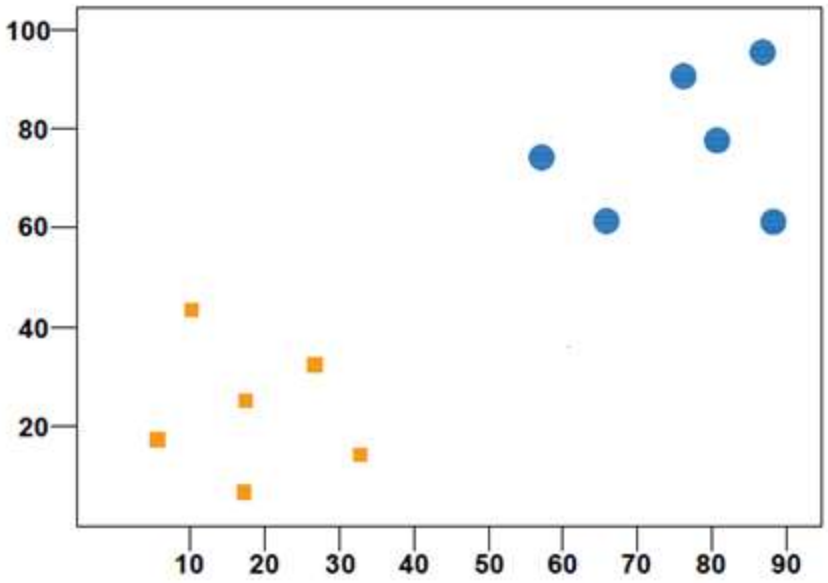
Ο αλγόριθμος k-Nearest Neighbors (KNN) είναι ένας απλός, εύκολος στην εφαρμογή εποπτευόμενος αλγόριθμος μηχανικής μάθησης που μπορεί να χρησιμοποιηθεί για την

επίλυση προβλημάτων ταξινόμησης και παλινδρόμησης Ο αλγόριθμος ΚΝΝ υποθέτει ότι παρόμοια πράγματα υπάρχουν πολύ κοντά το ένα με το άλλο.

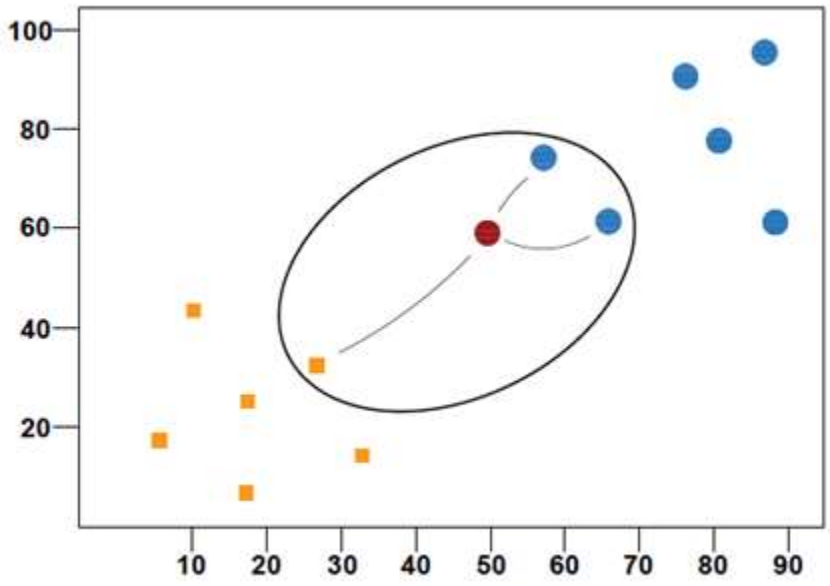
Ο αλγόριθμος ΚΝΝ ακολουθεί τα εξής βήματα.

1. Εισαγωγή δεδομένων
2. Αρχικοποίηση του Κ στον επιθυμητό αριθμό γειτόνων
3. Για κάθε σημείο στα δεδομένα υπολογίζεται η απόσταση μεταξύ του ερωτήματος (test data) και του τρέχοντος παραδείγματος (training data) από τα δεδομένα.
4. Ταξινόμηση συλλογής των αποστάσεων από τις μικρότερες έως τις μεγαλύτερες (σε αύξουσα σειρά)
5. Επιλογή πρώτων καταχωρήσεων Κ από την ταξινομημένη συλλογή
6. Λήψη ετικετών των επιλεγμένων καταχωρίσεων Κ

Για την καλύτερη κατανόηση ακολουθεί ένα απλό παράδειγμα του αλγορίθμου. Έστω μια βάση δεδομένων όπως φαίνεται στην παρακάτω εικόνα.



Εικόνα 2: Τυχαία βάση δεδομένων



Εικόνα 3: Εφαρμογή αλγορίθμου KNN

Έστω ότι πρέπει ένα καινούριο δεδομένο, για παράδειγμα το δεδομένο στην θέση (50,60), να ταξινομηθεί στην κατηγορία μπλε ή πορτοκαλί. Τίθεται το κ=3 και θα εντοπιστούν οι κοντινότεροι γείτονες του κόκκινου σημείου όπως ακριβώς στην επόμενη εικόνα. Φαίνεται λοιπόν ότι οι 3 κοντινότεροι γείτονες του καινούριου δεδομένου είναι 2 στην κατηγορία μπλε και 1 στην κατηγορία πορτοκαλί. Επομένως το καινούριο σημείο θα ταξινομηθεί στη μπλε κατηγορία. [18]

δ) Μπερνούλι αφελής ταξινομητής Bayes/Bernouli Naive Bayes Classifier

Ο αλγόριθμος Bernouli Naive Bayes Classifier είναι μια παραλλαγή του Naive Bayes. Ο Naive Bayes είναι ένας αλγόριθμος κατηγοριοποίησης μηχανικής μάθησης βασισμένος στο θεώρημα του Bayes, το οποίο δίνει την πιθανότητα να συμβεί ένα γεγονός. Ο κατηγοριοποιητής Naive Bayes είναι πιθανοτικός κατηγοριοποιητής που σημαίνει ότι με βάση τα δεδομένα εισόδου, προβλέπει τη πιθανότητα αυτών να έχουν κατηγοριοποιηθεί για όλες τις κλάσεις.

Το θεώρημα Bayes διατυπώνεται ως εξής:

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}$$

όπου

P(A|B) = η πιθανότητα του A δωθέντος του B

P(B|A) = η πιθανότητα του B δωθέντος του A

P(A) = η πιθανότητα να συμβεί το A

P(B) = η πιθανότητα να συμβεί το B

Το κύριο χαρακτηριστικό του Bernoulli Naive Bayes είναι ότι δέχεται δεδομένα μόνο με δυαδικές τιμές όπως yes/no, 0/1, true/false. [19]

Αξιολόγηση Αλγορίθμων

Η αξιολόγηση του αλγορίθμου μηχανικής μάθησης αποτελεί ουσιαστικό μέρος για την απόδοση του μοντέλου που δημιουργήθηκε. Αφού έχει δημιουργηθεί το μοντέλο είναι αναγκαίο να αξιολογηθούν οι προβλέψεις του. Για τον σκοπό αυτό έχουν δημιουργηθεί ορισμένα μέτρα απόδοσης.

Μέτρα Απόδοσης

Αρχικά είναι απαραίτητη η δημιουργία ενός πίνακα που παραθέτει τις προβλέψεις του μοντέλου σε μορφή True Positive, False Positive, True Negative και False Negative.

	Predicted Class		
Actual Class		Class = Yes	Class = No
	Class = Yes	True Positive	False Negative
	Class = No	False Positive	True Negative

Εικόνα 4: Πίνακας Προβλέψεων

True negative και True positive είναι οι παρατηρήσεις που προβλέπονται σωστά και επομένως εμφανίζονται με πράσινο χρώμα. Τα False Positive και False Negative πρέπει να ελαχιστοποιηθούν γι' αυτό και εμφανίζονται με κόκκινο χρώμα.

- **True Positives (TP) -** Αυτές είναι οι σωστά προβλεπόμενες θετικές τιμές που σημαίνει ότι η τιμή της πραγματικής κλάσης είναι "ναι" και η τιμή της προβλεπόμενης τάξης είναι επίσης "ναι".
- **True Negatives (TN)** - Αυτές είναι οι σωστά προβλεπόμενες αρνητικές τιμές που σημαίνει ότι η τιμή της πραγματικής κλάσης είναι "όχι" και η τιμή της προβλεπόμενης κλάσης είναι επίσης "όχι".

Ψευδώς θετικά και ψευδώς αρνητικά, αυτές οι τιμές εμφανίζονται όταν η πραγματική τάξη σας έρχεται σε αντίθεση με την προβλεπόμενη τάξη.

- **False Positives (FP)** - Όταν η πραγματική τάξη είναι "όχι" και η προβλεπόμενη τάξη είναι ναι.

- **False Negatives (FN)** - Όταν η πραγματική τάξη είναι “ναι” αλλά η προβλεπόμενη τάξη είναι “όχι”.

Με το που δημιουργηθούν οι παράμετροι αυτοί, είναι εφικτός ο υπολογισμός των παρακάτω μέτρων απόδοσης.

α) Accuracy / Ορθότητα

Η ορθότητα είναι το πιο διαισθητικό μέτρο απόδοσης και είναι απλώς μια αναλογία των σωστά προβλεπόμενων παρατηρήσεων προς τις συνολικές παρατηρήσεις. Η ορθότητα είναι ένα εξαιρετικό μέτρο, αλλά μόνο όταν τα σύνολα είναι συμμετρικά, όπου οι τιμές ψευδώς θετικών και ψευδών αρνητικών είναι σχεδόν ίσες.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

β) Precision / Ακρίβεια

Το μέτρο αυτό ερμηνεύεται ως ο λόγος των σωστά προβλεπόμενων θετικών παρατηρήσεων προς τις συνολικές προβλεπόμενες θετικές παρατηρήσεις.

$$Precision = \frac{TP}{TP + FP}$$

γ) Recall / Ανάκληση

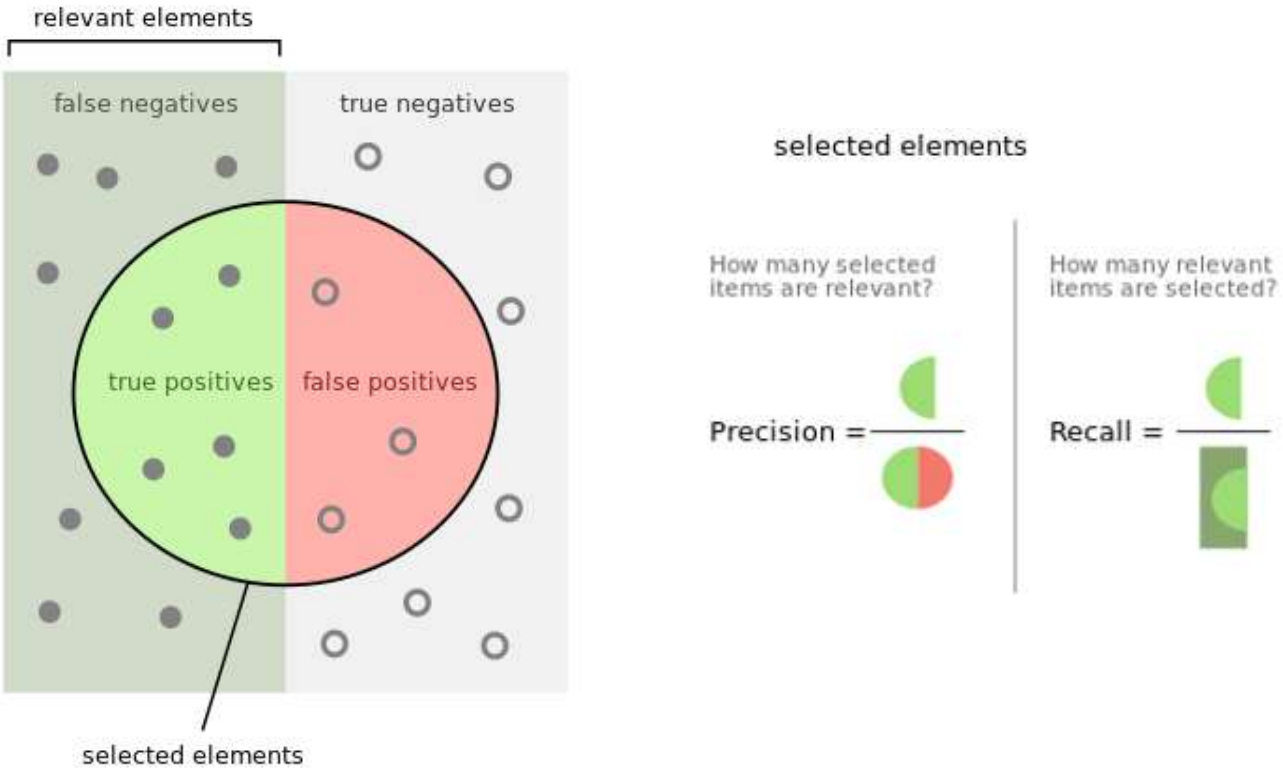
Το recall είναι η αναλογία των σωστά προβλεπόμενων θετικών παρατηρήσεων προς όλες τις παρατηρήσεις που θα έπρεπε να ταυτοποιηθούν θετικές.

$$Recall = \frac{TP}{TP + FN}$$

δ) F1 Score / F1 σκορ

Το F1-score είναι ο σταθμισμένος μέσος όρος Precision και Recall. Επομένως, αυτό το σκορ λαμβάνει υπόψη τόσο ψευδώς θετικές όσο και ψευδώς αρνητικές προβλέψεις. Διαισθητικά δεν είναι τόσο εύκολο να κατανοηθεί όσο η ακρίβεια, αλλά το F1-score είναι συνήθως πιο χρήσιμο από αυτήν, ειδικά όταν η κατανομή της τάξης είναι άνιση. Η ακρίβεια λειτουργεί καλύτερα αν οι ψευδώς θετικές και οι ψευδώς αρνητικές παρατηρήσεις έχουν παρόμοιο πλήθος. Εάν η κατανομή των ψευδώς θετικών και των ψευδώς αρνητικών παρατηρήσεων είναι πολύ διαφορετική, είναι καλύτερο να αξιοποιηθούν τόσο το Precision όσο και το Recall. Η υψηλότερη πιθανή τιμή που μπορεί να λάβει ένα F-score είναι το 1, όταν τα precision και recall είναι τέλεια ενώ το χαμηλότερο 0 όταν ένα από τα δύο είναι 0. [20] [21]

$$F1 = \frac{2}{recall^{-1} + precission^{-1}} = 2\frac{precion * recall}{precision + recall} = \frac{TP}{TP + \frac{1}{2}(FP + FN)}$$



Εικόνα 5: Precision και Recall

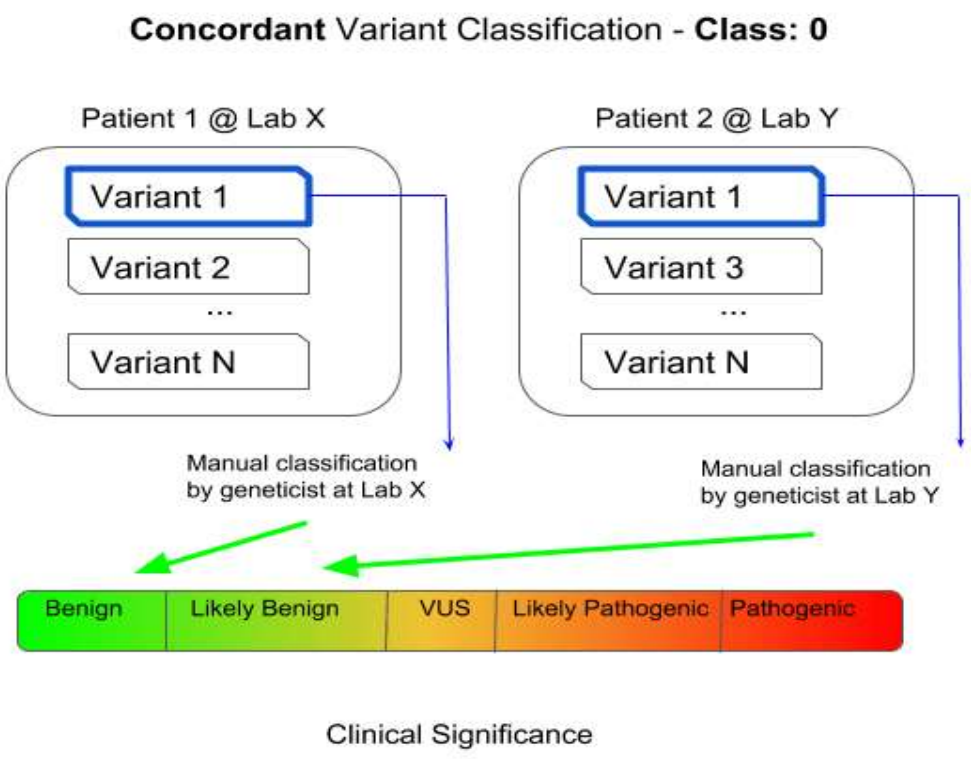
Κεφάλαιο 4: Δεδομένα

Βάση Δεδομένων

Το ClinVar είναι ένα δωρεάν διαθέσιμο, δημόσιο αρχείο που περιέχει ανθρώπινες γενετικές μεταλλάξεις και αναφέρει το αντίκτυπό τους για διάφορες ασθένειες. Διατηρείται στην Εθνική Βιβλιοθήκη Ιατρικής (NLM) των Εθνικών Ινστιτούτων Υγείας (NIH) στο Εθνικό Κέντρο Πληροφοριών Βιοτεχνολογίας (NCBI). Οι ισχυρισμοί της κλινικής σημασίας μιας μεταλλάξεις ή ενός συνόλου μεταλλάξεων υποβάλλονται στο ClinVar από εργαστήρια κλινικών δοκιμών, ερευνητικά εργαστήρια, βάσεις δεδομένων για συγκεκριμένους τόπους, ομάδες ειδικών και άλλες ομάδες. Οι υποβολές περιλαμβάνουν περιγραφή της παραλλαγής, την κατάσταση για την οποία ερμηνεύτηκε η παραλλαγή, την ερμηνεία της κλινικής σημασίας της παραλλαγής, με την επιλογή παροχής τρόπου κληρονομιάς, και αποδεικτικά στοιχεία για αυτήν την ερμηνεία. Το ClinVar συγκεντρώνει τις υποβολές που βασίζονται τόσο στην μετάλλαξη όσο και στο ζεύγος μετάλλαξης-συνθήκης και υπολογίζει μια συνολική ερμηνεία για να δείξει εάν υπάρχει συναίνεση ή διαφωνία μεταξύ των υποβαλλόντων.

Η ClinVar διαθέτει επί του παρόντος περισσότερα από μισό εκατομμύριο κατατεθέντα αρχεία που αντιστοιχούν σε περισσότερες από 331.000 παραλλαγές. Το ClinVar περιλαμβάνει παραλλαγές αλληλουχίας και δομικές παραλλαγές. Περισσότερες από 800 ομάδες από 60 χώρες υποβάλλουν στο ClinVar, συμπεριλαμβανομένων 76 εργαστηρίων που υποβάλλουν ερμηνείες από άμεσες κλινικές δοκιμές.

Το περιεχόμενο του ClinVar μπορεί να χωριστεί σε πέντε κύριες κατηγορίες: υποβολή, μετάλλαξη, φαινότυπος, ερμηνεία και στοιχεία. Ο μοναδικός συνδυασμός αποστολέα, παραλλαγής και φαινοτύπου καθορίζει μια μονάδα εγγραφής και εκχωρείται μια προσχώρηση στη μορφή SCV000000000.0 (SCV). Όποτε είναι δυνατόν, το περιεχόμενο είναι εξαιρετικά δομημένο και εναρμονισμένο με ελεγχόμενα λεξιλόγια ή άλλα πρότυπα δεδομένων. Η βάση δεδομένων αυτή διαθέτει 46 διαφορετικές μεταβλητές (στήλες) οι οποίες θα αναλυθούν στην ενότητα του κώδικα. [22] [23]



Εικόνα 6: Variant Classification

Συσχέτιση των Δεδομένων

Ένας όρος που θα επηρεάζει σημαντικά τα δεδομένα είναι ο όρος correlation (συσχέτιση). Εννοιολογικά correlation είναι η μέτρηση του πόσο δύο μεταβλητές σχετίζονται μεταξύ τους. Μαθηματικά ο ορισμός αυτού του όρου για δείγματα δεδομένων δίνεται από το R του Pearson με τα αποτελέσματα να κυμαίνονται από [-1,1] και με τύπο:

$$r = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \overline{y})^2}}$$

Η πλήρης συσχέτιση μεταξύ δύο μεταβλητών εκφράζεται είτε με +1 είτε -1. Όταν μια μεταβλητή αυξάνεται καθώς και η άλλη αυξάνεται, ο συσχετισμός είναι θετικός.

Αντίθετα όταν η μία μειώνεται καθώς η άλλη αυξάνεται, είναι αρνητικός. Η πλήρης απουσία συσχέτισης αντιπροσωπεύεται από το 0.

Η παρούσα πτυχιακή αξιοποιεί τόσο αριθμητικές όσο και κατηγορικές μεταβλητές οι οποίες πρόκειται να αναλυθούν στο κομμάτι του κώδικα. Ο παραπάνω τύπος εφαρμόζεται σε αριθμητικές μεταβλητές. Η διαδικασία συσχέτισης των κατηγορικών μεταβλητών αλλάζει. Για να μπορέσει να εφαρμοστεί το correlation σε κατηγορικές μεταβλητές θα αξιοποιηθεί η συνάρτηση της Python Cramer's V. Η στατιστική Cramer's V είναι η μέτρηση της συσχέτισης μεταξύ δύο κατηγορικών μεταβλητών, η οποία δίνει τιμές μεταξύ του 0 και 1. Βασίζεται πάνω στην Pearson's x^2 statistic και δημοσιεύτηκε από τον Harald Cramer το 1946. Η μέθοδος αυτή λειτουργεί ως εξής. Έστω ένα δείγμα μεγέθους n με τις μεταβλητές A και B διανεμημένες από i=1,…,r και j=1,…,k όπου το n_{ij} ισούται με τον αριθμό που οι τιμές A_i και B_j έχουν παρατηρηθεί.Η μέθοδος Cramer's V δίνεται από τον τύπο

$$V = \sqrt{\frac{\varphi^2}{min(k-1, r-1)}} = \sqrt{\frac{\frac{x^2}{n}}{min(k-1, r-1)}}$$

όπου $x^2 = \sum_{i,j} \frac{(n_{i,j} - \frac{n_i n_j}{n})^2}{\frac{n_i n_j}{n}}$

και k ο αριθμός των στηλών, r ο αριθμός των γραμμών, n το σύνολο παρατηρήσεων, φ συντελεστής και x^2 από Pearson's chi-squared test. [24]

Κεφάλαιο 5: Ανάπτυξη Κώδικα

Εισαγωγή Δεδομένων

Το προγραμματιστικό περιβάλλον που αναπτύσσεται ο κώδικας την πτυχιακής είναι η Python ,μια ερμηνευμένη γλώσσα προγραμματισμού υψηλού επιπέδου και γενικού σκοπού. Αρχικά, όπως και σε κάθε άλλη γλώσσα προγραμματισμού, απαραίτητη είναι η εισαγωγή των κατάλληλων βιβλιοθηκών. Στη συνέχεια για τη διαχείριση και επεξεργασία των δεδομένων , εισάγονται με τη μορφή CSV file (comma separated values).

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

df = pd.read_csv('C:/Users/User/OneDrive/Υπολογιστής/clinvar.txt', dtype={0:
object, 38: str, 40: object})
df.head()
```

```
##   CHROM     POS REF ALT  ...  LoFtool  CADD_PHRED  CADD_RAW BLOSUM62
## 0     1  955563   G   C  ...    0.421      11.390  1.133255     -2.0
## 1     1  955597   G   T  ...    0.421       8.150  0.599088      NaN
## 2     1  955619   G   C  ...    0.421       3.288  0.069819      1.0
## 3     1  957640   C   T  ...    0.421      12.560  1.356499      NaN
## 4     1  976059   C   T  ...    0.421      17.740  2.234711      NaN
##
## [5 rows x 46 columns]
```

```
df.info()
```

```
## <class 'pandas.core.frame.DataFrame'>
## RangeIndex: 65188 entries, 0 to 65187
## Data columns (total 46 columns):
##  #   Column              Non-Null Count  Dtype
## ---  ------              --------------  -----
```

```
##  0   CHROM              65188 non-null  object
##  1   POS                65188 non-null  int64
##  2   REF                65188 non-null  object
##  3   ALT                65188 non-null  object
##  4   AF_ESP             65188 non-null  float64
##  5   AF_EXAC            65188 non-null  float64
##  6   AF_TGP             65188 non-null  float64
##  7   CLNDISDB           65188 non-null  object
##  8   CLNDISDBINCL       76 non-null     object
##  9   CLNDN              65188 non-null  object
##  10  CLNDNINCL          76 non-null     object
##  11  CLNHGVS            65188 non-null  object
##  12  CLNSIGINCL         76 non-null     object
##  13  CLNVC              65188 non-null  object
##  14  CLNVI              27659 non-null  object
##  15  MC                 58219 non-null  object
##  16  ORIGIN             59065 non-null  float64
##  17  SSR                104 non-null    float64
##  18  CLASS              65188 non-null  int64
##  19  Allele             65188 non-null  object
##  20  Consequence        65188 non-null  object
##  21  IMPACT             65188 non-null  object
##  22  SYMBOL             65172 non-null  object
##  23  Feature_type       65174 non-null  object
##  24  Feature            65174 non-null  object
##  25  BIOTYPE            65172 non-null  object
##  26  EXON               56295 non-null  object
##  27  INTRON             8803 non-null   object
##  28  cDNA_position      56304 non-null  object
##  29  CDS_position       55233 non-null  object
##  30  Protein_position   55233 non-null  object
##  31  Amino_acids        55184 non-null  object
##  32  Codons             55184 non-null  object
```

```
##  33  DISTANCE            108 non-null    float64
##  34  STRAND              65174 non-null  float64
##  35  BAM_EDIT            31969 non-null  object
##  36  SIFT                24836 non-null  object
##  37  PolyPhen            24796 non-null  object
##  38  MOTIF_NAME          2 non-null      object
##  39  MOTIF_POS           2 non-null      float64
##  40  HIGH_INF_POS        2 non-null      object
##  41  MOTIF_SCORE_CHANGE  2 non-null      float64
##  42  LoFtool             60975 non-null  float64
##  43  CADD_PHRED          64096 non-null  float64
##  44  CADD_RAW            64096 non-null  float64
##  45  BLOSUM62            25593 non-null  float64
## dtypes: float64(13), int64(2), object(31)
## memory usage: 22.9+ MB
```

Προ-επεξεργασία δεδομένων

Οι στήλες που εμφανίζονται παραπάνω είναι οι μεταβλητές που σχετίζονται με τις γενετικές μεταλλάξεις και πρέπει να επεξεργαστούν. Οι μεταβλητές αυτές χωρίζονται σε αριθμητικές και κατηγορικές. Οι αριθμητικές μεταβλητές είναι οι μεταβλητές των οποίων οι τιμές εκφράζουν μια μετρήσιμη ποσότητα. Αντίθετα, οι τιμές των κατηγορικών μεταβλητών είναι παρατηρήσεις οι οποίες κατανέμονται σε κατηγορίες. Ένα απαραίτητο βήμα για την εξαγωγή του επιθυμητού αποτελέσματος είναι η προ-επεξεργασία των δεδομένων , κατά την οποία ακολουθείται μια σειρά αφαίρεσης και τροποποίησης των παραπάνω μεταβλητών που περιέχουν θόρυβο ή έχουν ελλιπή δεδομένα. Έτσι, οι μεταβλητές που έχουν λιγότερες από 600 καταχωρήσεις αφαιρούνται αφού αποτελούν το 1% της βάσης δεδομένων.

```
df = df.drop(['CLNDISDBINCL', 'CLNDNINCL', 'CLNSIGINCL', 'SSR', 'DISTANCE',
              'MOTIF_NAME', 'MOTIF_POS', 'HIGH_INF_POS', 'MOTIF_SCORE_CHANGE'
], axis=1)
```

Είναι επίσης εμφανές ότι ένας αριθμός μεταβλητών διαθέτει καταχωρήσεις που είναι λιγότερες από τις μισές του συνόλου (<32.594).Αυτές είναι απαραίτητο να συμπληρωθούν με μηδενικές τιμές έτσι ώστε το σύνολο των καταχωρήσεών τους να είναι ανάλογο των υπολοίπων. Με αυτή τη διαδικασία θέτω με 1 το σύνολο των δεδομένων που υπάρχουν και με 0 το πλήθος των κενών τιμών.

```
for var in ['CLNVI', 'INTRON', 'BAM_EDIT', 'SIFT', 'PolyPhen', 'BLOSUM62']:
    df[var] = df[var].apply(lambda x: 1 if x == x else 0).astype('category')
    print(df[var].value_counts())
```

```
## 0    37529
## 1    27659
## Name: CLNVI, dtype: int64
## 0    56385
## 1     8803
## Name: INTRON, dtype: int64
## 0    33219
## 1    31969
## Name: BAM_EDIT, dtype: int64
## 0    40352
## 1    24836
## Name: SIFT, dtype: int64
## 0    40392
## 1    24796
## Name: PolyPhen, dtype: int64
## 0    39595
## 1    25593
## Name: BLOSUM62, dtype: int64
```

Class

Η μεταβλητή Class αποτελεί την σημαντικότερη στήλη αφού εξηγεί εάν υπάρχουν αντιφατικές παρατηρήσεις πάνω σε κάποια μετάλλαξη στην βάση δεδομένων. Με 1 να συμβολίζεται η ύπαρξη αντιφάσεων ενώ με 0 το αντίθετο.

```
df['CLASS'] = df['CLASS'].astype('category')
print(df['CLASS'].value_counts())

## 0    48754
## 1    16434
## Name: CLASS, dtype: int64
```

CHROM

Αυτή η μεταβλητή δείχνει την θέση της γενετικής παραλλαγής στο χρωμόσωμα και για το λόγο αυτό πρέπει να είναι κατηγορική μεταβλητή. Στο παρακάτω απόσπασμα κώδικα στην αριστερή στήλη εκφράζεται το χρωμόσωμα και στην δεξιά η θέση της μετάλλαξης. Για παράδειγμα η πρώτη γραμμή υποδεικνύει μια μετάλλαξη στη θέση 8645 του 2ου χρωμοσώματος.

```
print(df['CHROM'].value_counts())

## 2     8645
## 17    5394
## 11    4846
## 1     4454
## 16    3927
## 19    3911
## 5     3643
## 7     2867
## 12    2750
## 9     2744
## 13    2646
## 3     2529
```

```
## 10    2341
## 6     2159
## 14    1974
## X     1926
## 8     1852
## 15    1725
## 4     1206
## 22    1135
## 18     896
## 21     843
## 20     759
## MT      16
## Name: CHROM, dtype: int64


df['CHROM'] = df['CHROM'].astype('category')
```

POS

Αυτή η στήλη δείχνει την θέση του γονιδίου πάνω στο χρωμόσωμα επομένως εξαρτάται από την στήλη CHROM.

REF, ALT, Allele

Αυτές οι μεταβλητές προσδιορίζουν τα αλληλόμορφα γονίδια και προσδιορίζουν το πλήθος των τιμών αδενίνης (A), κυτοσίνης (C), γουανίνης (G) και θυμίνης (T), καθώς και τους συνδυασμούς τους, που υπάρχουν στα αλληλόμορφα.

```
for var in ['REF', 'ALT', 'Allele']:
    print(df[var].value_counts()[0:10])

## C     21798
## G     21361
## A      9845
## T      9421
```

```
## CT      126
## GC      113
## TG      105
## AG      104
## AC      103
## GA       91
## Name: REF, dtype: int64
## T     20409
## A     20205
## G     11782
## C     11429
## TA      118
## CT       93
## CA       77
## AT       75
## GA       67
## GT       64
## Name: ALT, dtype: int64
## T     19991
## A     19800
## G     11397
## C     10761
## -      2510
## AA       46
## TT       38
## AT       20
## CA       19
## CT       17
## Name: Allele, dtype: int64
```

Ορισμένες δεοξυριβονουκλεϊκές βάσεις ή αλληλουχία βάσεων παρατηρούνται σε αρκετά χαμηλή συχνότητα σε σύγκριση με τις υπόλοιπες και για το λόγο αυτό θα

δημιουργηθεί μια καινούργια κατηγορία «other» (O), που θα τις περιέχει. Έτσι πλέον υπάρχουν οι εξής πέντε κατηγορίες A, C, G, T και O.

```
for var in ['REF', 'ALT', 'Allele']:
    df[var] = df[var].apply(lambda x: 'O' if x not in ['A', 'C', 'G', 'T'] el
se
    x).astype('category')
```

AF_ESP, AF_EXAC, AF_TGP

Αυτές οι τρεις στήλες χαρακτηρίζουν την συχνότητα των αλληλομόρφων σε άλλες βάσεις δεδομένων γεγονός που τις χαρακτηρίζει χαμηλής σημασίας στην ανάπτυξη του μοντέλου και επομένως αφαιρούνται.

```
df = df.drop(['AF_ESP', 'AF_EXAC', 'AF_TGP'], axis=1)
```

CLNDISDB

Αυτή η στήλη διαθέτει τα ID ασθενειών από άλλες βάσεις δεδομένων. Επομένως αφαιρείται διότι δεν προσφέρει κάποια ουσιαστική πληροφορία για την συγκεκριμένη ανάλυση.

```
df = df.drop('CLNDISDB', axis=1)
```

CLNDN

Αποτελεί την προτίμηση της ClinVar στην ονομασία των ασθενειών γεγονός που την καθιστά σημαντική.

```
print(len(df['CLNDN'].unique()))

## 9260

print(df['CLNDN'].value_counts()[0:20])

## not_specified
5344
## Hereditary_cancer-predisposing_syndrome|not_specified
```

```
1724
## not_specified|not_provided
1398
## Hereditary_cancer-predisposing_syndrome
1139
## Limb-girdle_muscular_dystrophy,_type_2J|Dilated_cardiomyopathy_1G|not_spec
ified                913
## Familial_hypercholesterolemia
732
## Ataxia-telangiectasia_syndrome|Hereditary_cancer-predisposing_syndrome
608
## Hereditary_cancer-predisposing_syndrome|Familial_cancer_of_breast
561
## Ciliary_dyskinesia|not_specified
526
## Hereditary_cancer-predisposing_syndrome|Neurofibromatosis,_type_1
513
## Ataxia-telangiectasia_syndrome|Hereditary_cancer-predisposing_syndrome|not
_specified           469
## Hereditary_cancer-predisposing_syndrome|not_provided
456
## Hereditary_cancer-predisposing_syndrome|Familial_cancer_of_breast|not_spec
ified                450
## Hereditary_nonpolyposis_colon_cancer|Hereditary_cancer-predisposing_syndro
me                   435
## Hereditary_nonpolyposis_colon_cancer|Hereditary_cancer-predisposing_syndro
me|not_specified     397
## Hereditary_cancer-predisposing_syndrome|Hereditary_breast_and_ovarian_canc
er_syndrome          384
## Hereditary_cancer-predisposing_syndrome|Familial_adenomatous_polyposis_1
368
## Hereditary_cancer-predisposing_syndrome|Lynch_syndrome
360
```

```
## Hereditary_cancer-predisposing_syndrome|Rhabdoid_tumor_predisposition_synd
rome_2               338
## Hereditary_cancer-predisposing_syndrome|Familial_adenomatous_polyposis_1|n
ot_specified         271
## Name: CLNDN, dtype: int64
```

Παρατηρείται ότι ο όγκος των διαφορετικών ασθενειών είναι αρκετά μεγάλος και με σκοπό την καλύτερη διαχείριση του δημιουργούνται dummy μεταβλητές για τις πρώτες 100 ασθένειες. Στη συνέχεια αυτές οι μεταβλητές εισάγονται στη βάση δεδομένων ως διαφορετικές κατηγορίες και αντικαθιστούν την μεταβλητή CLNDN.

```
name_df = df['CLNDN'].str.split(pat='|', expand=True)
name_df.head()
```

```
##                                    0             1     2   ...    37    3
8    39
## 0  Myasthenic_syndrome,_congenital,_8  not_specified  None  ...  None  Non
e  None
## 1                      not_specified           None  None  ...  None  Non
e  None
## 2  Myasthenic_syndrome,_congenital,_8  not_specified  None  ...  None  Non
e  None
## 3  Myasthenic_syndrome,_congenital,_8  not_specified  None  ...  None  Non
e  None
## 4                      not_specified           None  None  ...  None  Non
e  None
##
## [5 rows x 40 columns]
```

```
top_100_dn = name_df.apply(pd.value_counts).sum(axis=1).sort_values(ascending
=False)[0:100]
print(top_100_dn[0:10])
```

```
## not_specified                                    45547.0
## Hereditary_cancer-predisposing_syndrome          18367.0
## not_provided                                     13020.0
## Hereditary_breast_and_ovarian_cancer_syndrome     2058.0
## Familial_cancer_of_breast                         1977.0
## Dilated_cardiomyopathy_1G                         1951.0
## Limb-girdle_muscular_dystrophy,_type_2J           1920.0
## Cardiovascular_phenotype                          1853.0
## Hypertrophic_cardiomyopathy                       1556.0
## Ataxia-telangiectasia_syndrome                    1510.0
## dtype: float64

top_100_dn_list = list(top_100_dn.index)


for dn in top_100_dn_list:
    df[dn] = df['CLNDN'].apply(lambda x: 1 if dn in x else 0).astype('categor
y')
df = df.drop('CLNDN', axis=1)
```

CLNHGVS

Αυτή η μεταβλητή αποτελείται από 65188 μοναδικές καταχωρήσεις οι οποίες περιέχουν τους κωδικούς για κάθε γονίδιο, έτσι όπως χαρακτηρίζονται από την θεσπισμένη γλώσσα HGVS nomenclature. Άρα θα αφαιρεθεί αφού δεν προσφέρει κάποια ουσιαστική βοήθεια στην ανάλυση.

```
print(len(df['CLNHGVS'].unique()))

## 65188

df = df.drop('CLNHGVS', axis=1)
```

CLNVC

Καταχωρεί τις μεταλλάξεις που έγιναν στο γενετικό υλικό.

```
print(df['CLNVC'].value_counts())

## single_nucleotide_variant    61281
## Deletion                      2509
## Duplication                   1034
## Indel                          247
## Insertion                       95
## Inversion                       17
## Microsatellite                   5
## Name: CLNVC, dtype: int64
```

Παρατηρείται ότι ο μεγαλύτερος όγκος τιμών συσσωρεύονται σε μια κατηγορία την single_nucleotide_variant. Επομένως η μεταβλητή αυτή μετατρέπεται σε κατηγορική με 4 κατηγορίες.

```
clnvc_types = ['single_nucleotide_variant', 'Deletion', 'Duplication']
df['CLNVC'] = df['CLNVC'].apply(lambda x: x if x in clnvc_types else 'Other')
.astype('category')
print(df['CLNVC'].value_counts())

## single_nucleotide_variant    61281
## Deletion                      2509
## Duplication                   1034
## Other                          364
## Name: CLNVC, dtype: int64
```

Οι τρεις πρώτες κατηγορίες παραμένουν ίδιες ενώ οι υπόλοιπες κατατάσσονται σε μία καινούργια κατηγορία «other» (O).

MC

Η πλήρης ονομασία της μεταβλητής αυτής είναι Molecular consequence, δηλαδή μοριακές συνέπειες, άρα αποτελεί μια κατηγορική μεταβλητή.

```
print(df['MC'].value_counts()[0:10])
```

```
## SO:0001583|missense_variant                                        23034
## SO:0001819|synonymous_variant                                      16549
## SO:0001627|intron_variant                                           7534
## SO:0001583|missense_variant,SO:0001627|intron_variant               2422
## SO:0001589|frameshift_variant                                       1622
## SO:0001587|nonsense                                                 1573
## SO:0001627|intron_variant,SO:0001819|synonymous_variant             1148
## SO:0001583|missense_variant,SO:0001623|5_prime_UTR_variant           599
## SO:0001623|5_prime_UTR_variant                                       516
## SO:0001575|splice_donor_variant                                      504
## Name: MC, dtype: int64
```

Όπως και σε προηγούμενη μεταβλητή με σκοπό την καλύτερη διαχείριση των δεδομένων της MC δημιουργούνται dummies μεταβλητές για κάθε μία μοριακή συνέπεια, ενώ με την εντολή fillna αντικαθίστανται οι κενές τιμές ,NA/NAN values, με «unknown».

```
name_df = df['MC'].str.split(pat='[|,]', expand=True)
name_df.head()
```

```
##             0                    1     2     3     4     5     6     7
## 0  SO:0001583    missense_variant  None  None  None  None  None  None
## 1  SO:0001819  synonymous_variant  None  None  None  None  None  None
## 2  SO:0001583    missense_variant  None  None  None  None  None  None
## 3  SO:0001819  synonymous_variant  None  None  None  None  None  None
## 4  SO:0001819  synonymous_variant  None  None  None  None  None  None
```

```
top_mc = name_df.apply(pd.value_counts).sum(axis=1).sort_values(ascending=Fal
se)[0:20]
print(top_mc)
```

```
## missense_variant           26642.0
## SO:0001583                 26642.0
```

```
## synonymous_variant         18379.0
## SO:0001819                 18379.0
## intron_variant             11967.0
## SO:0001627                 11967.0
## 5_prime_UTR_variant         1985.0
## SO:0001623                  1985.0
## SO:0001589                  1861.0
## frameshift_variant          1861.0
## nonsense                    1783.0
## SO:0001587                  1783.0
## 3_prime_UTR_variant          704.0
## SO:0001624                   704.0
## splice_donor_variant         579.0
## SO:0001575                   579.0
## SO:0001636                   429.0
## 2KB_upstream_variant         429.0
## SO:0001574                   428.0
## splice_acceptor_variant      428.0
## dtype: float64
```

```
top_mc_list = [x for x in list(top_mc.index) if 'SO:' not in x]


df['MC'] = df['MC'].fillna('unknown')
for mc in top_mc_list:
    df[mc] = df['MC'].apply(lambda x: 1 if mc in x else 0).astype('category')
    print(df[mc].value_counts())
```

```
## 0    38546
## 1    26642
## Name: missense_variant, dtype: int64
## 0    46809
## 1    18379
## Name: synonymous_variant, dtype: int64
## 0    53221
```

```
## 1    11967
## Name: intron_variant, dtype: int64
## 0    63203
## 1     1985
## Name: 5_prime_UTR_variant, dtype: int64
## 0    63327
## 1     1861
## Name: frameshift_variant, dtype: int64
## 0    63405
## 1     1783
## Name: nonsense, dtype: int64
## 0    64484
## 1      704
## Name: 3_prime_UTR_variant, dtype: int64
## 0    64609
## 1      579
## Name: splice_donor_variant, dtype: int64
## 0    64759
## 1      429
## Name: 2KB_upstream_variant, dtype: int64
## 0    64760
## 1      428
## Name: splice_acceptor_variant, dtype: int64

df = df.drop('MC', axis=1)
```

ORIGIN

Η συγκεκριμένη μεταβλητή δείχνει την προέλευση του αλληλόμορφου και δέχεται μια ή παραπάνω από τις τιμές 0 – unknown, 1 – germline, 2 – somatic, 4 – inherited, 8 – paternal, 16 – maternal, 32 - de-novo, 64 – biparental, 128 – uniparental, 256 - not-tested, 512 - tested-inconclusive, 1073741824 – other.

```
print(df['ORIGIN'].value_counts())
```

```
## 1.0      58216
## 33.0       246
## 5.0        149
## 3.0        142
## 17.0       139
## 9.0        100
## 0.0         13
## 25.0        11
## 32.0        10
## 49.0         7
## 513.0        5
## 65.0         4
## 2.0          3
## 21.0         3
## 13.0         2
## 41.0         2
## 16.0         2
## 4.0          2
## 129.0        1
## 27.0         1
## 43.0         1
## 35.0         1
## 29.0         1
## 37.0         1
## 12.0         1
## 69.0         1
## 53.0         1
## Name: ORIGIN, dtype: int64
```

Είναι φανερό ότι το μεγαλύτερο ποσοστό ανήκει στην κατηγορία 1, άρα όλες οι άλλες τιμές θα καταχωρηθούν σε μία καινούρια dummy μεταβλητή ως 0. Επίσης, με την εντολή fillna οι κενές τιμές αντικαθίστανται με 0.

```
df['ORIGIN'] = df['ORIGIN'].fillna(0).apply(lambda x: 1 if x == 1.0 else 0).a
stype('category')
```

CONSEQUENCE

Είναι εμφανές ότι αυτή η μεταβλητή είναι παρόμοια με την MC με μικρές διαφορές στις τιμές και για το λόγο αυτό κρίνεται πως πρέπει να αφαιρεθεί αφού τα δεδομένα της υπάρχουν ήδη μια φορά.

```
df = df.drop('Consequence', axis=1)
```

IMPACT

Αυτή η μεταβλητή εκφράζει την επίδραση της μετάλλαξης και επομένως είναι κατηγορική.

```
print(df['IMPACT'].value_counts())

## MODERATE    33212
## LOW         21642
## MODIFIER     5582
## HIGH         4752
## Name: IMPACT, dtype: int64

df['IMPACT'] = df['IMPACT'].astype('category')
```

SYMBOL

Η μεταβλητή SYMBOL περιέχει το όνομα του γονιδίου με αποτέλεσμα να υπάρχουν πολλές τιμές.

```
print(len(df['SYMBOL'].unique()))

## 2329
```

Έτσι μετατρέπεται σε κατηγορική μεταβλητή για τις πρώτες 100 τιμές και οι υπόλοιπες κατατάσσονται σε μια επιπλέον κατηγορία «Other».

```
top_100_symb = df['SYMBOL'].value_counts()[0:100].index
df['SYMBOL'] = df['SYMBOL'].apply(lambda x: x if x in top_100_symb
else 'Other').astype('category')
print(df['SYMBOL'].value_counts()[0:100])

## Other      30800
## TTN         2765
## BRCA2       1934
## ATM         1909
## APC         1228
##              ...
## CNTNAP2      123
## VPS13B       121
## FH           118
## NOTCH1       118
## ALMS1        117
## Name: SYMBOL, Length: 100, dtype: int64
```

FEATURE

Αυτή η μεταβλητή αποτελεί μια ακόμα ταυτοποίηση για το όνομα των γονιδίων επομένως αφαιρείται εφόσον υπάρχει ήδη μεταβλητή που εξυπηρετεί το σκοπό αυτό.

```
df = df.drop('Feature', axis=1)
```

Feature_type, BIOTYPE

Οι δύο αυτές στήλες έχουν μια ίδια τιμή. Έτσι αφαιρούνται.

```
for var in ['Feature_type', 'BIOTYPE']:
    print(df[var].value_counts())
    df = df.drop(var, axis=1)
```

```
## Transcript      65172
## MotifFeature        2
## Name: Feature_type, dtype: int64
## protein_coding    65158
## misc_RNA             14
## Name: BIOTYPE, dtype: int64
```

EXON

Η μεταβλητή αυτή περιέχει τα εξώνια των γονιδίων. Αποτελείται από διαφορετικές τιμές και επομένως αφαιρείται αφού δεν προσφέρει κάποια ουσιαστική πληροφορία για το μοντέλο που δημιουργείται.

```
print(len(df['EXON'].unique()))

## 3265

df = df.drop('EXON', axis=1)
```

cDNA_position, CDS_position, Protein_position

Αυτές οι μεταβλητές δεν προβάλουν χρήσιμα δεδομένα διότι αναφέρονται στην θέση του cDNA, coding sequence και της πρωτεΐνης άρα αφαιρούνται.

```
df = df.drop(['cDNA_position', 'CDS_position', 'Protein_position'], axis=1)
```

Amino_acids, Codons

Η μεταβλητή Amino_acids δίνεται μόνο όταν η μετάλλαξη επηρεάζει την ακολουθία πρωτεΐνης. Η στήλη Codons περιέχει τα κωδικόνια. Και οι δύο μεταβλητές αποτελούνται από μεγάλο πλήθος διαφορετικών τιμών χωρίς όμως να προσφέρουν ουσιαστική σημασία στην ανάλυση οπότε αφαιρούνται.

```
df = df.drop(['Amino_acids', 'Codons'], axis=1)
```

STRAND

Μεταβλητή που καθορίζει την αλυσίδα του DNA και χωρίζεται στις κατηγορίες +(forward) και -(reverse), άρα αποτελεί μέρος των κατηγορικών μεταβλητών.

```
print(df['STRAND'].value_counts())

## -1.0    32804
##  1.0    32370
## Name: STRAND, dtype: int64

df['STRAND'] = df['STRAND'].astype('category')
```

LoFtool

Αριθμητική μεταβλητή που αντικατοπτρίζει το σκορ των μη λειτουργικών μεταλλάξεων. Η στήλη αυτή διαθέτει κάποιες κενές τιμές οι οποίες θα αντικατασταθούν με την διάμεσο (median) του συνόλου των τιμών.

```
df['LoFtool'] = df['LoFtool'].fillna(df['LoFtool'].median())
```

CADD_PHRED, CADD_RAW

Οι συγκεκριμένες μεταβλητές απεικονίζουν το Cadd Score και την βλαβερότητα της μετάλλαξης αντίστοιχα. Αποτελούν παρόμοιες μεταβλητες με την LoFtool και για τον λόγο αυτό ακολουθείται η ίδια διαδικασία.

```
df['CADD_PHRED']     =     df['CADD_PHRED'].fillna(df['CADD_PHRED'].median())
df['CADD_RAW'] = df['CADD_RAW'].fillna(df['CADD_RAW'].median())
```

Καταλήγοντας η κατάσταση του αρχείου με τα δεδομένα μου είναι η εξής:

```
print(df.info())

## <class 'pandas.core.frame.DataFrame'>
## RangeIndex: 65188 entries, 0 to 65187
## Columns: 130 entries, CHROM to splice_acceptor_variant
## dtypes: category(126), float64(3), int64(1)
```

```
## memory usage: 9.8 MB
## None
```

Κατηγοριοποίηση Μεταβλητών

Με σκοπό την κατηγοριοποίηση των μεταβλητών που αναλύθηκαν παραπάνω η βάση δεδομένων χωρίζεται σε δύο καινούργιους πίνακες. Ο πρώτος πίνακας θα αποτελείται από μια μοναδική μεταβλητή, την CLASS μετονομασμένη σε target, και ο δεύτερος πίνακας θα περιέχει τις υπόλοιπες μεταβλητές. Με τον τρόπο αυτό επιτυγχάνεται η σωστή εφαρμογή των αλγορίθμων.

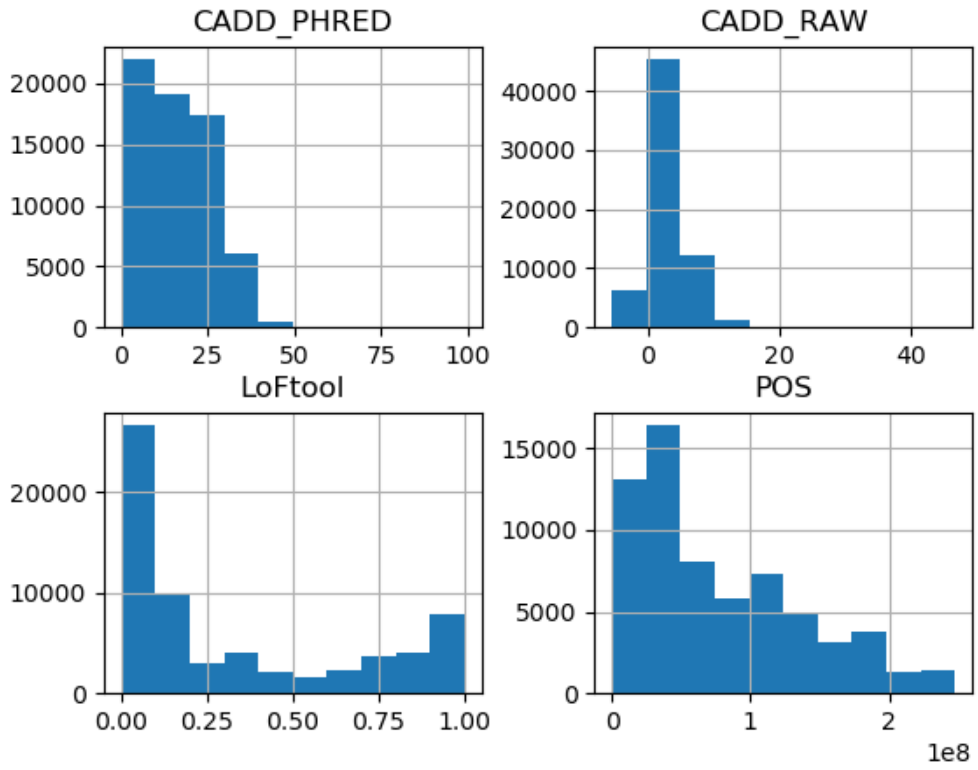
```
df = df.rename({'CLASS': 'target'}, axis=1)
df['target'] = df['target'].astype('category')

target = df['target']
features = df.drop('target', axis=1)
```

Αριθμητικές Μεταβλητές

Ο πίνακας features περιέχει 4 μεταβλητές με αριθμητικά δεδομένα οι οποίες κανονικοποιούνται χρησιμοποιώντας το StandarScaler από scikitlearn. Η διαδικασία ξεκινάει θέτοντας μια λίστα num_var_list που περιέχει τις στήλες με τα αριθμητικά δεδομένα. Έπειτα εισάγεται η απαραίτητη βιβλιοθήκη, StandarScaler, που εφαρμόζεται στη λίστα που δημιουργήθηκε.

```
from sklearn.preprocessing import StandardScaler

num_var_list = ['POS', 'LoFtool', 'CADD_PHRED', 'CADD_RAW']
scl = StandardScaler()
df[num_var_list] = scl.fit_transform(df[num_var_list])
```

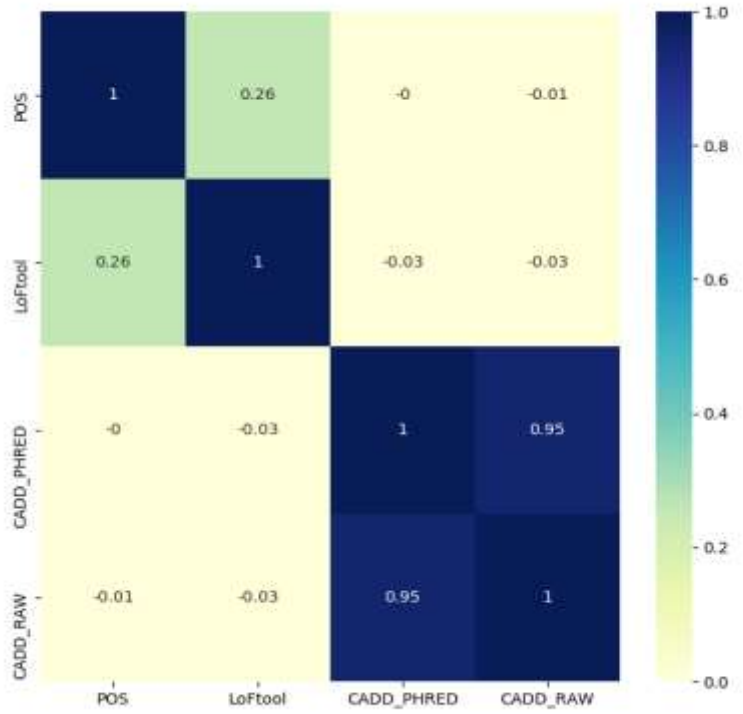
```
print(features[num_var_list].describe())
print(features[num_var_list].hist())
```



Εικόνα 7: Κανονικοποίηση αριθμητικών μεταβλητών

Αφού κανονικοποιήθηκαν οι αριθμητικές μεταβλητές θα ήταν ορθό να βρεθεί η μεταξύ τους συσχέτιση (correlation). Για τον σκοπό αυτό δημιουργείται ένα heatmap.

```
plt.figure(figsize=(8, 8))
sns.heatmap(features[num_var_list].corr(),
            vmin=0,
            vmax=1,
            cmap='YlGnBu',
            annot=np.round(features[num_var_list].corr(), 2))
plt.show()
```



Εικόνα 8: Heatmap αριθμητικών μεταβλητών

Είναι εμφανές ότι μεταξύ των CADD_RAW και CADD_PHRED μεταβλητών η συσχέτιση είναι αρκετά υψηλή (πολύ κοντά στο 1), πράγμα που σημαίνει ότι και οι δύο μεταβλητές δίνουν την ίδια πληροφορία επομένως εάν αφαιρεθεί μία από αυτές το αποτέλεσμα δεν θα αλλοιωθεί. Λόγω της χαμηλής τυπικής απόκλισης (STD) επιλέγεται να αφαιρεθεί η μεταβλητή CADD_RAW.

```
features = features.drop('CADD_RAW', axis=1)
```

Κατηγορικές Μεταβλητές

Πρώτα ορίζεται μια λίστα που περιέχει μόνο τις κατηγορικές μεταβλητές.

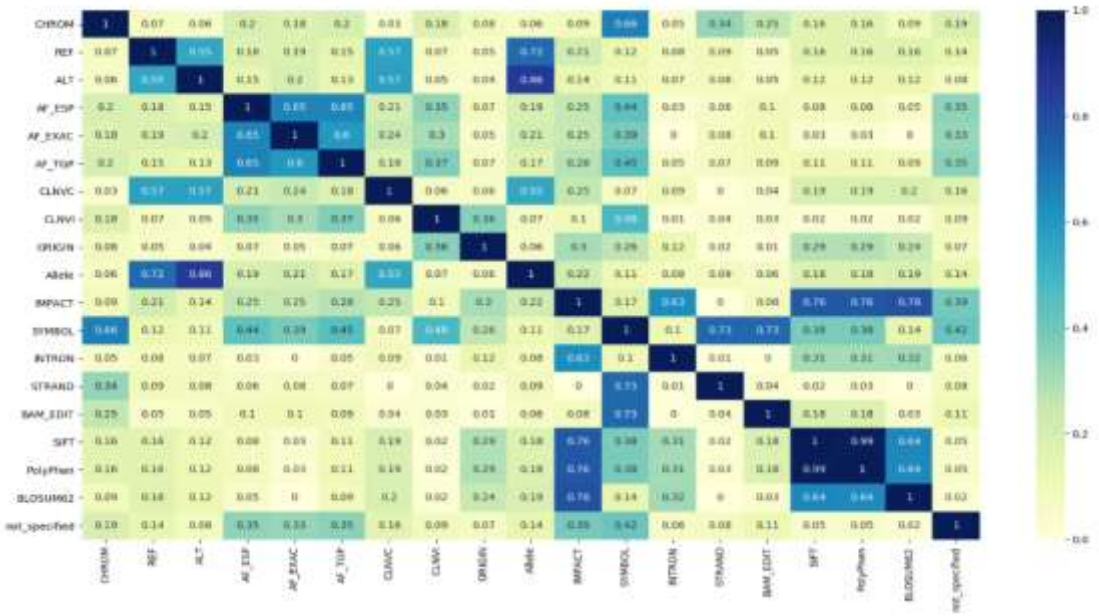
```
orig_feat = list(features.columns[0:22])
orig_feat_cat = [x for x in orig_feat if x not in num_var_list]
```

Στη συνέχεια εφαρμόζεται η μέθοδος Crammer's V για την εύρεση της μεταξύ τους συσχέτισης.

```
import scipy.stats as ss



def cramers_v(x, y):
    confusion_matrix = pd.crosstab(x, y)
    chi2 = ss.chi2_contingency(confusion_matrix)[0]
    n = confusion_matrix.sum().sum()
    phi2 = chi2 / n
    r, k = confusion_matrix.shape
    phi2corr = max(0, phi2 - ((k - 1) * (r - 1)) / (n - 1))
    rcorr = r - ((r - 1) ** 2) / (n - 1)
    kcorr = k - ((k - 1) ** 2) / (n - 1)
    return np.sqrt(phi2corr / min((kcorr - 1), (rcorr - 1)))



num_feat = len(orig_feat_cat)
cat_corr_arr = np.empty((num_feat, num_feat))
for i, row in enumerate(orig_feat_cat):
    for j, col in enumerate(orig_feat_cat):
        cat_corr_arr[i, j] = cramers_v(features[row], features[col])
print(cat_corr_arr[0:5, 0:5])
```

Τέλος, όπως και στις αριθμητικές μεταβλητές, δημιουργείται το heatmap.

```
plt.figure(figsize=(16, 14))
sns.heatmap(cat_corr_arr,
            vmin=0,
            vmax=1,
            cmap='YlGnBu',
            xticklabels=orig_feat_cat,
            yticklabels=orig_feat_cat,
            annot=np.round(cat_corr_arr, 2))
```



Εικόνα 9: Heatmap κατηγορικών μεταβλητών

Παρατηρώντας το διάγραμμα γίνεται φανερό ότι η μεταβλητή Allele εμφανίζει μεγάλο ποσοστό συσχέτισης με την ALT(0,86),η IMPACT με την BLOSUM62 (0,78), η SYMBOL με την STRAND(0,73) και η PolyPhen με την SIFT (0.99) και έτσι αφαιρούνται οι Allele, IMPACT , SYMBOL και PolyPhen .

```
features = features.drop(['Allele', 'IMPACT', 'SYMBOL', 'PolyPhen'], axis=1)
```

Εφαρμογή Αλγορίθμων

Στην ενότητα αυτή θα εφαρμοστούν στα δεδομένα οι αλγόριθμοι ταξινόμησης που αναπτύχθηκαν παραπάνω. Με σκοπό την αξιολόγηση τους και την εύρεση του αποδοτικότερου αλγόριθμου θα εφαρμοστεί το μέτρο απόδοσης F1 score για τον καθένα από αυτούς.

Dummy Classifier

Πρώτα εφαρμόζεται ο Dummy Classifier, διότι αποτελεί ένα απλό αλγόριθμο που χρησιμοποιείται ως μέτρο σύγκρισης για άλλους αλγορίθμους.

```
from sklearn.dummy import DummyClassifier
from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier
from sklearn.naive_bayes import BernoulliNB
from sklearn.model_selection import train_test_split,

from sklearn.metrics import classification_report,
from sklearn.neighbors import KNeighborsClassifier


features = pd.get_dummies(features, drop_first=True)
print(features.columns)

## Index(['POS', 'LoFtool', 'CADD_PHRED', 'CHROM_10', 'CHROM_11', 'CHROM_12',
##        'CHROM_13', 'CHROM_14', 'CHROM_15', 'CHROM_16',
##        ...
##        'missense_variant_1', 'synonymous_variant_1', 'intron_variant_1',
##        '5_prime_UTR_variant_1', 'frameshift_variant_1', 'nonsense_1',
##        '3_prime_UTR_variant_1', 'splice_donor_variant_1',
##        '2KB_upstream_variant_1', 'splice_acceptor_variant_1'],
##       dtype='object', length=154)
```

```
X_train, X_test, y_train, y_test = train_test_split(features, target, test_si
ze=0.33, random_state=42)
# Dummy Classifier
m_clf = DummyClassifier(random_state=42)

dm_clf.fit(X_train, y_train)

dm_pred = dm_clf.predict(X_test)

print("Dummy Classifier ")

print(classification_report(y_test, dm_pred))

## Dummy Classifier

##               precision    recall  f1-score   support


##           0       0.75      0.75      0.75     16160

##           1       0.26      0.26      0.26      5353


##    accuracy                           0.63     21513

##   macro avg       0.51      0.51      0.51     21513

## weighted avg      0.63      0.63      0.63     21513
```

Random Forest

Στη συνέχεια εφαρμόζεται ο αλγόριθμος Random Forest. Η παράμετρος n_estimator εκφράζει τον αριθμό των δέντρων που θα δημιουργηθούν.

```
rf_clf = RandomForestClassifier(n_estimators=100, random_state=42)
rf_clf.fit(X_train, y_train)

rf_pred = rf_clf.predict(X_test)
```

```
print("Random Forest")
print(classification_report(y_test, rf_pred))
```

```
## Random Forest
```

```
##               precision    recall  f1-score   support
```

```
##            0       0.80      0.94      0.86     16160
```

```
##            1       0.59      0.27      0.37      5353
```

```
##     accuracy                           0.77     21513
```

```
##    macro avg       0.69      0.61      0.62     21513
```

```
## weighted avg       0.74      0.77      0.74     21513
```

K Neighbors Classifier

Επόμενος στη σειρά είναι ο ταξινομητής KNN.

```
knn = KNeighborsClassifier()
knn.fit(X_train, y_train)

KNN_pred = knn.predict(X_test)

print("K Neighbors")
print(classification_report(y_test, KNN_pred))
```

```
## K Neighbors
```

```
##               precision    recall  f1-score   support
```

```
##            0       0.77      0.88      0.82     16160
```

```
##            1       0.34      0.19      0.24      5353

##     accuracy                           0.71     21513
##    macro avg       0.55      0.53      0.53     21513
## weighted avg       0.66      0.71      0.68     21513
```

Bernoulli Naive Bayes Classifier

Τέλος εφαρμόζεται ο αλγόριθμος Bernoulli Naive Bayes Classifier.

```
bnb_clf = BernoulliNB()
bnb_clf.fit(X_train, y_train)
bnb_pred = bnb_clf.predict(X_test)



print("Bernoulli")
print(classification_report(y_test, bnb_pred))
```

```
Bernoulli
##               precision    recall  f1-score   support

##            0       0.78      0.79      0.78     16160
##            1       0.33      0.32      0.33      5353
```

```
##     accuracy                           0.67     21513
##    macro avg       0.55      0.55      0.55     21513
## weighted avg       0.67      0.67      0.67     21513
```

Απόδοση Αλγορίθμων

Όπως φαίνεται στον παραπάνω κώδικα, εμφανίζονται δύο αποτελέσματα για κάθε μέτρο απόδοσης. Αυτό οφείλεται στο γεγονός ότι τα μέτρα απόδοσης έχουν υπολογιστεί με βάση τη μεταβλητή Class, η οποία χωρίζεται σε δύο κατηγορίες, την 0 και 1. Η κατηγορία 0 αναπαριστά τα δεδομένα που δεν εμφανίζουν απόκλιση στις καταχωρήσεις των εργαστηρίων ενώ η κατηγορία 1 αυτά που εμφανίζουν απόκλιση. Για την εύρεση του αποτελεσματικότερου αλγόριθμου ταξινόμησης θα συγκριθούν τα αποτελέσματα της κατηγορίας 1 των μέτρων απόδοσης, καθώς το ζητούμενο είναι πόσο καλά μπορεί ο κάθε αλγόριθμος να προβλέψει τις αποκλίσεις.

Αλγόριθμοι	F1 - Score	Precision	Recall	Accuracy
Dummy Classifier	0.26	0.26	0.26	0.63
Random Forest Classifier	0.37	0.59	0.27	0.77
K- Neighbors Classifier	0.24	0.34	0.19	0.24
Bernoulli Naive Bayes Classifier	0.33	0.33	0.32	0.67

Πίνακας 1: Πίνακας μέτρων απόδοσης αλγορίθμων με βάση την κατηγορία 1

Αλγόριθμοι	F1 - Score	Precision	Recall	Accuracy
Dummy Classifier	0.75	0.75	0.75	0.63
Random Forest Classifier	0.86	0.80	0.94	0.77
K- Neighbors Classifier	0.82	0.77	0.88	0.24
Bernoulli Naive Bayes Classifier	0.78	0.78	0.79	0.67

Πίνακας 2: Πίνακας μέτρων απόδοσης αλγορίθμων με βάση την κατηγορία 0

Συμπεράσματα

Η τεχνητή νοημοσύνη είναι ένας τομέας που εμφανίζει ραγδαία ανάπτυξη τα τελευταία χρόνια καθώς αποτελεί ένα από τα πιο νέα ερευνητικά πεδία. Στην προσέγγιση της τεχνητής νοημοσύνης με τους νόμους της σκέψης, η έμφαση δινόταν στην σωστή εξαγωγή των συμπερασμάτων. Με αυτό τον τρόπο ,έχει σαν σκοπό την μελέτη ιδεών που επιτρέπουν σε ένα υπολογιστή να φαίνεται νοήμων. Μια από τις πιο σημαντικές εφαρμογές της τεχνητής νοημοσύνης είναι η ανάλυση δεδομένων μεγάλου όγκου. Η ανάλυση δεδομένων μεγάλου όγκου μπορεί να χρησιμοποιηθεί σε κάθε περίπτωση που απαιτείται εξαγωγή συμπεράσματος από μεγάλο πλήθος δεδομένων. Γίνεται λοιπόν εύκολα αντιληπτό , ότι λόγω του μεγάλου όγκου του ανθρώπινου γονιδιώματος , είναι απαραίτητη η χρήση της τεχνητής νοημοσύνης και συγκεκριμένα της ανάλυσης και εξόρυξης δεδομένων μεγάλου όγκου με σκοπό να εξαχθούν συμπεράσματα για το ανθρώπινο γονιδίωμα και τις μεταλλάξεις του. Απαραίτητα εργαλεία γι' αυτήν τη διαδικασία αποτελούν οι αλγόριθμοι της τεχνητής νοημοσύνης, οι οποίοι έχουν διαφορετική αποτελεσματικότητα ανάλογα με το μοντέλο που αναπτύσσεται. Για το λόγο αυτό κρίνεται αναγκαία η σύγκριση της αποτελεσματικότητας

κάθε αλγορίθμου που χρησιμοποιήθηκε με βάση τα μέτρα απόδοσης του καθενός έτσι ώστε να βρεθεί ο αποτελεσματικότερος εξ΄ αυτών. Στο παραπάνω μοντέλο που αναπτύχθηκε υπολογίστηκαν ξεχωριστά τα μέτρα απόδοσης κάθε αλγόριθμου. Πιο συγκεκριμένα για τον αλγόριθμο Dummy Classifier βρέθηκαν τα precision =0.26, recall =0.26, accuracy =0.63 , F1-score = 0.26 , για τον αλγόριθμο Random Forest Classifier υπολογίστηκαν αντίστοιχα τα precision = 0.59, recall =0.27, accuracy =0.77 , F1- score =0.37 και για τον αλγόριθμο K-Neighbors Classifier τα precision = 0.34 , recall = 0.19 , accuracy=0.24 , F1- score =0.24 . Επιπλέον ο αλγόριθμος Bernoulli Naive Bayes Classifier έδωσε αποτελέσματα precision = 0.33 , recall =0.32 , accuracy= 0.67, F1- score =0.33 . Από τα παραπάνω επιλέχθηκε η σύγκριση για την εύρεση του αποτελεσματικότερου αλγόριθμου ταξινόμησης να γίνει με βάση το F1 – score των αλγορίθμων καθώς το F1-score είναι ο σταθμισμένος μέσος όρος Precision και Recall και επομένως λαμβάνει υπόψη τόσο ψευδώς θετικές όσο και ψευδώς αρνητικές προβλέψεις. Γίνεται αντιληπτό λοιπόν, πως την καλύτερη απόδοση με βάση το μέτρο F1 – score είχε ο αλγόριθμος Random Forest Classifier γεγονός που υποδεικνύει πως επιτυγχάνει την καλύτερη πρόβλεψη των αντιφατικών καταχωρήσεων των μεταλλάξεων ενώ αντίστοιχα την χειρότερη απόδοση κατέχει ο αλγόριθμος KNN Classifier με F1 – score =0.24.

Βιβλιογραφική Αναφορά

[1] J. McCarthy, «What is Artificial Intelligence,» 2007. [Ηλεκτρονικό]. Available: http://jmc.stanford.edu/articles/whatisai/whatisai.pdf.

[2] P. Norvig και S. Russel, «Τεχνητή νοημοσύνη, μια σύγχρονη προσέγγιση,» , Β' έκδοση, επιμ., Εκδόσεις Κλειδάριθμος, 2005.

[3] Wikipedia, «History of artificial intelligence,» [Ηλεκτρονικό]. Available: https://en.wikipedia.org/wiki/History_of_artificial_intelligence.

[4] B. Buchanan, « A (Very) Brief History of Artificial Intelligence,» 2005. [Ηλεκτρονικό]. Available: https://web.archive.org/web/20070926023314/http://www.aaai.org/AITopics/assets/PDF/AIMag26-04-016.pdf.

[5] Ι. Βλαχάβας, Π. Κεφαλάς, Ν. Βασιλειάδης, Φ. Κόκκορας και Η. Σακελλαρίου, Τεχνητή Νοημοσύνη, Εκδόσεις Πανεπιστημίου Μακεδονίας, 2011.

[6] R. T. Dias, «Artificial intelligence in clinical and genomic diagnostics. Genome Med 11, 70,» 2019. [Ηλεκτρονικό]. Available: https://doi.org/10.1186/s13073-019-0689-8A.

[7] C. Clifton, «Data mining,» 2009. [Ηλεκτρονικό]. Available: https://www.britannica.com/technology/data-mining.

[8] R. Li, «History of Data Mining,» 2016. [Ηλεκτρονικό]. Available: https://www.kdnuggets.com/2016/06/rayli-history-data-mining.html.

[9] U. Fayyad, G. Piatetsky-Shapiro και P. Smyth, «From Data Mining to Knowledge Discovery in Databases,» 1996. [Ηλεκτρονικό]. Available: https://www.kdnuggets.com/gpspubs/aimag-kdd-overview-1996-Fayyad.pdf.

[10] . Π. Μποζάνη, Δομές δεδομένω, Εκδόσεις Τζιόλα, 2006.

[11] scikit-learn, «LinearRegression,» [Ηλεκτρονικό]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html?highlight=linear%20regression#sklearn.linear_model.LinearRegression.

[12] scikit-learn, «Lasso,» [Ηλεκτρονικό]. Available: https://scikit-learn.org/stable/modules/linear_model.html#lasso.

[13] scikit-learn, «LogisticRegression,» [Ηλεκτρονικό]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html?highlight=logistic%20regression#sklearn.linear_model.LogisticRegression.

[14] scikit-learn, «K-means,» [Ηλεκτρονικό]. Available: https://scikit-learn.org/stable/modules/clustering.html#k-means.

[15] scikit-learn, «Mean Shift,» [Ηλεκτρονικό]. Available: https://scikit-learn.org/stable/modules/clustering.html#mean-shift.

[16] scikit-learn, «Hierarchical clustering,» [Ηλεκτρονικό]. Available: https://scikit-learn.org/stable/modules/clustering.html#hierarchical-clustering.

[17] T. Yiu, «Understanding Random Forest,» 2019. [Ηλεκτρονικό]. Available: https://towardsdatascience.com/understanding-random-forest-58381e0602d2.

[18] O. Harrison, «Machine Learning Basics with the K-Nearest Neighbors Algorithm,» [Ηλεκτρονικό]. Available: https://towardsdatascience.com/understanding-random-forest-58381e0602d2.

[19] R. Gandhi, «Naive Bayes Classifier,» 2018. [Ηλεκτρονικό]. Available: https://towardsdatascience.com/naive-bayes-classifier-81d512f50a7c.

[20] M. Zaki και W. J. Meira, Εξόρυξη και ανάλυση δεδομένων, Εκδόσεις Κλειδάριθμος, 2017.

[21] K. P. Shung, «Accuracy, Precision, Recall or F1?,» 2018. [Ηλεκτρονικό]. Available: https://towardsdatascience.com/accuracy-precision-recall-or-f1-331fb37c5cb9.

[22] M. Landrum, J. Lee, M. Benson, G. Brown, C. Chao, S. Chitipiralla, B. Gu, J. Hart, D. Hoffman, W. Jang, K. Karapetyan, K. Katz, C. Liu, Z. Maddipatla, A. Malheiro, K. McDaniel, M. Ovetsky, G. Riley, G. Zhou, J. B. Holmes, B. Kattman και D. Maglott, «ClinVar: improving access to variant interpretations and supporting evidence,» 2018. [Ηλεκτρονικό]. Available: https://academic.oup.com/nar/article/46/D1/D1062/4641904.

[23] M. Landrum, J. Lee, G. Riley, W. Jang, W. Rubinstein, D. Church και D. Maglott, «ClinVar: public archive of relationships among sequence variation and human phenotype,» 2014. [Ηλεκτρονικό]. Available: https://doi.org/10.1093/nar/gkt1113.

[24] S. Zychlinski, «The Search for Categorical Correlation,» 2018. [Ηλεκτρονικό]. Available: https://towardsdatascience.com/the-search-for-categorical-correlation-a1cf7f1888c9.