ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΤΜΗΜΑ ΕΦΑΡΜΟΣΜΕΝΩΝ ΜΑΘΗΜΑΤΙΚΩΝ ΚΑΙ ΦΥΣΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

Μηχανική Μάθηση για τη Διάγνωση του Διαβήτη

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Αντώνιος Μήτσης

Επιβλέπων: Πέτρος Στεφανέας
Αναπληρωτής Καθηγητής Ε.Μ.Π.

Αθήνα, Σεπτέμβριος 2024

……

Αντώνιος Μήτσης

ΠΕΡΙΛΗΨΗ

Η παρούσα διατριβή εξετάζει τη χρήση τεχνικών μηχανικής μάθησης για την πρόβλεψη του κινδύνου εμφάνισης διαβήτη, χρησιμοποιώντας κλινικά και δημογραφικά δεδομένα. Χρησιμοποιήθηκαν τόσο επιβλεπόμενες όσο και μη επιβλεπόμενες μέθοδοι μηχανικής μάθησης για την ανάπτυξη μοντέλων που μπορούν να εντοπίσουν άτομα με υψηλό κίνδυνο. Κύριοι επιβλεπόμενοι αλγόριθμοι, όπως η Λογιστική Παλινδρόμηση, το Random Forest και οι Support Vector Machines (SVM), υλοποιήθηκαν και συγκρίθηκαν για την εύρεση του πιο ακριβούς μοντέλου. Για τη βελτίωση της απόδοσης και της ερμηνευσιμότητας των μοντέλων, χρησιμοποιήθηκαν τεχνικές μείωσης διαστάσεων, όπως η Ανάλυση Κύριων Συνιστωσών (PCA) και η Γραμμική Διακριτή Ανάλυση (LDA).

Αφού βρέθηκε το βέλτιστο μοντέλο, ακολούθησε λεπτομερής ανάλυση για να εξηγηθεί η σημασία των χαρακτηριστικών που επηρεάζουν την πρόβλεψη του διαβήτη. Εντοπίστηκαν τα πιο σημαντικά χαρακτηριστικά, όπως τα επίπεδα γλυκόζης και η ηλικία, και αξιολογήθηκε η συμβολή τους στις προβλέψεις του μοντέλου. Η ερμηνευσιμότητα αυτή είναι απαραίτητη για τη διασφάλιση της ευθυγράμμισης των προβλέψεων με την κλινική γνώση και για την ενίσχυση της εμπιστοσύνης των χρηστών στη χρήση του μοντέλου.

Για την επίδειξη της πρακτικής εφαρμογής, αναπτύχθηκε μια διαδικτυακή διεπαφή χρησιμοποιώντας τη γλώσσα προγραμματισμού Julia και το πλαίσιο Dash.jl. Αυτή η διεπαφή επιτρέπει στους χρήστες να εισάγουν κλινικά δεδομένα και να λαμβάνουν προβλέψεις σε πραγματικό χρόνο για τον κίνδυνο εμφάνισης διαβήτη. Η εφαρμογή αξιοποιεί τις υψηλές επιδόσεις της Julia, διασφαλίζοντας αποδοτικότητα και κλιμάκωση, και είναι κατάλληλη για χρήση τόσο σε κλινικά όσο και σε ερευνητικά περιβάλλοντα.

Η μελέτη αυτή αναδεικνύει το δυναμικό της μηχανικής μάθησης στον τομέα της υγείας, ιδιαίτερα στην προγνωστική ανάλυση για την έγκαιρη διάγνωση και πρόληψη του διαβήτη. Παράλληλα, παρέχει μια βάση για μελλοντική έρευνα σε πιο σύνθετους ιατρικούς τομείς, όπου η μηχανική μάθηση μπορεί να οδηγήσει σε περαιτέρω καινοτομίες.

Λέξεις-Κλειδιά: Τεχνητή Νοημοσύνη, Μηχανική Μάθηση, Στατιστική, Διαβήτης,Python, Julia

ABSTRACT

This thesis explores the use of machine learning techniques for predicting the risk of diabetes using clinical and demographic data. The study utilizes both supervised and unsupervised learning methods to develop models capable of identifying high-risk individuals. Key supervised algorithms, including Logistic Regression, Random Forest, and Support Vector Machine (SVM), were implemented and compared to determine the most accurate model. Dimensionality reduction techniques, such as Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA), were used to improve model performance and interpretability.

After identifying the optimal model, a detailed analysis was conducted to explain the significance of various features in predicting diabetes risk. The most influential factors, such as glucose levels and age, were identified and their contribution to the model's predictions was assessed. This interpretability is crucial for ensuring that the predictions align with clinical understanding and for fostering trust in the model's use in medical decision-making.

To demonstrate practical applications, a web-based interface was developed using the Julia programming language and the Dash.jl framework. This interface allows users to input clinical data and receive real-time predictions about diabetes risk. The application leverages the high-performance capabilities of Julia, ensuring efficiency and scalability, and is suitable for deployment in both clinical and research environments.

This work highlights the potential of machine learning in healthcare, particularly for predictive analytics in the early diagnosis and prevention of diabetes. The study also provides a foundation for future research into more complex medical domains, where machine learning can drive further innovations.

Keywords: Artificial Intelligence (AI), Machine Learning, Statistics, Diabetes, Python, Julia

Ευχαριστίες

Ευχαριστώ θερμά τον επιβλέποντα καθηγητή μου, κ. Πέτρο Στεφανέα, για την πολύτιμη καθοδήγηση και στήριξή του κατά τη διάρκεια της διπλωματικής μου εργασίας. Η βοήθειά του ήταν καθοριστική για την ολοκλήρωση αυτής της προσπάθειας.

Ευχαριστώ όλους τους φίλους που έκανα κατά τη διάρκεια των φοιτητικών μου χρόνων με ιδιαίτερη σημασία στον Γεώργιο Λεβή για τη βοήθεια και τον συναγωνισμό που μου προσέφερε. Η στήριξή του και η ώθηση να γίνομαι καλύτερος έκαναν αυτή τη διαδρομή πιο δυνατή και ουσιαστική.

Τέλος, θα ήθελα να ευχαριστήσω την οικογένειά μου και την κοπέλα μου για την αμέριστη στήριξη και την αγάπη που μου πρόσφεραν καθ' όλη τη διάρκεια αυτής της διαδρομής.

Αντώνης Μήτσης,
Σεπτέμβριος 2024

Περιεχόμενα

ΛΙΣΤΑ ΕΙΚΟΝΩΝ

1. ΕΙΣΑΓΩΓΗ

1.1 Μηχανική μάθηση στην ιατρική

Η μηχανική μάθηση (ΜΜ) είναι μια παλιά έννοια που πρόσφατα απέκτησε μεγάλη προσοχή λόγω της έκρηξης των διαδικασιών παραγωγής δεδομένων στην υγειονομική περίθαλψη. Περίπου το 86% των οργανισμών υγειονομικής περίθαλψης χρησιμοποιούν κάποια μορφή λύσεων ΜΜ και πάνω από το 80% των ηγετών οργανισμών υγειονομικής περίθαλψης έχουν ένα σχέδιο τεχνητής νοημοσύνης (AI). Η ΜΜ είναι ένας σημαντικός κλάδος στον ευρύτερο τομέα της τεχνητής νοημοσύνης. Η ΜΜ ορίζεται ως «η ικανότητα μιας μηχανής να μιμείται την ευφυή ανθρώπινη συμπεριφορά». Ένας αλγόριθμος εκπαιδεύεται ώστε να μαθαίνει από δεδομένα και στη συνέχεια να λαμβάνει αποφάσεις με βάση παρόμοια χαρακτηριστικά ή μεταβλητές από νέα δεδομένα.

Η διασταύρωση διαφόρων επιστημονικών κλάδων, ιδίως των μαθηματικών, της στατιστικής και της επιστήμης των υπολογιστών, είναι μια κρίσιμη πτυχή της επιστήμης των δεδομένων που απαιτείται για την εφαρμογή διαφόρων μοντέλων ΜΜ. Παρά τη δύναμη της ανθρώπινης νοημοσύνης, οι άνθρωποι τείνουν να κάνουν λάθη λόγω της περιορισμένης βραχυπρόθεσμης μνήμης τους. Όταν η τεράστια αύξηση των δεδομένων συνδυάζεται με την αυξανόμενη ικανότητα των υπολογιστών να επεξεργάζονται και να χρησιμοποιούν τα δεδομένα για τη διαμόρφωση διαφόρων αλγορίθμων μηχανικής μάθησης, υπάρχει η ευκαιρία να χρησιμοποιηθεί η μηχανική μάθηση για να βοηθήσει τους ανθρώπους να λαμβάνουν αποφάσεις, λαμβάνοντας υπόψη ένα σημαντικό αριθμό πληροφοριών που σχετίζονται με το πλαίσιο.

Πρόσφατα, η ΜΜ έχει χρησιμοποιηθεί σε διάφορους ιατρικούς κλάδους, συμπεριλαμβανομένης της διαχείρισης της καρδιακής ανεπάρκειας, της υποστήριξης κλινικών αποφάσεων στην κλινική ιατρική, της ιατρικής απεικόνισης καθώς και στην πρόβλεψη του διαβήτη.

Σε σύγκριση με την παραδοσιακή στατιστική ανάλυση που βασίζεται σε υποθέσεις, η μηχανική μάθηση (ΜΜ) επικεντρώνεται στην ακρίβεια πρόβλεψης ενός μοντέλου. Οι επιδημιολογικές μέθοδοι καθοδηγούν παραδοσιακά τη διαδικασία παραγωγής δεδομένων στην υγειονομική περίθαλψη. Αν και οι μέθοδοι της ΜΜ επιτρέπουν νέους τρόπους για την απάντηση διαφόρων ερωτημάτων, η έλλειψη αποτελεσματικής ολοκλήρωσης μεταξύ των δύο κλάδων αποτελεί σημαντική πρόκληση, ιδίως όταν οι επιστήμονες δεδομένων και οι επιδημιολόγοι επικοινωνούν ως ομάδα.

Έχει επίσης υποστηριχθεί ότι παρόλο που χρησιμοποιούνται πολλές διαφορετικές ορολογίες σε αυτούς τους κλάδους, συχνά χρησιμοποιούνται διαφορετικές λέξεις για επικαλυπτόμενες ή σχεδόν ίδιες έννοιες. Επειδή οι επιδημιολόγοι και οι στατιστικολόγοι χρησιμοποιούν εδώ και καιρό τέτοιες έννοιες στην έρευνα που βασίζεται σε υποθέσεις, είναι αρκετά εύκολο γι' αυτούς να κατανοήσουν αυτές τις έννοιες, αν ευαισθητοποιηθούν στο γεγονός ότι η ΜΜ εφαρμόζει αυτές τις έννοιες μέσα σε ένα νέο πλαίσιο.

Επιπλέον, άλλοι επαγγελματίες της υγειονομικής περίθαλψης, όπως κλινικοί γιατροί, γιατροί δημόσιας υγείας, παθολόγοι και ακτινολόγοι, είναι περισσότερο εξοικειωμένοι με την έρευνα βάσει υποθέσεων παρά με τους αλγορίθμους ΜΜ. Ως εκ τούτου, είναι απαραίτητο να μεταφερθεί ο εννοιολογικός παραλληλισμός μεταξύ των δύο διαφορετικών αλλά στενά συνδεδεμένων κλάδων.

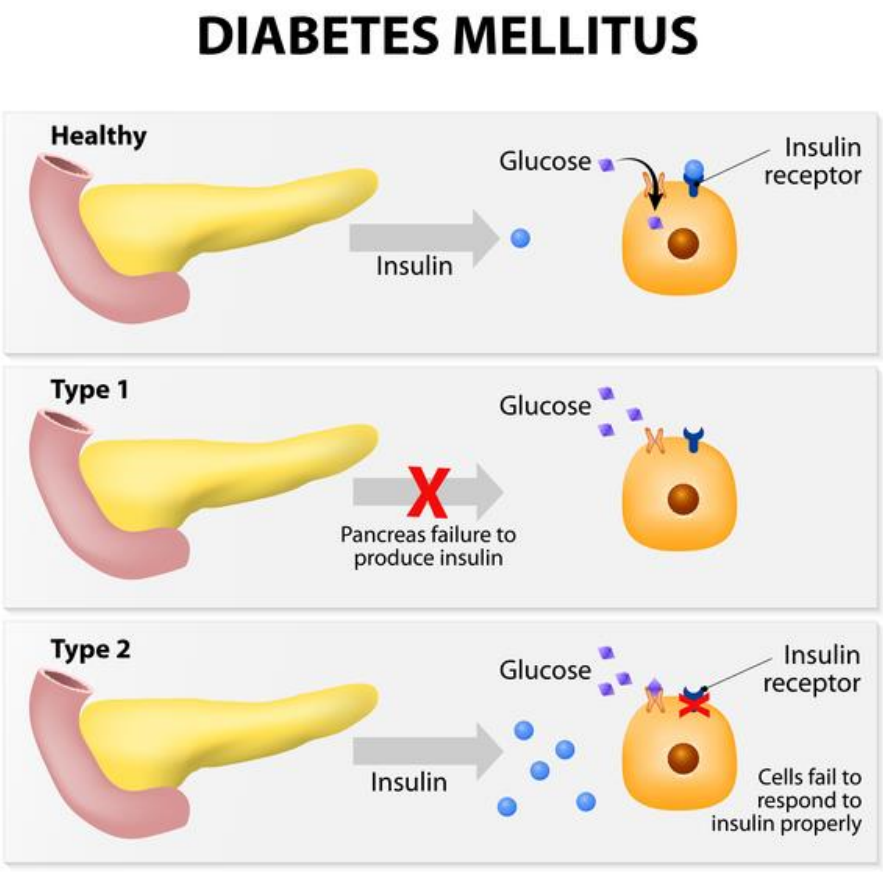
1.2 Διαβήτης

Ο σακχαρώδης διαβήτης (ΣΔ) ορίζεται ως μια ομάδα μεταβολικών διαταραχών που προκαλούνται κυρίως από μη φυσιολογική έκκριση ή/και δράση ινσουλίνης. Η ανεπάρκεια ινσουλίνης έχει ως αποτέλεσμα αυξημένα επίπεδα γλυκόζης στο αίμα (υπεργλυκαιμία) και διαταραχή του μεταβολισμού των υδατανθράκων, του λίπους και των πρωτεϊνών. Ο ΣΔ είναι μία από τις πιο συχνές ενδοκρινικές διαταραχές, επηρεάζοντας περισσότερα από 200 εκατομμύρια άτομα παγκοσμίως. Η εμφάνιση του διαβήτη εκτιμάται ότι θα αυξηθεί δραματικά τα επόμενα χρόνια.

Ο ΣΔ μπορεί να χωριστεί σε διάφορους διακριτούς τύπους. Ωστόσο, υπάρχουν δύο κύριοι κλινικοί τύποι, ο διαβήτης τύπου 1 (Τ1Δ) και ο διαβήτης τύπου 2 (Τ2Δ), ανάλογα με την αιτιοπαθολογία της διαταραχής. Ο Τ2Δ φαίνεται να είναι η πιο συχνή μορφή διαβήτη (90% όλων των διαβητικών ασθενών), που χαρακτηρίζεται κυρίως από αντίσταση στην ινσουλίνη. Τα κύρια αίτια του Τ2Δ περιλαμβάνουν τον τρόπο ζωής, τη σωματική δραστηριότητα, τις διατροφικές συνήθειες και την κληρονομικότητα, ενώ ο Τ1Δ θεωρείται ότι οφείλεται σε αυτοανοσολογική καταστροφή των νησιδίων Langerhans που φιλοξενούν τα παγκρεατικά-β κύτταρα. Ο Τ1Δ προσβάλλει σχεδόν το 10% όλων των διαβητικών ασθενών παγκοσμίως, ενώ το 10% αυτών αναπτύσσει τελικά ιδιοπαθή διαβήτη.

Άλλες μορφές ΣΔ, οι οποίες ταξινομούνται με βάση το προφίλ έκκρισης ινσουλίνης ή/και την έναρξη, περιλαμβάνουν τον διαβήτη κύησης, τις ενδοκρινοπάθειες, τον MODY

(Maturity Onset Diabetes of the Young), τον νεογνικό και τον μιτοχονδριακό. Τα συμπτώματα του ΣΔ περιλαμβάνουν μεταξύ άλλων πολυουρία, πολυδιψία και σημαντική απώλεια βάρους.



Εικόνα 1: Τύποι Διαβήτη

1.3 Σκοπός

Στην παρούσα διατριβή, εκπαιδεύουμε διάφορους αλγορίθμους Μηχανικής Μάθησης με στόχο να προβλέψουμε ποιοι ασθενείς διατρέχουν κίνδυνο να αναπτύξουν διαβήτη. Εξετάζουμε διάφορες μεθόδους προεπεξεργασίας δεδομένων, συμπεριλαμβανομένης της κανονικοποίησης δεδομένων και της επιλογής χαρακτηριστικών, για να βελτιώσουμε την ποιότητα και τη συνάφεια των δεδομένων εισόδου. Στη συνέχεια πραγματοποιούμε μια ολοκληρωμένη σύγκριση αυτών των αλγορίθμων, αξιολογώντας την απόδοσή τους με βάση βασικές μετρήσεις όπως η ακρίβεια. Στόχος μας είναι να εντοπίσουμε τον αλγόριθμο που όχι

μόνο προβλέπει τον διαβήτη με την υψηλότερη ακρίβεια αλλά και προσφέρει αξιόπιστα αποτελέσματα, λαμβάνοντας υπόψη την ευαισθησία και την ειδικότητα των προβλέψεων.

Επιπλέον, η παρούσα μελέτη εμβαθύνει στην ερμηνευσιμότητα των επιλεγμένων μοντέλων, διασφαλίζοντας ότι τα αποτελέσματα είναι σαφή και εφαρμόσιμα για τους επαγγελματίες του τομέα της υγείας. Αξιολογούμε επίσης τη δυνητική επεκτασιμότητα αυτών των μοντέλων, εξετάζοντας τη δυνατότητα εφαρμογής τους σε διάφορα περιβάλλοντα υγειονομικής περίθαλψης, ώστε να διασφαλίσουμε ότι μπορούν να εφαρμοστούν αποτελεσματικά σε διαφορετικά κλινικά περιβάλλοντα. Τέλος, κατασκευάζουμε ένα μοντέλο πρόβλεψης στην Julia, αξιοποιώντας τις δυνατότητες υψηλής απόδοσης που διαθέτει για να παρέχουμε μια αποτελεσματική και κλιμακούμενη λύση για την πρόβλεψη του διαβήτη. Μέσω αυτής της έρευνας, στοχεύουμε να προσφέρουμε μια ολοκληρωμένη προσέγγιση για την πρόβλεψη του διαβήτη που είναι τόσο ακριβής όσο και πρακτική για ευρεία κλινική χρήση.

2. ΜΗΧΑΝΙΚΗ ΜΑΘΗΣΗ

2.1 Γενικές Πληροφορίες

Η μηχανική μάθηση επικεντρώνεται στο πώς μπορούμε να σχεδιάσουμε υπολογιστικά συστήματα που βελτιώνονται αυτόματα μέσα από την εμπειρία τους. Αποτελεί έναν από τους ταχύτερα αναπτυσσόμενους κλάδους της τεχνολογίας, καθώς συνδυάζει την επιστήμη των υπολογιστών και τη στατιστική, αποτελώντας θεμέλιο για την τεχνητή νοημοσύνη και την επιστήμη δεδομένων. Η πρόσφατη πρόοδος στον τομέα αυτό προέρχεται από την ανάπτυξη νέων αλγορίθμων και θεωριών, καθώς και από τη διαρκώς αυξανόμενη διαθεσιμότητα μεγάλων όγκων δεδομένων και των χαμηλού κόστους υπολογιστικών υποδομών. Οι μέθοδοι της μηχανικής μάθησης, οι οποίες βασίζονται σε δεδομένα, έχουν βρει εφαρμογή σε πλήθος τομέων, όπως η υγεία, η τεχνολογία, η εκπαίδευση, τα οικονομικά, η δημόσια ασφάλεια και το μάρκετινγκ, οδηγώντας σε πιο τεκμηριωμένες και ακριβείς αποφάσεις. Οι κύριες κατηγορίες αλγορίθμων ΜΜ είναι οι επιβλεπόμενη μάθηση, μη-επιβλεπόμενη μάθηση, ημι-επιβλεπόμενη μάθηση και ενισχυτική μάθηση.

1. **Επιβλεπόμενη Μάθηση:** Η μηχανική μάθηση με επίβλεψη είναι η διαδικασία εκμάθησης μιας συνάρτησης που συσχετίζει μια είσοδο με μια έξοδο, χρησιμοποιώντας παραδείγματα όπου τόσο η είσοδος όσο και η έξοδος είναι

γνωστές. Οι αλγόριθμοι αυτοί εκπαιδεύονται με επισημασμένα δεδομένα, που περιλαμβάνουν ένα σύνολο ζευγών εισόδου-εξόδου. Στην ουσία, οι αλγόριθμοι εποπτευόμενης μάθησης απαιτούν εξωτερική καθοδήγηση, καθώς χρησιμοποιούν τα επισημασμένα δεδομένα για να μάθουν τον τρόπο αντιστοίχισης των εισόδων στις αντίστοιχες εξόδους.

2. **Μη Επιβλεπόμενη Μάθηση:** Σε αντίθεση με την επιβλεπόμενη μάθηση, στην μη επιβλεπόμενη μάθηση δεν υπάρχουν σωστές απαντήσεις και δεν υπάρχει δάσκαλος. Οι αλγόριθμοι αφήνονται στην τύχη τους να ανακαλύψουν και να παρουσιάσουν την ενδιαφέρουσα δομή στα δεδομένα. Οι αλγόριθμοι μάθησης χωρίς επίβλεψη μαθαίνουν λίγα χαρακτηριστικά από τα δεδομένα. Όταν εισάγονται νέα δεδομένα, χρησιμοποιούν τα χαρακτηριστικά που έχουν μάθει προηγουμένως για να αναγνωρίσουν την τάξη των δεδομένων. χρησιμοποιείται κυρίως για την ομαδοποίηση και τη μείωση των χαρακτηριστικών.

3. **Ημι-Επιβλεπόμενη Μάθηση:** Η μηχανική μάθηση με ημιεπίβλεψη είναι ένας συνδυασμός εποπτευόμενων και μη εποπτευόμενων μεθόδων μηχανικής μάθησης. Μπορεί να αποδώσει καρπούς σε εκείνους τους τομείς της μηχανικής μάθησης και της εξόρυξης δεδομένων όπου τα μη επισημασμένα δεδομένα είναι ήδη παρόντα και η απόκτηση των επισημασμένων δεδομένων είναι μια κουραστική διαδικασία.

4. **Ενισχυτική Μάθηση:** Η ενισχυτική μάθηση είναι ένας κλάδος της μηχανικής μάθησης που εστιάζει στο πώς οι πράκτορες λογισμικού μπορούν να λαμβάνουν αποφάσεις σε ένα περιβάλλον για να μεγιστοποιήσουν μια συνολική ανταμοιβή. Ουσιαστικά, οι πράκτορες μαθαίνουν να επιλέγουν ενέργειες που θα τους προσφέρουν τη μεγαλύτερη δυνατή ανταμοιβή με την πάροδο του χρόνου, βάσει της εμπειρίας που αποκτούν από την αλληλεπίδρασή τους με το περιβάλλον.

2.2 Επιβλεπόμενη Μάθηση

Η μηχανική μάθηση με επίβλεψη είναι η διαδικασία εκμάθησης μιας συνάρτησης που συσχετίζει μια είσοδο με μια έξοδο, χρησιμοποιώντας παραδείγματα όπου τόσο η είσοδος όσο και η έξοδος είναι γνωστές. Οι αλγόριθμοι αυτοί εκπαιδεύονται με επισημασμένα δεδομένα, που περιλαμβάνουν ένα σύνολο ζευγών εισόδου-εξόδου. Στην ουσία, οι αλγόριθμοι επιβλεπόμενης μάθησης απαιτούν εξωτερική καθοδήγηση, καθώς χρησιμοποιούν τα επισημασμένα δεδομένα για να μάθουν τον τρόπο αντιστοίχισης των εισόδων στις αντίστοιχες εξόδους.

2.2.1 Πως Λειτουργεί

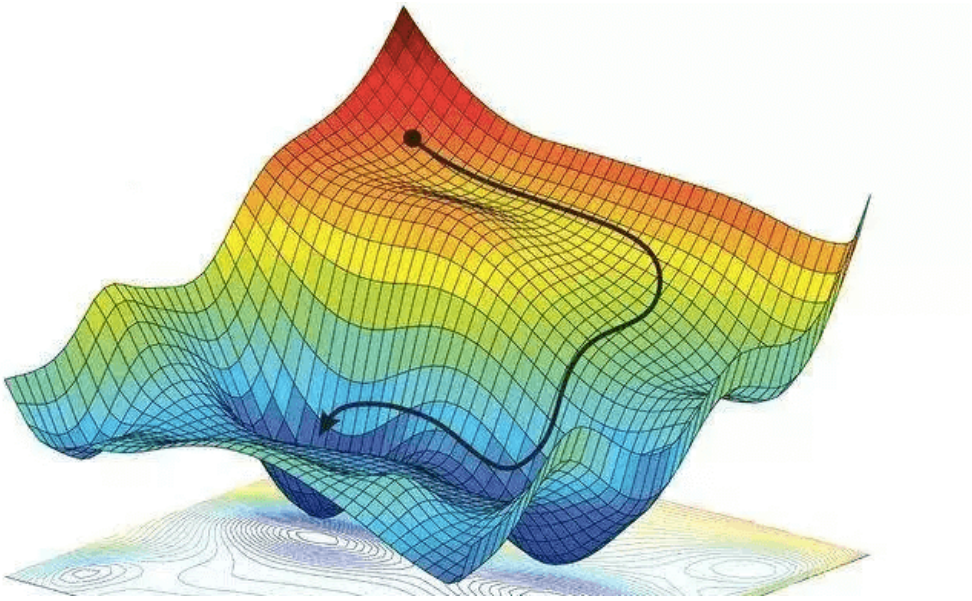
Ας θεωρήσουμε ένα σύνολο N παραδειγμάτων εκπαίδευσης, το οποίο συμβολίζεται ως $\{(x_1, y_1), \ldots, (x_N, y_N)\}$ όπου x_i είναι το διάνυσμα χαρακτηριστικών για το i-οστό παράδειγμα, και y_i είναι η αντίστοιχη ετικέτα. Ο στόχος του αλγορίθμου μάθησης είναι να ανακαλύψει μια συνάρτηση $g: X \rightarrow Y$ όπου το X είναι το σύνολο των εισόδων και το Y είναι το σύνολο των πιθανών εξόδων. Η συνάρτηση g μπορεί να αναπαρασταθεί από μια άλλη συνάρτηση $f: X \times Y \rightarrow R$, έτσι ώστε το g να προσδιορίζει την έξοδο y που μεγιστοποιεί τη βαθμολογία για μια δεδομένη είσοδο x. Αυτό μπορεί να εκφραστεί ως:

$$g(x) = arg_y max f(x, y)$$

Για να αξιολογήσουμε πόσο καλά η συνάρτηση g αντιστοιχεί τις εξόδους στις εισόδους, ορίζουμε μια συνάρτηση απώλειας $L: X \times Y \rightarrow R^{\geq 0}$ Για ένα δεδομένο ζεύγος (x_i, y_i) η απώλεια όταν προβλέπεται το y′ δίνεται από τη συνάρτηση $L(y_i, y')$. Επιπλέον, ορίζουμε μια συνάρτηση κινδύνου, R(g), η οποία υπολογίζει την αναμενόμενη απώλεια κατά τη χρήση της συνάρτησης g. Ο εμπειρικός κίνδυνος, εκφράζεται ως εξής:

$$R_{emp}(g) = \frac{1}{N} \sum_i L(y_i, g(x_i))$$

Ο στόχος μας είναι να βρούμε τη συνάρτηση g που ελαχιστοποιεί αυτόν τον εμπειρικό κίνδυνο, Remp(g).



Εικόνα 2: Gradient Descent

Κατά το σχεδιασμό ενός αλγορίθμου επιβλεπόμενης μάθησης υπάρχουν τέσσερα κύρια ζητήματα που πρέπει να ληφθούν υπόψη:

1. **Το πρόβλημα αντιστάθμισης μεταξύ μεροληψίας και διακύμανσης.** Η μεροληψία αφορά την ακρίβεια των προβλέψεων. Υψηλή μεροληψία σημαίνει ότι ο αλγόριθμος είναι ανακριβής, κάτι που συνήθως είναι σημάδι υποεκπαίδευσης (ο αλγόριθμος κάνει υποθέσεις χωρίς να λαμβάνει υπόψη όλα τα δεδομένα ή τα δεδομένα είναι πολύ λίγα). Η διακύμανση είναι η ευαισθησία σε μικρές αλλαγές στις εισόδους. Προκαλεί θόρυβο και είναι σημάδι υπερεκπαίδευσης (ο αλγόριθμος είναι υπερβολικά περίπλοκος, με αποτέλεσμα να λαμβάνει υπόψη πολλά χαρακτηριστικά των δεδομένων που δεν συσχετίζονται με το αποτέλεσμα και αποτυγχάνει να γενικεύσει σε νέα δεδομένα). Πρέπει να βρούμε μια ισορροπία μεταξύ των δύο, καθώς συνήθως η μείωση του ενός αυξάνει το άλλο.
2. **Σύνθετοτητα συνάρτησης και ποσότητα δεδομένων εκπαίδευσης.** Αν μια συνάρτηση είναι απλή (η συνάρτηση χρειάζεται μόνο λίγα χαρακτηριστικά για να δώσει σωστό αποτέλεσμα), τότε ένα πρόγραμμα με υψηλή μεροληψία και χαμηλή διακύμανση θα είναι αρκετό. Ωστόσο, αν τα σημαντικά χαρακτηριστικά είναι πάρα πολλά και η συνάρτηση είναι σύνθετη, χρειαζόμαστε πολλά δεδομένα για τον αλγόριθμο εκπαίδευσης, που θα έχει χαμηλή μεροληψία και υψηλή διακύμανση.
3. **Διαστατικότητα εισόδου.** Αν τα διανύσματα εισόδου έχουν πάρα πολλά χαρακτηριστικά, το πρόγραμμα δεν θα βρει εύκολα τη συνάρτηση. Σε αυτήν την

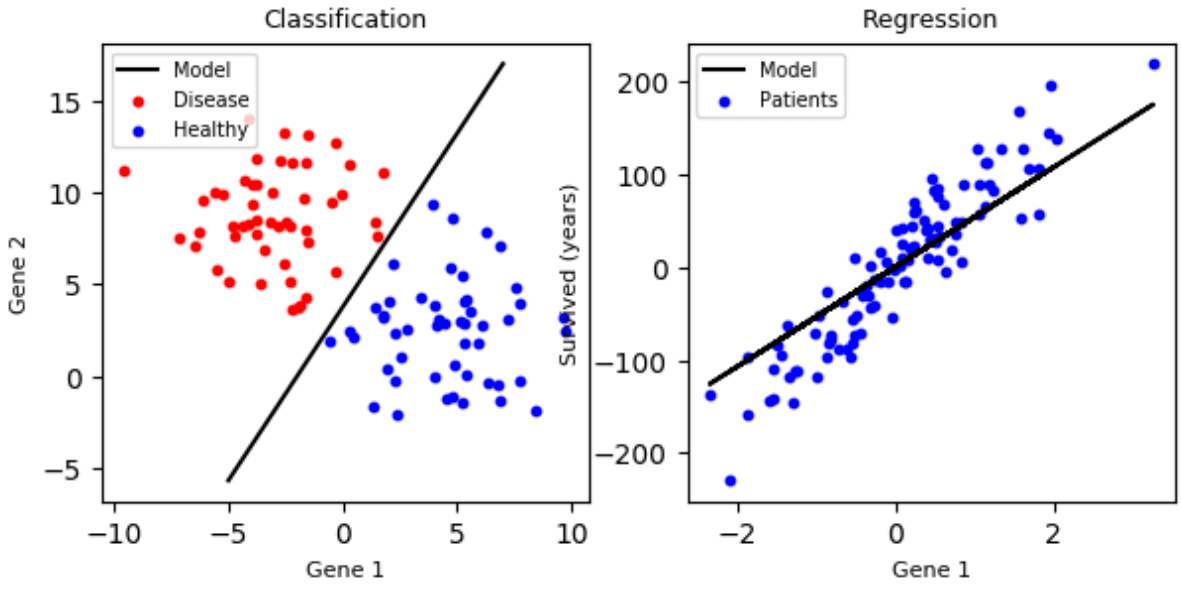
περίπτωση, συνήθως αφαιρούμε τα μη σχετιζόμενα χαρακτηριστικά, χρησιμοποιώντας στρατηγικές μείωσης διαστατικότητας.

4. **Θόρυβος στην έξοδο.** Αν πολλά δεδομένα έχουν λανθασμένο αποτέλεσμα, τότε ο αλγόριθμος δεν πρέπει να προσπαθήσει να είναι πολύ ακριβής στις εικασίες του, καθώς αυτό μπορεί να οδηγήσει σε υπερεκπαίδευση. Η πρόωρη διακοπή και η ανίχνευση είναι δύο από τις πιο κοινές προσεγγίσεις για την αντιμετώπιση αυτού του ζητήματος.

2.2.2 Είδη Προβλημάτων Επιβλεπόμενης Μάθησης

Τα προβλήματα μάθησης με επίβλεψη μπορούν να κατηγοριοποιηθούν σε δύο τύπους: παλινδρόμηση και ταξινόμηση.

- **Παλινδρόμηση:** Η παλινδρόμηση χρησιμοποιείται συνήθως για την πρόβλεψη συνεχών τιμών. Αυτό σημαίνει ότι η εξαρτημένη μεταβλητή που προσπαθούμε να προβλέψουμε ή να εξηγήσουμε μπορεί να πάρει οποιαδήποτε αριθμητική τιμή εντός ενός συγκεκριμένου εύρους. Παραδείγματα περιλαμβάνουν την πρόβλεψη της θερμοκρασίας, του εισοδήματος, ή της διάρκειας ζωής μιας μπαταρίας.
- **Ταξινόμηση:** Από την άλλη πλευρά, η ταξινόμηση (classification) χρησιμοποιείται όταν η εξαρτημένη μεταβλητή είναι διακριτή, δηλαδή παίρνει μια από περιορισμένο αριθμό κατηγοριών ή τιμών. Παραδείγματα περιλαμβάνουν την πρόβλεψη του τύπου ασθένειας ενός ασθενούς, την αναγνώριση της κατηγορίας ενός αντικειμένου σε μια εικόνα, ή την κατάταξη ενός email ως spam ή όχι.



Εικόνα 3: Ταξινόμηση και Παλινδρόμηση

2.3 Αλγόριθμοι Επιβλεπόμενης Μάθησης

Σε αυτήν την ενότητα θα αναλύσουμε τους αλγόριθμους επιβλεπόμενης μάθησης που χρησιμοποιήθηκαν στην παρούσα διατριβή.

2.3.1 Λογιστική Παλινδρόμηση

Η Λογιστική Παλινδρόμηση είναι μια στατιστική μέθοδος που χρησιμοποιείται για προβλήματα δυαδικής ταξινόμησης, όπου το αποτέλεσμα είναι 0 ή 1. Χρησιμοποιεί τη λογιστική (σιγμοειδή) συνάρτηση για να μοντελοποιήσει την πιθανότητα ένα δεδομένο εισόδου να ανήκει σε μια συγκεκριμένη κλάση. Η σιγμοειδής συνάρτηση εξάγει μια τιμή μεταξύ 0 και 1, η οποία μπορεί να ερμηνευτεί ως η πιθανότητα το αποτέλεσμα να ανήκει σε μια συγκεκριμένη κλάση. Η συνάρτηση ορίζεται ως εξής:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

όπου z είναι ο γραμμικός συνδυασμός των χαρακτηριστικών εισόδου σταθμισμένων με τους συντελεστές, συν έναν όρο μεροληψίας. Συγκεκριμένα, μπορούμε να γράψουμε z ως:

$$z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n$$

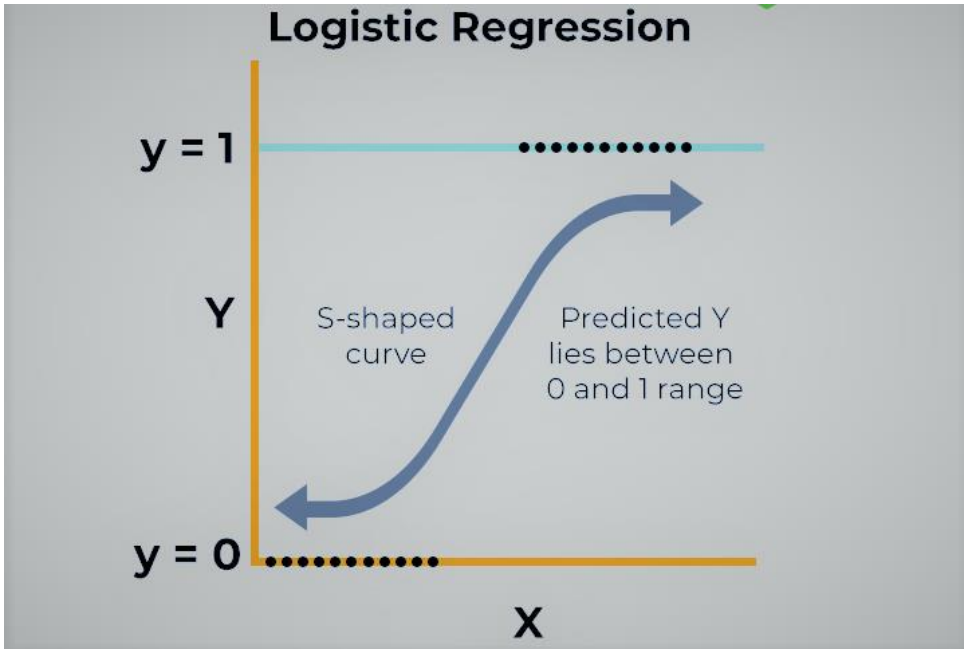
Όπου:

- β_0 είναι ο όρος μεροληψίας,
- β_i είναι οι συντελεστές του μοντέλου,
- x_i είναι τα χαρακτηριστικά εισόδου

Η συνάρτηση logit (λογαριθμικές πιθανότητες) σχετίζεται άμεσα με τη σιγμοειδή συνάρτηση και εκφράζει τη φυσική λογαριθμική τιμή της πιθανότητας επιτυχίας προς την πιθανότητα αποτυχίας. Αυτή η συνάρτηση δίνεται από τον τύπο:

$$logit(p) = \ln\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n$$

Το μοντέλο λογιστικής παλινδρόμησης προσδιορίζει τις παραμέτρους βi έτσι ώστε να μεγιστοποιηθεί η πιθανότητα των δεδομένων που παρατηρήθηκαν.



Εικόνα 4: Λογιστική Παλινδρόμηση

2.3.2 SVM

Ο Support Vector Machine (SVM) είναι ένας αλγόριθμος επιβλεπόμενης μηχανικής μάθησης που χρησιμοποιείται τόσο για ταξινόμηση όσο και για παλινδρόμηση, αν και είναι περισσότερο γνωστός για τα προβλήματα ταξινόμησης. Ο κύριος στόχος του SVM είναι να βρει το καλύτερο δυνατό υπερεπίπεδο σε έναν χώρο πολλαπλών διαστάσεων, το οποίο μπορεί να διαχωρίσει τα δεδομένα σε διαφορετικές κλάσεις με το μέγιστο δυνατό περιθώριο. Το υπερεπίπεδο αυτό επιτυγχάνει το διαχωρισμό, μεγιστοποιώντας την απόσταση μεταξύ των πλησιέστερων σημείων από κάθε κλάση, που ονομάζονται υποστηρικτικά διανύσματα.

Η διάσταση του υπερεπιπέδου εξαρτάται από τον αριθμό των χαρακτηριστικών του συνόλου δεδομένων. Για παράδειγμα, αν έχουμε δύο χαρακτηριστικά εισόδου, το υπερεπίπεδο είναι μια γραμμή, ενώ με τρία χαρακτηριστικά το υπερεπίπεδο είναι ένα επίπεδο. Σε δεδομένα με περισσότερες από τρεις διαστάσεις, το υπερεπίπεδο είναι πιο δύσκολο να οπτικοποιηθεί, αλλά η βασική ιδέα παραμένει η ίδια.

Ας εξετάσουμε ένα πρόβλημα δυαδικής ταξινόμησης, όπου οι δύο κλάσεις είναι +1 και -1. Το σύνολο δεδομένων εκπαίδευσης αποτελείται από διανύσματα χαρακτηριστικών (X) και τις αντίστοιχες ετικέτες κλάσεων (Y). Ο SVM προσπαθεί να βρει το υπερεπίπεδο που διαχωρίζει τα δεδομένα των δύο κλάσεων με τον καλύτερο δυνατό τρόπο, δηλαδή με το μεγαλύτερο περιθώριο μεταξύ των σημείων που βρίσκονται κοντά στα όρια των κλάσεων.
Η εξίσωση για το γραμμικό υπερεπίπεδο μπορεί να γραφεί ως εξής:

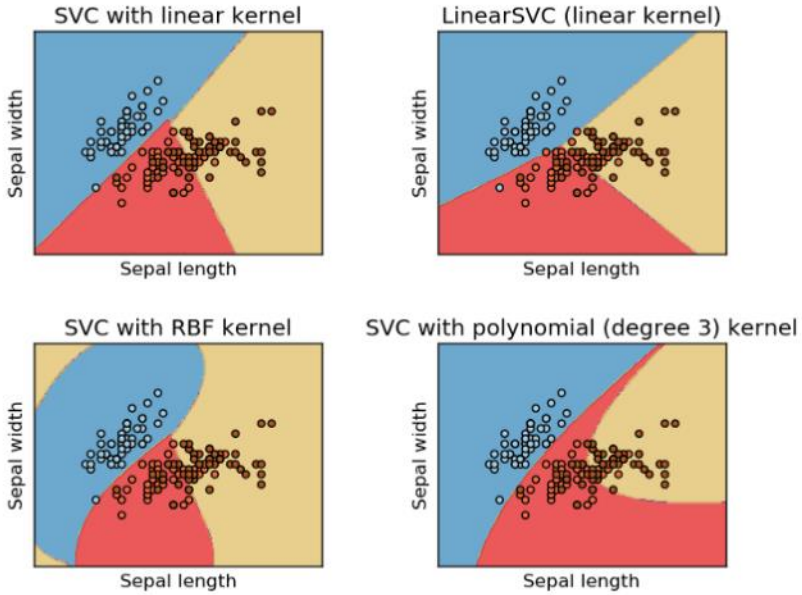
$$w^T x + b = 0$$

Το διάνυσμα w αντιπροσωπεύει το κανονικό διάνυσμα στο υπερεπίπεδο, δηλαδή τη διεύθυνση που είναι κάθετη στο υπερεπίπεδο. Η παράμετρος b στην εξίσωση αντιπροσωπεύει τη μετατόπιση ή την απόσταση του υπερεπιπέδου από την αρχή κατά μήκος του κανονικού διανύσματος w.

Για γραμμικό ταξινομητή SVM:

$$\hat{y} = \begin{cases} 1: \ w^T x + b \geq 0 \\ 0: \ w^T x + b < 0 \end{cases}$$

Στην περίπτωση που τα δεδομένα μας δεν μπορούν να διαχωριστούν από ένα γραμμικό όριο χρησιμοποιούμε μη-γραμμικό SVM. Σε αυτές τις περιπτώσεις, οι SVM χρησιμοποιούν το τέχνασμα του πυρήνα K(xi, xj) για να μετατρέψουν τα δεδομένα σε έναν χώρο υψηλότερων διαστάσεων όπου μπορεί να βρεθεί ένας γραμμικός διαχωρισμός. Αυτός ο μετασχηματισμός γίνεται με τη χρήση μιας συνάρτησης πυρήνα (kernel) , η οποία υπολογίζει το εσωτερικό γινόμενο στο μετασχηματισμένο χώρο χαρακτηριστικών. Οι συνήθεις συναρτήσεις πυρήνα περιλαμβάνουν τον πολυωνυμικό πυρήνα, τον Radial Basis Function (RBF) και τον σιγμοειδή πυρήνα.



Εικόνα 5: Είδη SVC

2.3.3 Naïve Bayes

Ο αλγόριθμος Naïve Bayes είναι μια απλή αλλά ισχυρή μέθοδος στατιστικής ταξινόμησης που βασίζεται στο Θεώρημα του Bayes, με την υπόθεση ότι τα χαρακτηριστικά είναι ανεξάρτητα μεταξύ τους. Αυτή η υπόθεση, αν και συχνά δεν είναι ακριβής στην πράξη (εξ ου και το " Naïve " στο όνομα), επιτρέπει την απλοποίηση του υπολογισμού και κάνει τον αλγόριθμο πολύ αποτελεσματικό.

Το Θεώρημα του Bayes εκφράζει την πιθανότητα ενός γεγονότος βάσει προηγούμενης γνώσης για συνθήκες που σχετίζονται με το γεγονός αυτό. Συγκεκριμένα, για δύο γεγονότα A και B, το Θεώρημα του Bayes δίνεται από τον τύπο:

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

όπου:

- P(A|B) είναι η πιθανότητα του γεγονότος A δεδομένου ότι το γεγονός B συνέβη (ονομάζεται και a posteriori πιθανότητα),
- P(B|A) είναι η πιθανότητα του γεγονότος B δεδομένου ότι το γεγονός A συνέβη,
- P(A) είναι η a priori πιθανότητα του γεγονότος A (η πιθανότητα του A ανεξαρτήτως άλλων παραμέτρων),
- P(B) είναι η a priori πιθανότητα του γεγονότος B.

Στο πλαίσιο της ταξινόμησης, θεωρούμε ότι έχουμε ένα σύνολο χαρακτηριστικών $x_1, x_{2,\dots,} x_n$ θέλουμε να προβλέψουμε την κλάση C. Η a posteriori πιθανότητα μιας κλάσης Ck δεδομένων των χαρακτηριστικών $x = (x_1, x_{2,\dots,} x_n)$ δίνεται από:

$$P(C_k|x) = \frac{P(x|C_k) \cdot P(C_k)}{P(x)}$$

Επειδή το P(x) είναι σταθερό για όλες τις κλάσεις, μπορούμε να αγνοήσουμε αυτόν τον όρο στην πράξη και να συγκρίνουμε τις τιμές των $P(x|C_k) \cdot P(C_k)$ για τις διάφορες κλάσεις C_k για να προσδιορίσουμε την πιο πιθανή κλάση.

Η υπόθεση της ανεξαρτησίας των χαρακτηριστικών απλοποιεί περαιτέρω τον υπολογισμό του $P(x|C_k)$ ως:

$$P(x|C_k) = P(x_1|C_k) \cdot P(x_2|C_k) \cdot \dots \cdot P(x_n|C_k)$$

Έτσι, η τελική πρόβλεψη γίνεται επιλέγοντας την κλάση με τη μέγιστη a posteriori πιθανότητα:

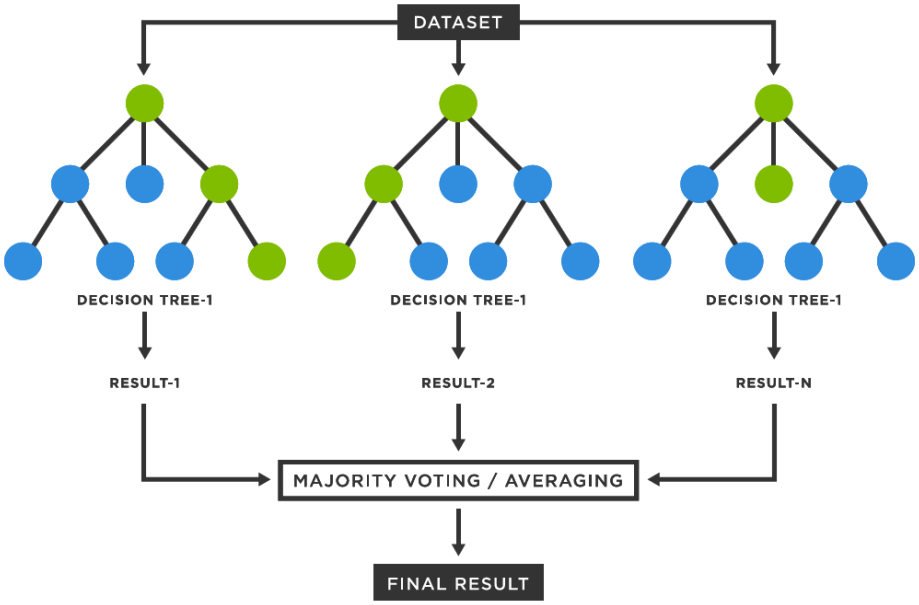
$$\hat{C} = \arg\max_{C_k} P(C_k) \prod_{i=1}^{n} P(x_i|C_k)$$

2.3.4 Random Forest

Ο αλγόριθμος Random Forest αποτελείται από ένα μεγάλο σύνολο μεμονωμένων δέντρων απόφασης. Κάθε δέντρο απόφασης παρέχει μια πρόβλεψη για μια δεδομένη είσοδο, και η πρόβλεψη του τελικού μοντέλου καθορίζεται από την ψήφο πλειοψηφίας μεταξύ όλων των δέντρων του δάσους. Για την αποτελεσματική απόδοση των τυχαίων δασών, είναι ζωτικής σημασίας τα μεμονωμένα δέντρα να μην έχουν υψηλή συσχέτιση μεταξύ τους. Αυτό επιτυγχάνεται μέσω μιας τεχνικής γνωστής ως bagging.

Το bagging περιλαμβάνει την επιλογή τυχαίων υποσυνόλων χαρακτηριστικών για τη διάσπαση κάθε δέντρου, διασφαλίζοντας ότι κάθε δέντρο απόφασης λειτουργεί με διαφορετικά τμήματα δεδομένων από τα υπόλοιπα. Σε αντίθεση με τα τυπικά δέντρα απόφασης, τα οποία συνήθως επιλέγουν διασπάσεις που ελαχιστοποιούν μια συνάρτηση κόστους, τα δέντρα σε ένα τυχαίο δάσος επιλέγουν διασπάσεις από ένα τυχαίο σύνολο χαρακτηριστικών χωρίς απαραίτητα να αναζητούν εκείνη που ελαχιστοποιεί το κόστος. Αυτή η προσέγγιση έχει σχεδιαστεί για να αυξήσει την ποικιλομορφία μεταξύ των δέντρων του δάσους.

Σε προβλήματα ταξινόμησης, όπου υπάρχουν p χαρακτηριστικά, είναι κοινή πρακτική για κάθε δέντρο να χρησιμοποιεί περίπου $\sqrt{p}$ χαρακτηριστικά σε κάθε διάσπαση. Ενώ τα τυχαία δάση αποδίδουν γενικά καλύτερη προβλεπτική απόδοση και μετριάζουν την υπερπροσαρμογή, αυτό γίνεται με κόστος την ερμηνευσιμότητα. Σε αντίθεση με τα δέντρα αποφάσεων, τα οποία είναι από τα πιο ερμηνεύσιμα μοντέλα μηχανικής μάθησης με επίβλεψη, ο αλγόριθμος Random Forest δημιουργεί ένα πιο πολύπλοκο σύνολο που είναι πιο δύσκολο να ερμηνευτεί.



Εικόνα 6: Ο αλγόριθμος του τυχαίου δάσους

2.3.5 XGBoost

Ο XGBoost, ή eXtreme Gradient Boosting, είναι ένας ισχυρός αλγόριθμος μηχανικής μάθησης που βασίζεται στο πλαίσιο Gradient Boosting. Έχει σχεδιαστεί για να ενισχύει τόσο την ταχύτητα όσο και τις επιδόσεις, καθιστώντας τον δημοφιλή επιλογή για ένα ευρύ φάσμα εργασιών προγνωστικής μοντελοποίησης.

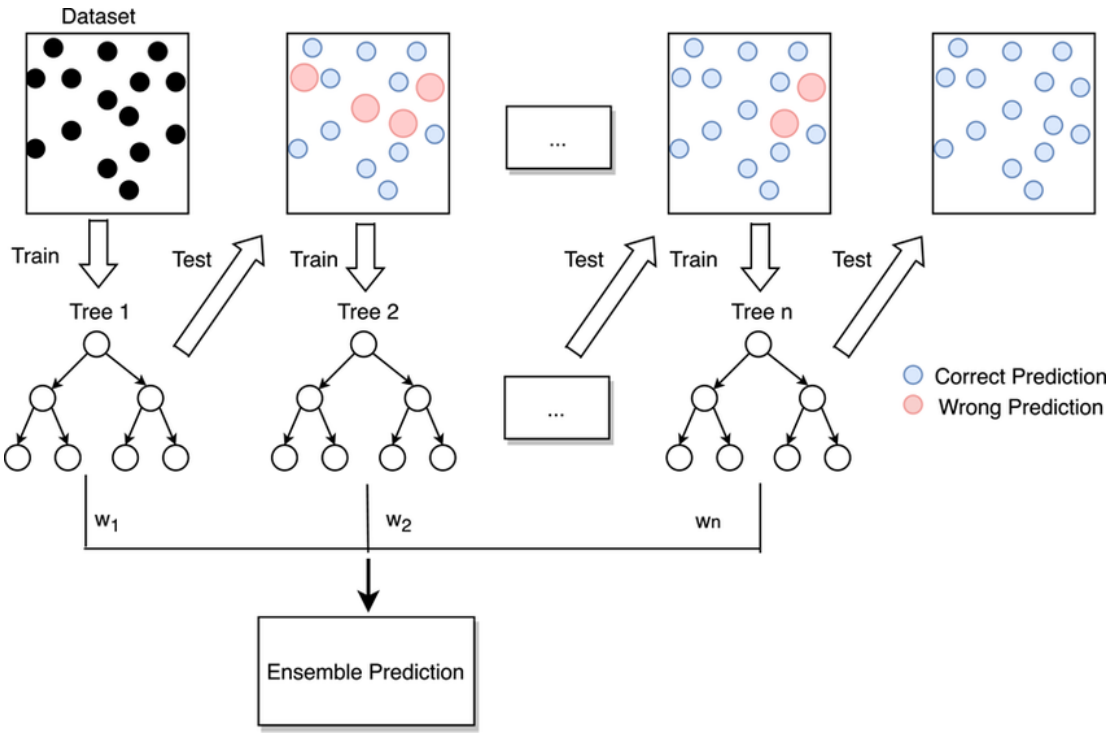
Ο πυρήνας του XGBoost έγκειται στην επαναληπτική του προσέγγιση για την κατασκευή δέντρων απόφασης. Κάθε δέντρο κατασκευάζεται διαδοχικά, με κάθε νέο δέντρο να επικεντρώνεται στη διόρθωση των σφαλμάτων που έγιναν από τα προηγούμενα δέντρα. Η διαδικασία αυτή αποσκοπεί στην ελαχιστοποίηση μιας καθορισμένης συνάρτησης απωλειών, βελτιώνοντας τη συνολική ακρίβεια του μοντέλου.

Ένα ιδιαίτερο χαρακτηριστικό του XGBoost είναι η ενσωμάτωση της κανονικοποίησης στην αντικειμενική συνάρτηση. Αυτή η προσθήκη βοηθά στη διαχείριση της πολυπλοκότητας του μοντέλου και αποτρέπει την υπερβολική προσαρμογή (overfitting). Η αντικειμενική συνάρτηση συνδυάζει τη συνάρτηση απώλειας με έναν όρο κανονικοποίησης που τιμωρεί την πολυπλοκότητα των δέντρων, εξισορροπώντας έτσι την ακρίβεια με την απλότητα του μοντέλου.

Ο XGBoost χρησιμοποιεί επίσης μια αποτελεσματική μέθοδο για τη δημιουργία δέντρων, χρησιμοποιώντας μια ευριστική μέθοδο που επιλέγει τις καλύτερες διασπάσεις με

βάση το κέρδος στην αντικειμενική συνάρτηση. Αυτή η προσέγγιση όχι μόνο επιταχύνει τη διαδικασία κατασκευής των δέντρων αλλά και βελτιώνει την απόδοση του αλγορίθμου. Επιπλέον, ο XGBoost είναι σε θέση να χειρίζεται αποτελεσματικά τα δεδομένα που λείπουν, μαθαίνοντας την καλύτερη κατεύθυνση για τις διασπάσεις όταν λείπουν δεδομένα.

Η επεκτασιμότητα του αλγορίθμου είναι ένα άλλο σημαντικό πλεονέκτημα. Ο XGBoost υποστηρίζει παράλληλους και κατανεμημένους υπολογισμούς, επιτρέποντάς του να χειρίζεται αποτελεσματικά μεγάλα σύνολα δεδομένων και δεδομένα υψηλών διαστάσεων. Αυτή η επεκτασιμότητα τον καθιστά κατάλληλο τόσο για εφαρμογές μικρής όσο και μεγάλης κλίμακας.



Εικόνα 7: XGBoost

2.3.6 Νευρωνικά Δίκτυα

Στη μηχανική μάθηση, ένα νευρωνικό δίκτυο (Neural Network) είναι ένα υπολογιστικό μοντέλο εμπνευσμένο από τα βιολογικά νευρωνικά δίκτυα που βρίσκονται στους εγκεφάλους των ζώων.

Ένα ΝΔ αποτελείται από διασυνδεδεμένες μονάδες που ονομάζονται τεχνητοί νευρώνες, οι οποίες προσομοιώνουν τη συμπεριφορά των νευρώνων στον εγκέφαλο. Αυτοί οι νευρώνες συνδέονται με συνδέσεις που αντιπροσωπεύουν συνάψεις. Κάθε τεχνητός νευρώνας

λαμβάνει σήματα εισόδου από άλλους νευρώνες, επεξεργάζεται αυτές τις εισόδους και στη συνέχεια μεταδίδει ένα σήμα εξόδου σε άλλους νευρώνες. Αυτό το σήμα εξόδου είναι ένας πραγματικός αριθμός, ο οποίος καθορίζεται με την εφαρμογή μιας μη γραμμικής συνάρτησης, γνωστής ως συνάρτηση ενεργοποίησης, στο σταθμισμένο άθροισμα των εισόδων του. Τα βάρη σε αυτές τις συνδέσεις προσαρμόζονται κατά τη διάρκεια της διαδικασίας εκπαίδευσης για να βελτιωθεί η απόδοση του δικτύου.

Οι νευρώνες σε ένα νευρωνικό δίκτυο οργανώνονται σε επίπεδα, με κάθε επίπεδο να εκτελεί ενδεχομένως διαφορετικούς τύπους μετασχηματισμών στις εισόδους του. Η ροή των σημάτων ξεκινά από το στρώμα εισόδου, προχωρά μέσω ενός ή περισσότερων ενδιάμεσων στρωμάτων (γνωστά ως κρυφά στρώματα) και φτάνει στο στρώμα εξόδου. Ένα νευρωνικό δίκτυο αναφέρεται ως βαθύ νευρωνικό δίκτυο (Deep Neural Network) εάν περιλαμβάνει τουλάχιστον δύο κρυφά στρώματα.

Κάθε τεχνητός νευρώνας υπολογίζει ένα σταθμισμένο άθροισμα των εισόδων του και στη συνέχεια εφαρμόζει μια συνάρτηση ενεργοποίησης. Μαθηματικά, για έναν νευρώνα i με εισόδους $x_1, x_{2,\dots,} x_n$), βάρη w_i και μεροληψία b, η έξοδος a_i δίνεται από:

$$z_i = \sum_{j=1}^{n} w_j x_j + b$$

$$a_i = \varphi(z_i)$$

όπου φ η συνάρτηση ενεργοποίησης.

Οι συναρτήσεις ενεργοποίησης εισάγουν μη γραμμικότητα στο δίκτυο, επιτρέποντάς του να μοντελοποιεί σύνθετες σχέσεις. Οι συνήθεις συναρτήσεις ενεργοποίησης περιλαμβάνουν:

1. **Σιγμοειδή Συνάρτηση:**

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

2. **Συνάρτηση υπερβολικής εφαπτομένης (tanh):**

$$\tanh(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$$

3. **Rectified Linear Unit (ReLU) Function:**

$ReLU(z) = \max(0, z)$

Σε ένα νευρωνικό δίκτυο, η προς τα εμπρός διάδοση υπολογίζει την έξοδο για μια δεδομένη είσοδο περνώντας την μέσα από τα επίπεδα. Για ένα δίκτυο με L στρώματα, η έξοδος μπορεί να υπολογιστεί ως εξής:

1. Για το κ-οστό στρώμα, η τιμή προ-ενεργοποίησης είναι:
   $z^{(k)} = w^{(k)} a^{(k-1)} + b^{(k)}$

2. Η τιμή ενεργοποίησης είναι:
   $a^{(k)} = \varphi(z^{(k)})$

3. Η έξοδος του τελευταίου στρώματος είναι:
   $\hat{y} = a^{(L)}$

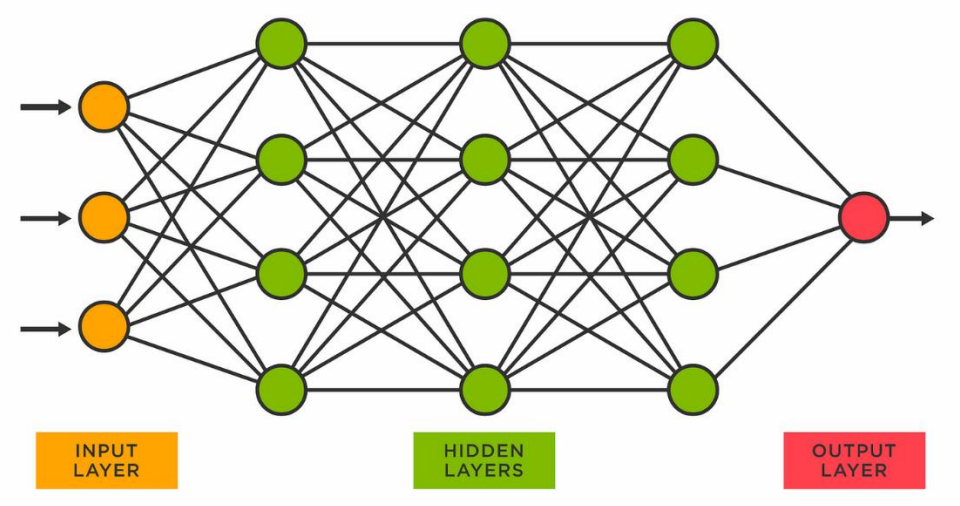
Η συνάρτηση απώλειας ποσοτικοποιεί το σφάλμα μεταξύ της προβλεπόμενης εξόδου $\hat{y}$ και του πραγματικού στόχου y. Οι συνήθεις συναρτήσεις απώλειας περιλαμβάνουν:

4. **Μέσο Τετραγωνικό Σφάλμα (για παλινδρόμηση)**

$$MSE = \frac{1}{m}\sum_{i=1}^{m}(\hat{y}_i - y_i)^2$$

5. **Διασταυρούμενη απώλεια εντροπίας (Cross-Entropy Loss, για ταξινόμηση)**

$$\text{CrossEntropy} = -\frac{1}{m}\sum_{i=1}^{m}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)]$$

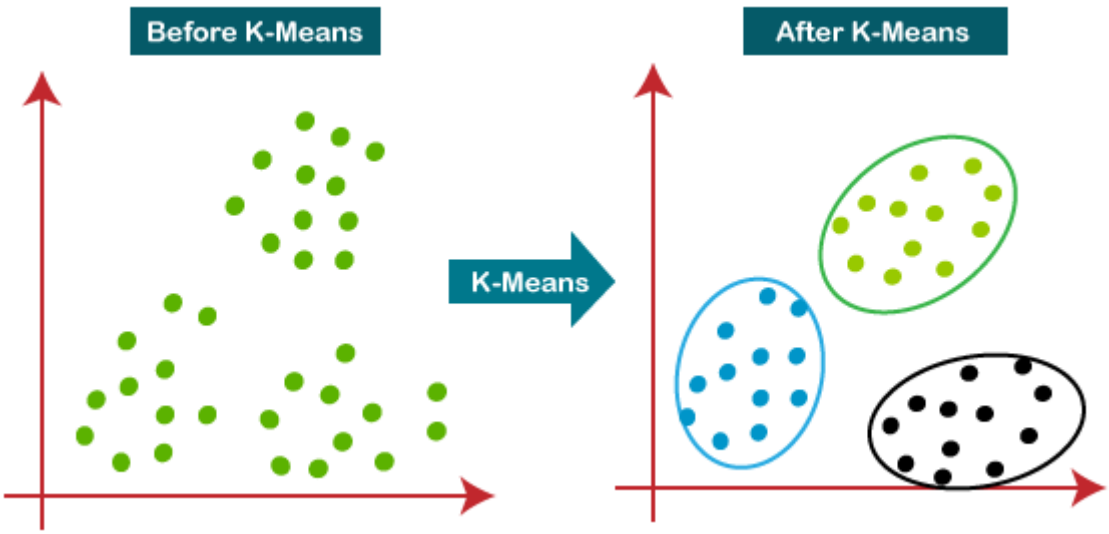


Εικόνα 8: Νευρωνικό Δίκτυο

2.4 Μη Επιβλεπόμενη Μάθηση

Η μη επιβλεπόμενη μάθηση αποτελεί ένα κρίσιμο κομμάτι της μηχανικής μάθησης, καθώς μας επιτρέπει να ανακαλύπτουμε κρυμμένες δομές και μοτίβα στα δεδομένα χωρίς την ανάγκη ετικετών ή προκαθορισμένων αποτελεσμάτων. Σε πολλές περιπτώσεις πραγματικού κόσμου, η απόκτηση δεδομένων με ετικέτες μπορεί να είναι δαπανηρή, χρονοβόρα ή και αδύνατη. Αυτό ισχύει ιδιαίτερα όταν υπάρχουν μεγάλες ποσότητες δεδομένων που δεν συνοδεύονται από προκαθορισμένες κατηγορίες. Μέσω της μη επιβλεπόμενης μάθησης, μπορούμε να εξερευνήσουμε την εγγενή δομή των δεδομένων και να ανακαλύψουμε σημαντικές σχέσεις που θα μπορούσαν να περάσουν απαρατήρητες.

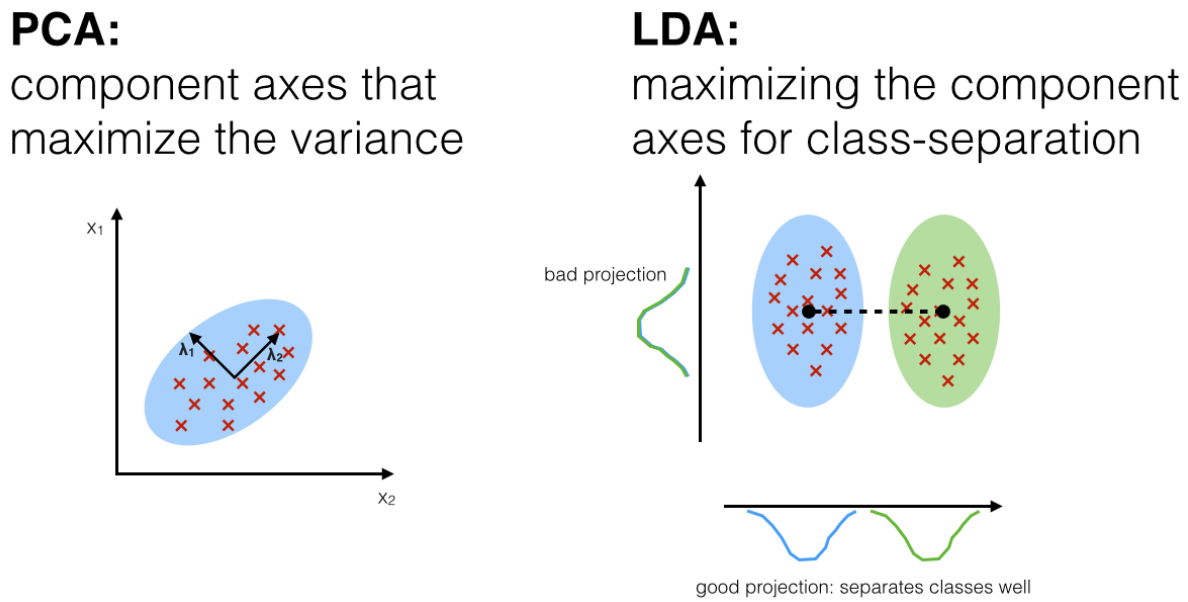
Ένας από τους βασικούς λόγους χρήσης της μη επιβλεπόμενης μάθησης είναι η ικανότητά της να εντοπίζει υποκείμενες ομαδοποιήσεις ή συστάδες στα δεδομένα, βοηθώντας στην ταυτοποίηση φυσικών δομών εντός ενός συνόλου δεδομένων. Για παράδειγμα, τεχνικές όπως οι αλγόριθμοι ομαδοποίησης (clustering), όπως το K-Means ή το DBSCAN, μας επιτρέπουν να χωρίσουμε τα δεδομένα σε διακριτές ομάδες, κάτι που μπορεί να είναι ζωτικής σημασίας σε εφαρμογές όπως η τμηματοποίηση πελατών, η ανίχνευση ανωμαλιών ή η αναγνώριση παρόμοιων μοτίβων στα βιολογικά δεδομένα. Αυτοί οι αλγόριθμοι μπορούν συχνά να αποκαλύψουν πληροφορίες που είναι δύσκολο να εντοπιστούν χειροκίνητα ή μέσω επιβλεπόμενων προσεγγίσεων, ειδικά σε σύνολα δεδομένων όπου δεν υπάρχει σαφώς καθορισμένο αποτέλεσμα.



Εικόνα 9: Ομαδοποίηση K- μέσων

Ένας άλλος σημαντικός λόγος για τη χρήση της μη επιβλεπόμενης μάθησης είναι η μείωση των διαστάσεων, που είναι ζωτικής σημασίας όταν έχουμε να κάνουμε με δεδομένα

υψηλής διαστατικότητας που μπορεί να υπερφορτώσουν τα μοντέλα και να οδηγήσουν σε προβλήματα, όπως η υπερπροσαρμογή (overfitting) ή η υπολογιστική αναποτελεσματικότητα. Μέθοδοι όπως η Ανάλυση Κύριων Συνιστωσών (PCA) και η Γραμμική Διακριτή Ανάλυση (LDA) επιτρέπουν τη μείωση της πολυπλοκότητας των δεδομένων, ενώ διατηρούν σημαντικές πληροφορίες. Αυτό μπορεί να είναι ιδιαίτερα χρήσιμο στην απεικόνιση δεδομένων, στη συμπίεση συνόλων δεδομένων ή στην προετοιμασία δεδομένων για περαιτέρω επιβλεπόμενες διαδικασίες, εξασφαλίζοντας ότι αναδεικνύονται οι πιο σημαντικές μεταβλητές.



Εικόνα 10:Διαφορά PCA και LDA

2.5 Προεπεξεργασία των δεδομένων

Η προεπεξεργασία δεδομένων περιλαμβάνει μια σειρά από βήματα που λαμβάνονται πριν από την πραγματική διαδικασία ανάλυσης δεδομένων. Αυτή η διαδικασία λειτουργεί ως μια μετασχηματιστική λειτουργία T που μετατρέπει τα αρχικά διανύσματα δεδομένων X_{ik} από τον πραγματικό κόσμο σε ένα σύνολο νέων διανυσμάτων δεδομένων Y_{ij}. Οι στόχοι αυτού του μετασχηματισμού είναι:

1. **Διατήρηση Πολύτιμων Πληροφοριών**: Τα μετασχηματισμένα δεδομένα Y_{ij} πρέπει να διατηρούν τις κρίσιμες πληροφορίες που υπάρχουν στα αρχικά δεδομένα X_{ik}. Αυτές οι πολύτιμες πληροφορίες συνήθως περιλαμβάνουν στοιχεία γνώσης,

όπως σημαντικά μοτίβα, που η ανάλυση δεδομένων στοχεύει να ανακαλύψει και να παρουσιάσει με νόημα.

2. **Αντιμετώπιση Προβλημάτων στα Αρχικά Δεδομένα**: Τα νέα διανύσματα δεδομένων Y_{ij} πρέπει να μετριάσουν τουλάχιστον ένα από τα προβλήματα που υπάρχουν στα αρχικά διανύσματα δεδομένων X_{ik}.
3. **Βελτίωση Χρησιμότητας**: Τα μετασχηματισμένα δεδομένα Y_{ij} πρέπει να είναι πιο χρήσιμα για ανάλυση σε σχέση με τα αρχικά δεδομένα X_{ik}.

Όπου,

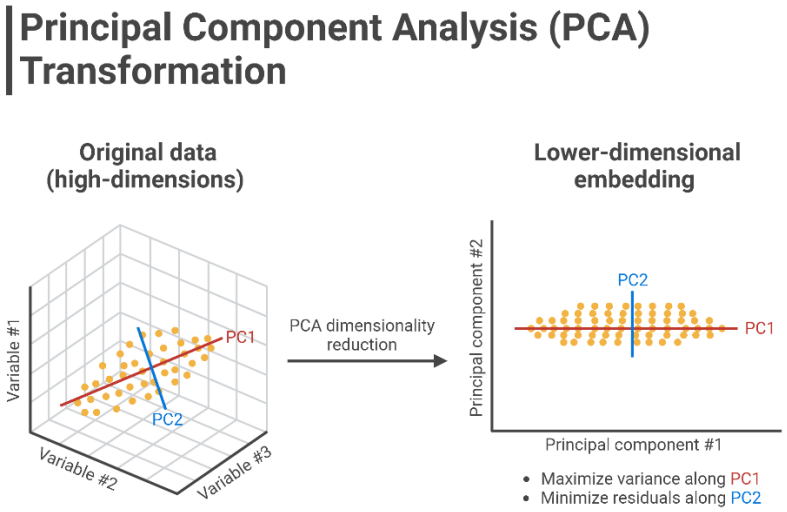
- $i = 1, \dots, n$ όπου n είναι ο αριθμός των αντικειμένων δεδομένων,
- $j = 1, \dots, m$ όπου m είναι ο αριθμός των χαρακτηριστικών μετά την προεπεξεργασία,
- $k = 1, \dots, I$ όπου I είναι ο αριθμός των χαρακτηριστικών πριν την προεπεξεργασία.

Πάντα υπάρχουν προβλήματα με τα δεδομένα του πραγματικού κόσμου. Η φύση και η σοβαρότητα αυτών των προβλημάτων εξαρτώνται από πολλούς παράγοντες που είναι μερικές φορές πέρα από τον έλεγχο των ανθρώπινων χειριστών. Η ανησυχία μας οφείλεται στις επιπτώσεις αυτών των προβλημάτων στα αποτελέσματα της ανάλυσης δεδομένων, με στόχο είτε τη διόρθωση των προβλημάτων των δεδομένων εκ των προτέρων είτε την αναγνώριση των επιπτώσεων των προβλημάτων δεδομένων στα αποτελέσματα. Τα προβλήματα δεδομένων μπορούν να ταξινομηθούν σε τρεις κατηγορίες: υπερβολικά πολλά δεδομένα, υπερβολικά λίγα δεδομένα και κατακερματισμένα δεδομένα.

Υπερβολικά πολλά δεδομένα: Η ύπαρξη υπερβολικά πολλών δεδομένων μπορεί να δυσχεραίνει την ανάλυση, καθώς μπορεί να περιέχουν πλεονάζουσες ή μη σχετικές πληροφορίες. Αυτό οδηγεί σε αυξημένη πολυπλοκότητα και απαιτήσεις αποθήκευσης και επεξεργασίας. Η υπερβολική ποσότητα δεδομένων μπορεί επίσης να περιπλέξει την εξαγωγή των ουσιωδών μοτίβων και τάσεων. Για να αντιμετωπίσουμε αυτό το ζήτημα, επιλέξαμε να χρησιμοποιήσουμε τις ακόλουθες δύο μεθόδους:

- **Principal Component Analysis (PCA)**: Το PCA μειώνει τις διαστάσεις μεγάλων συνόλων δεδομένων αναγνωρίζοντας τα πιο σημαντικά συστατικά, βοηθώντας έτσι στην εξάλειψη των πλεοναζόντων ή λιγότερο σημαντικών χαρακτηριστικών.

- **Linear Discriminant Analysis (LDA)**: Το LDA χρησιμοποιείται επίσης για τη μείωση των διαστάσεων, αλλά επικεντρώνεται στη μεγιστοποίηση της διακριτότητας μεταξύ των διαφορετικών κατηγοριών. Μειώνει τα δεδομένα σε ένα μικρότερο σύνολο χαρακτηριστικών που διακρίνουν καλύτερα τις κατηγορίες.



Εικόνα 11: Μείωση διαστάσεων με PCA

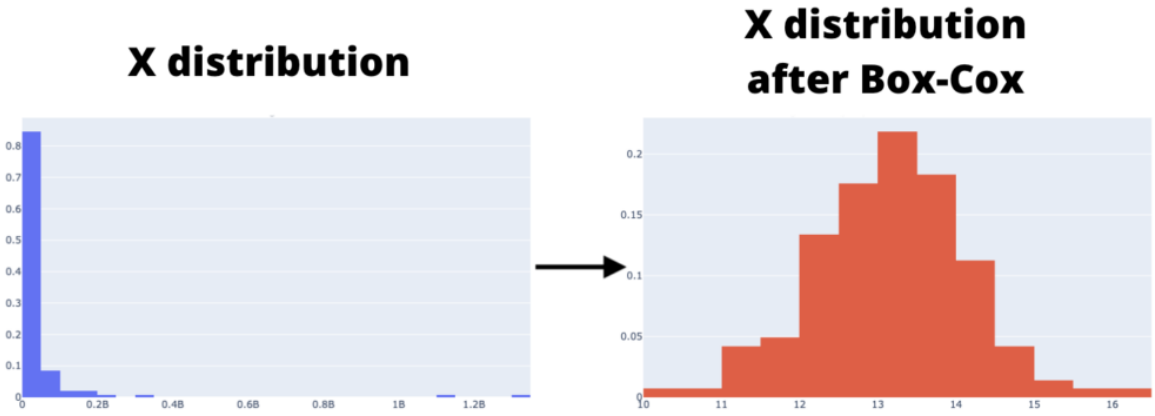
Υπερβολικά λίγα δεδομένα: Όταν υπάρχουν υπερβολικά λίγα δεδομένα, η ανάλυση μπορεί να είναι ανεπαρκής και να μην καταλήγει σε αξιόπιστα αποτελέσματα. Η έλλειψη επαρκών δεδομένων μπορεί να οδηγήσει σε ασταθείς υπολογισμούς και συμπεράσματα που δεν είναι αντιπροσωπευτικά του γενικότερου πληθυσμού ή φαινομένου. Μερικές λύσεις για αυτό το πρόβλημα, που δεν συναντήθηκε στα δικά μας δεδομένα, είναι:

- **Μεταβιβαστική μάθηση:** Χρησιμοποιούμε προ-εκπαιδευμένα μοντέλα ή σύνολα δεδομένων από συναφείς εργασίες ή τομείς. Για παράδειγμα, στη μηχανική μάθηση, ένα μοντέλο που έχει εκπαιδευτεί σε ένα μεγάλο, παρόμοιο σύνολο δεδομένων μπορεί να ρυθμιστεί στο μικρότερο σύνολο δεδομένων σας, αξιοποιώντας τα μοτίβα και τα χαρακτηριστικά που έχουν μάθει.
- **Cross-Validation:** Χρησιμοποιούμε ισχυρές τεχνικές cross-validation, όπως η k-fold ή η διασταυρούμενη επικύρωση leave-one-out, για να αξιοποιήσετε με τον καλύτερο δυνατό τρόπο τα περιορισμένα δεδομένα για την εκπαίδευση και την επικύρωση του

μοντέλου. Αυτό συμβάλλει στη διασφάλιση ότι το μοντέλο είναι καλά αξιολογημένο και λιγότερο επιρρεπές σε υπερπροσαρμογή.

Κατακερματισμένα δεδομένα: Τα κατακερματισμένα δεδομένα εμφανίζονται όταν τα δεδομένα είναι ατελή ή ασυνεπή, περιέχουν κενά, ελλιπείς πληροφορίες ή διαφορετικές μορφές. Αυτό μπορεί να προκύψει από διάφορες πηγές ή ασυνεπείς μεθόδους συλλογής δεδομένων. Τα κατακερματισμένα δεδομένα καθιστούν δύσκολη την εξαγωγή σαφών και συνεκτικών συμπερασμάτων και απαιτούν πρόσθετη προεπεξεργασία για την ενοποίησή τους. Για να αντιμετωπίσουμε αυτό το ζήτημα, επιλέξαμε να χρησιμοποιήσουμε τις ακόλουθες δύο μεθόδους:

- **Τυποποίηση**: Αυτή η τεχνική περιλαμβάνει την τυποποίηση του εύρους των ανεξάρτητων μεταβλητών ή χαρακτηριστικών των δεδομένων, συχνά μετατρέποντας τα δεδομένα σε μια συγκεκριμένη κλίμακα, όπως 0 έως 1 ή -1 έως 1. Είναι ιδιαίτερα χρήσιμη όταν τα δεδομένα προέρχονται από διαφορετικές πηγές με διάφορες κλίμακες, διασφαλίζοντας ότι κάθε χαρακτηριστικό συμβάλλει ισότιμα στην ανάλυση.
- **Κανονικοποίηση:** Η κανονικοποίηση αναφέρεται στην προσαρμογή των δεδομένων ώστε να ακολουθούν μια πιο κανονική κατανομή. Για να επιτύχουμε αυτόν τον στόχο, εφαρμόζουμε τη μετατροπή Box-Cox. Η μετατροπή Box-Cox είναι μια στατιστική μέθοδος που μετασχηματίζει τα δεδομένα ώστε να γίνουν πιο κανονικά κατανεμημένα, βοηθώντας στη βελτίωση της ποιότητας των δεδομένων και στην αύξηση της ακρίβειας των αλγορίθμων ανάλυσης που υποθέτουν κανονική κατανομή των δεδομένων.



Εικόνα 12: Μετασχηματισμός Box-Cox

2.6 Μέτρα Απόδοσης

Η ακρίβεια (accuracy) είναι συχνά η πιο χρησιμοποιούμενη μετρική για την αξιολόγηση μοντέλων δυαδικής ταξινόμησης λόγω της απλότητάς της και της άμεσης ερμηνείας της—μετρά το ποσοστό των σωστών προβλέψεων επί του συνόλου των προβλέψεων. Ωστόσο, η ακρίβεια μπορεί να είναι ανεπαρκής σε περιπτώσεις όπου υπάρχει σημαντική ανισορροπία μεταξύ των κατηγοριών, καθώς μπορεί να δώσει μια παραπλανητική αίσθηση της απόδοσης του μοντέλου. Για παράδειγμα, εάν μια κατηγορία είναι πολύ πιο επικρατούσα από την άλλη, ένα μοντέλο που προβλέπει πάντα την πλειονότητα των περιπτώσεων θα επιτυγχάνει υψηλή ακρίβεια, παρά το ότι είναι αναποτελεσματικό για την μειονοτική κατηγορία.

Για αυτόν τον λόγο, είναι επίσης σημαντικό να χρησιμοποιούμε μετρικές όπως η πιστότητα (precision), η ανάκληση (recall) και ο δείκτης F1. Η πιστότητα εστιάζει στην ακρίβεια των θετικών προβλέψεων, η ανάκληση μετρά την ικανότητα του μοντέλου να εντοπίζει όλες τις θετικές περιπτώσεις και ο δείκτης F1 προσφέρει ένα αρμονικό μέσο της πιστότητας και της ανάκλησης, παρέχοντας μια ισορροπημένη άποψη για την απόδοση του μοντέλου, ιδιαίτερα όταν η κατανομή των κατηγοριών είναι ασύμμετρη. Χρησιμοποιώντας αυτές τις μετρικές μαζί, αποκτάται μια πιο ολοκληρωμένη κατανόηση των δυνατοτήτων και των αδυναμιών του μοντέλου.

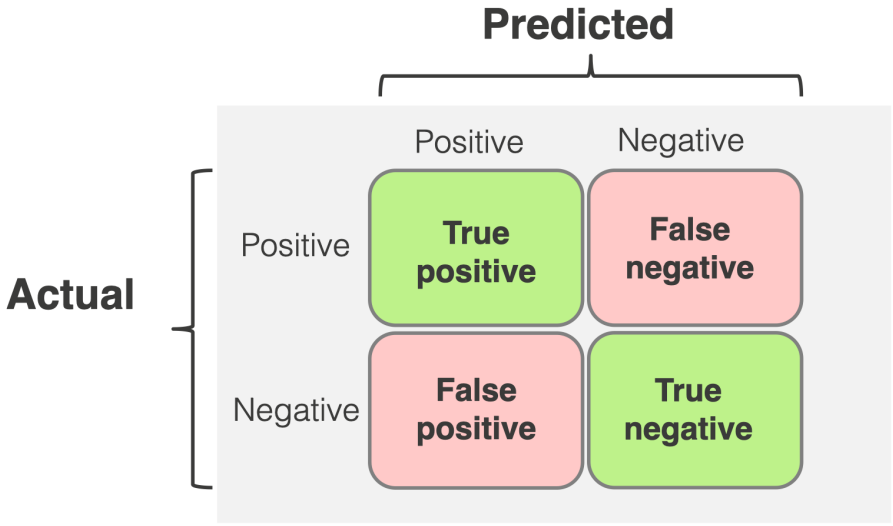
Αν κατηγοριοποιήσουμε τα αποτελέσματα σε τέσσερις ομάδες - αληθώς θετικά (TP), αληθώς αρνητικά (TN), ψευδώς θετικά (FP), και ψευδώς αρνητικά (FN) – οι μετρικές μπορούν να εκφραστούν μαθηματικά ως εξής:

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

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

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

$$F1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$$



Εικόνα 13: Κατηγοριοποίηση προβλέψεων

3. JULIA

3.1 Γιατί Julia;

Η μηχανική μάθηση προωθεί την πρόοδο σε πολλούς επιστημονικούς και μηχανολογικούς κλάδους. Η χρήση μιας απλής και αποτελεσματικής γλώσσας προγραμματισμού θα μπορούσε να ενισχύσει σημαντικά την εφαρμογή της μηχανικής μάθησης σε αυτούς τους τομείς. Η Python, το MATLAB και η C/C++ είναι σήμερα οι πιο δημοφιλείς γλώσσες για την ανάπτυξη αλγορίθμων μηχανικής μάθησης. Ωστόσο, οι γλώσσες αυτές συχνά δυσκολεύονται να εξισορροπήσουν τόσο την αποτελεσματικότητα όσο και την ευκολία χρήσης. Η Julia, μια γλώσσα προγραμματισμού ανοικτού κώδικα ειδικά σχεδιασμένη για υπολογιστές υψηλών επιδόσεων, προσφέρει μια λύση παρέχοντας τόσο ταχύτητα όσο και απλότητα.

Η Julia είναι μια σύγχρονη, εκφραστική και υψηλής απόδοσης γλώσσα προγραμματισμού για επιστημονικούς υπολογισμούς και επεξεργασία δεδομένων. Η ανάπτυξη της ξεκίνησε το 2009, και η τρέχουσα σταθερή έκδοση από τον Ιούνιο 2024 είναι η v1.10.4. Παρόλο που αυτός ο χαμηλός αριθμός έκδοσης υποδηλώνει ότι η γλώσσα εξακολουθεί να

αναπτύσσεται με ταχείς ρυθμούς, είναι αρκετά σταθερή ώστε να επιτρέπει την ανάπτυξη ερευνητικού κώδικα. Η γραμματική της Julia είναι ευανάγνωστη, παρόμοια με εκείνη της MATLAB ή της Python, και μπορεί να προσεγγίσει την τη γλώσσα C/C++ σε επιδόσεις με τη μεταγλώττιση σε πραγματικό χρόνο. Στο Επιπλέον, η Julia είναι μια δωρεάν γλώσσα ανοικτού κώδικα που τρέχει σε όλα τα δημοφιλή λειτουργικά συστήματα.

Εικόνα 14: Προβλήματα της γλώσσας Julia

3.2 Αλγόριθμοι και Βιβλιοθήκες

Η Julia έχει αναπτύξει ένα ισχυρό οικοσύστημα βιβλιοθηκών για μηχανική μάθηση που καλύπτουν ένα ευρύ φάσμα αλγορίθμων. Αυτές οι βιβλιοθήκες προσφέρουν αποδοτικές και κλιμακούμενες λύσεις για μοντελοποίηση, καθιστώντας τη Julia μια εξαιρετική εναλλακτική γλώσσα για προγνωστική ανάλυση και μηχανική μάθηση. Ακολουθεί μια επισκόπηση των κύριων βιβλιοθηκών και των αλγορίθμων που υποστηρίζουν:

❖ **MLJ.jl**

Το MLJ.jl είναι ένα ισχυρό πλαίσιο που ενοποιεί διάφορους αλγορίθμους μηχανικής μάθησης κάτω από ένα ενιαίο API, καθιστώντας την υλοποίηση και την αξιολόγηση μοντέλων εύκολη. Οι αλγόριθμοι που υποστηρίζονται περιλαμβάνουν:

- Λογιστική Παλινδρόμηση (Logistic Regression): Κατάλληλη για δυαδική ταξινόμηση, όπως η πρόβλεψη κινδύνου εμφάνισης διαβήτη.

- Random Forest: Ένας συναρμολογημένος αλγόριθμος βασισμένος σε πολλαπλά δέντρα απόφασης.
- Support Vector Machine (SVM): Μπορεί να χρησιμοποιηθεί για ταξινόμηση και παλινδρόμηση.
- Γραμμική Παλινδρόμηση (Linear Regression): Για την πρόβλεψη συνεχών τιμών.

❖ **DecisionTree.jl**

Η DecisionTree.jl είναι μια βιβλιοθήκη που επικεντρώνεται στην υλοποίηση δέντρων απόφασης και συναρμολογημένων αλγορίθμων. Περιλαμβάνει:

- Δέντρα Αποφάσεων (Decision Trees): Χρήσιμα για τόσο ταξινόμηση όσο και παλινδρόμηση.
- Random Forest: Ένας από τους πιο δημοφιλείς αλγόριθμους για ταξινομήσεις με πολλαπλά δέντρα.
- Gradient Boosted Trees: Χρήσιμο για προβλήματα όπου απαιτείται μεγαλύτερη ακρίβεια από απλά δέντρα.

❖ **Flux.jl**

Η Flux.jl είναι η κύρια βιβλιοθήκη για βαθιά μάθηση στην Julia. Παρέχει ένα απλό και ευέλικτο API για την κατασκευή νευρωνικών δικτύων και άλλων προηγμένων μοντέλων. Υποστηρίζει:

- Νευρωνικά Δίκτυα (Neural Networks): Ιδανικά για την αναγνώριση σύνθετων μοτίβων και τη μη γραμμική μοντελοποίηση.
- Βαθιά Μάθηση (Deep Learning): Υποστηρίζει βαθιά μοντέλα με πολλαπλά κρυφά επίπεδα για ανάλυση δεδομένων μεγάλης κλίμακας.

Πέρα από αυτά τα κυρίαρχα πακέτα, υπάρχουν πολλές άλλες βιβλιοθήκες, οι οποίες προσφέρουν εκτεταμένες λειτουργίες και εναλλακτικές υλοποιήσεις αλγόριθμων. Η ανοιχτή φύση της Julia έχει δημιουργήσει μια ακμάζουσα κοινότητα όπου νέες βιβλιοθήκες και εργαλεία αναπτύσσονται, δοκιμάζονται και μοιράζονται συνεχώς στο GitHub. Μερικά αξιόλογα παραδείγματα περιλαμβάνουν:

- TSML.jl (Time Series Machine Learning): Παρέχει εργαλεία για τη διαχείριση δεδομένων χρονοσειρών και την υλοποίηση μοντέλων μηχανικής μάθησης ειδικά για χρονοσειρές.
- Turing.jl: Μια βιβλιοθήκη πιθανολογικού προγραμματισμού που επιτρέπει στους χρήστες να δημιουργούν και να εκτελούν Μπεϋζιανά μοντέλα, χρήσιμη για εργασίες που απαιτούν εκτίμηση αβεβαιότητας.
- Knet.jl: Μια εναλλακτική λύση στο Flux.jl για βαθιά μάθηση, προσφέροντας μεγαλύτερο έλεγχο και ευελιξία στην κατασκευή νευρωνικών δικτύων.
- EvoTrees.jl: Μια γρήγορη υλοποίηση gradient boosting decision trees, βελτιστοποιημένη για την Julia, προσφέροντας μια εναλλακτική λύση στο XGBoost.

3.3 Η Διεπαφή Dash

Η δημιουργία μιας διεπαφής (interface) που επιτρέπει στους χρήστες να αλληλεπιδρούν με τα μοντέλα μηχανικής μάθησης είναι ένα κρίσιμο βήμα για την πρακτική εφαρμογή αυτών των μοντέλων. Για την παρούσα διατριβή, δημιουργήθηκε μια διαδικτυακή διεπαφή χρησιμοποιώντας το Dash.jl, το οποίο είναι αντίστοιχο πλαίσιο στη Julia, του δημοφιλούς Dash στην Python. Η διεπαφή αυτή επιτρέπει στους χρήστες να εισάγουν προσωπικά δεδομένα υγείας και να λαμβάνουν προβλέψεις για τον κίνδυνο εμφάνισης διαβήτη.

Για την υλοποίηση της διεπαφής στην Julia, χρησιμοποιήθηκε το Dash.jl σε συνδυασμό με τις βιβλιοθήκες μηχανικής μάθησης που αναλύθηκαν προηγουμένως. Το Dash.jl επιτρέπει τη σύνδεση του frontend με τα μοντέλα της Julia, δημιουργώντας έτσι μια ομαλή εμπειρία για τον τελικό χρήστη.

Ένα βασικό πλεονέκτημα του Dash.jl είναι ότι οι διεπαφές που δημιουργούνται μπορούν να τρέχουν τοπικά στον υπολογιστή του χρήστη, αλλά είναι και έτοιμες για μελλοντική κλιμάκωση σε διαδικτυακούς servers αν χρειαστεί. Αυτό σημαίνει ότι η εφαρμογή μπορεί να επεκταθεί για χρήση σε κλινικά περιβάλλοντα ή σε έρευνα μεγάλης κλίμακας.

4. ΥΛΟΠΟΙΗΣΗ

4.1 Σκοπός

Αν και ο διαβήτης μπορεί να μην είναι ένας τομέας που χρειάζεται άμεσα μοντέλα μηχανικής μάθησης για πρόβλεψη—καθώς πολλοί από τους παράγοντες κινδύνου του, όπως το BMI, η ηλικία και τα επίπεδα γλυκόζης, είναι καλά τεκμηριωμένοι—αυτή η εργασία επιδεικνύει πώς η τεχνητή νοημοσύνη (AI) μπορεί να βελτιώσει τον τρόπο με τον οποίο εντοπίζουμε άτομα που διατρέχουν υψηλό κίνδυνο. Η έγκαιρη διάγνωση του διαβήτη παραμένει κρίσιμη για την πρόληψη σοβαρών επιπλοκών, τη μείωση του κόστους υγείας και τη βελτίωση των αποτελεσμάτων των ασθενών. Με την αξιοποίηση της μηχανικής μάθησης, μπορούμε όχι μόνο να αυτοματοποιήσουμε και να βελτιώσουμε τη διαδικασία πρόβλεψης, αλλά και να παρέχουμε πολύτιμες πληροφορίες σχετικά με τους συγκεκριμένους παράγοντες που οδηγούν σε αυτές τις προβλέψεις. Αυτό επιτρέπει στους επαγγελματίες υγείας να εστιάσουν την προσοχή τους στα άτομα που διατρέχουν τον μεγαλύτερο κίνδυνο και να προτεραιοποιήσουν τις πρώιμες παρεμβάσεις πιο αποτελεσματικά.

Επιπλέον, αυτή η έρευνα υπογραμμίζει τη σημασία της εξηγήσιμης τεχνητής νοημοσύνης στα μοντέλα, εξασφαλίζοντας ότι οι προβλέψεις είναι διαφανείς και εφαρμόσιμες. Η δυνατότητα κατανόησης του γιατί ένας ασθενής θεωρείται υψηλού κινδύνου ενδυναμώνει τους ιατρικούς επαγγελματίες να λαμβάνουν τεκμηριωμένες αποφάσεις και να προσαρμόζουν τα σχέδια θεραπείας αναλόγως.

Παρόλο που η εφαρμογή της εξηγήσιμης τεχνητής νοημοσύνης στην πρόβλεψη του διαβήτη είναι σημαντική, οι μεθοδολογίες και οι προσεγγίσεις που παρουσιάζονται σε αυτήν την εργασία μπορούν να ανοίξουν τον δρόμο για πιο σύνθετους ιατρικούς τομείς, όπως η ανάλυση ιατρικών εικόνων και η ακτινολογία, όπου η προγνωστική μοντελοποίηση είναι σημαντικά πιο δύσκολη. Με την επίδειξη της αξίας της μηχανικής μάθησης σε έναν σχετικά απλό ιατρικό τομέα, αυτή η εργασία θέτει τις βάσεις για τη χρήση της σε πιο απαιτητικά διαγνωστικά καθήκοντα στον χώρο της υγείας.

4.2 Σετ Δεδομένων

Το σετ δεδομένων που χρησιμοποιήθηκε είναι από το Kaggle και εστιάζει σε ιατρικά και φυσικά χαρακτηριστικά που μπορούν να χρησιμοποιηθούν για την αξιολόγηση παραγόντων κινδύνου για τον διαβήτη. Για την κατηγοριοποίηση ενός ασθενή ως διαβητικό

ή μη θα χρησιμοποιήσουμε την μεταβλητή glhyb που είναι το επίπεδο Γλυκοζυλιωμένης αιμοσφαιρίνης (HbA1c). Για τιμές της μεταβλητής άνω του 6.5 ο ασθενής θα θεωρείται διαβητικός ενώ για χαμηλότερες ασφαλής. Οι υπόλοιπες μεταβλητές θα χρησιμοποιηθούν για την πρόβλεψη της προαναφερόμενης και επεξηγούνται παρακάτω.

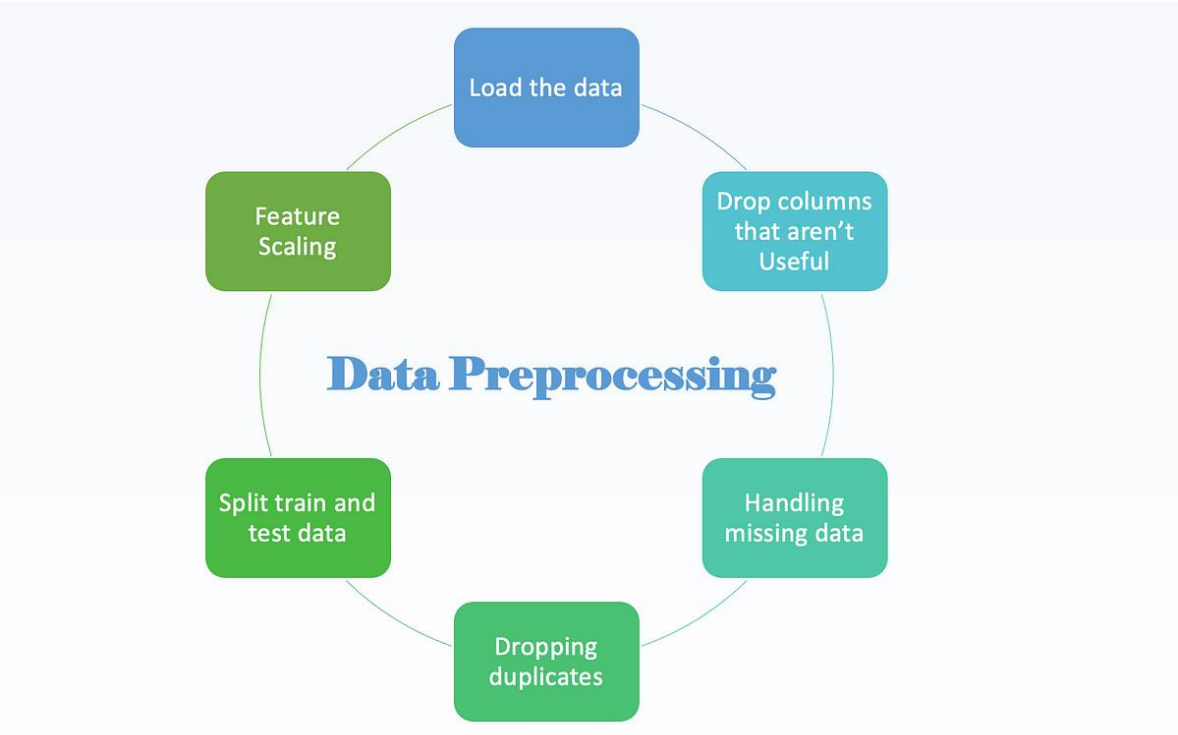
- **id:** Μοναδικό αναγνωριστικό για κάθε ασθενή
- **chol:** Επίπεδα χοληστερόλης
- **stab.glu:** Σταθερά επίπεδα γλυκόζης.
- **hdl**: Λιποπρωτεΐνη υψηλής πυκνότητας (καλή χοληστερόλη).
- **ratio**: Αναλογία συνολικής χοληστερόλης προς HDL.
- **glyhb**: Γλυκοζυλιωμένη αιμοσφαιρίνη (HbA1c), δείκτης του επιπέδου σακχάρου στο αίμα με την πάροδο του χρόνου.
- **location**: Τοποθεσία (Louisa ή Buckingham).
- **age**: Ηλικία του ασθενούς.
- **gender**: Φύλο του ασθενούς(αρσενικό ή θηλυκό).
- **height**: Ύψος του ασθενούς (σε ίντσες).
- **weight**: Βάρος του ασθενούς (σε λίβρες).
- **frame**: Μέγεθος σώματος (μικρό, μεσαίο, μεγάλο).
- **bp.1s**: Πρώτη συστολική αρτηριακή πίεση.
- **bp.1d**: Πρώτη διαστολική αρτηριακή πίεση.
- **bp.2s**: Δεύτερη συστολική αρτηριακή πίεση.
- **bp.2d**: Δεύτερη διαστολική αρτηριακή πίεση.
- **waist**: Περίμετρος μέσης(σε ίντσες).
- **hip:** Περίμετρος γοφών(σε ίντσες).
- **time.ppn**: Χρόνος από το τελευταίο γεύμα ή χρόνος σε συγκεκριμένο ιατρικό πλαίσιο.

Για να έχουμε μια καλύτερη εικόνα των δεδομένων θα παρουσιάσουμε σε έναν πίνακα τον τύπο δεδομένων, το πλήθος, το πλήθος διακεκριμένων τιμών, το πλήθος ελλειπόντων δεδομένων, την μέγιστη, ελάχιστη και μέση τιμή.

	variable	dtype	count	unique	missing value	Min	Max	Mean
0	id	int64	403	403	0	1000	41756	15978.310174
1	chol	float64	403	155	1	78.0	443.0	207.845771
2	stab.glu	int64	403	116	0	48	385	106.672457
3	hdl	float64	403	78	1	12.0	120.0	50.445274
4	ratio	float64	403	70	1	1.5	19.299999	4.521642
5	glyhb	float64	403	240	13	2.68	16.110001	5.589769
6	location	object	403	2	0	Str	Str	N/A
7	age	int64	403	68	0	19	92	46.851117
8	gender	object	403	2	0	Str	Str	N/A
9	height	float64	403	23	5	52.0	76.0	66.020101
10	weight	float64	403	141	1	99.0	325.0	177.59204
11	frame	object	403	4	12	Str	Str	N/A
12	bp.1s	float64	403	72	5	90.0	250.0	136.904523
13	bp.1d	float64	403	58	5	48.0	124.0	83.321608
14	bp.2s	float64	403	49	262	110.0	238.0	152.382979
15	bp.2d	float64	403	37	262	60.0	124.0	92.524823
16	waist	float64	403	31	2	26.0	56.0	37.900249
17	hip	float64	403	33	2	30.0	64.0	43.0399
18	time.ppn	float64	403	61	3	5.0	1560.0	341.25

Εικόνα 15: Πληροφορίες μεταβλητών

4.3 Προεπεξεργασία Δεδομένων



Εικόνα 16: Στάδια προεπεξεργασίας δεδομένων

4.3.1 Καθαρισμός Δεδομένων (Data Cleaning)

Για τον καθαρισμό των δεδομένων το πρώτο στάδιο είναι να μετατρέψουμε την μεταβλητή απόκρισης (glhyb) σε κατηγορική μεταβλητή με τιμές 0 ή 1 με το κριτήριο που αναφέραμε πριν. Στη συνέχεια θα ξεφορτωθούμε τις μεταβλητές που δεν προσφέρουν προβλεπτική ικανότητα όπως η id και η τοποθεσία. Στη συνέχεια θέλουμε να συμπληρώσουμε τις κενές τιμές με το διάμεσο στις ποιοτικές μεταβλητές και την πιο συχνή τιμή στις κατηγορικές. Στις κατηγορικές μεταβλητές θα μετατρέψουμε τα επίπεδα σε αριθμούς αντί για λέξεις για να είναι πιο εύκολη η μελέτη τους. Τέλος, θα χωρίσουμε τα δεδομένα σε δεδομένα εκπαίδευσης και δεδομένα δοκιμών που επισκοπούν στην αξιολόγηση των αλγορίθμων μηχανικής μάθησης. Ο κώδικας για την παραπάνω διαδικασία είναι ο εξής:

```python
df['outcome'] = (df['glyhb'] > 6).astype(int)
x = df.drop(['id', 'glyhb', 'outcome', 'location'], axis=1)
y = df['outcome']

numerical_cols = x.select_dtypes(include=['float64', 'int64']).columns
categorical_cols = x.select_dtypes(include=['object']).columns

imputer_num = SimpleImputer(strategy='median')
x[numerical_cols] = imputer_num.fit_transform(x[numerical_cols])

imputer_cat = SimpleImputer(strategy='most_frequent')
x[categorical_cols] = imputer_cat.fit_transform(x[categorical_cols])

# Encode categorical variables
label_encoder = LabelEncoder()
for col in categorical_cols:
    x[col] = label_encoder.fit_transform(x[col])

X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=42)
```

4.3.2 Τυποποίηση (Standardization)

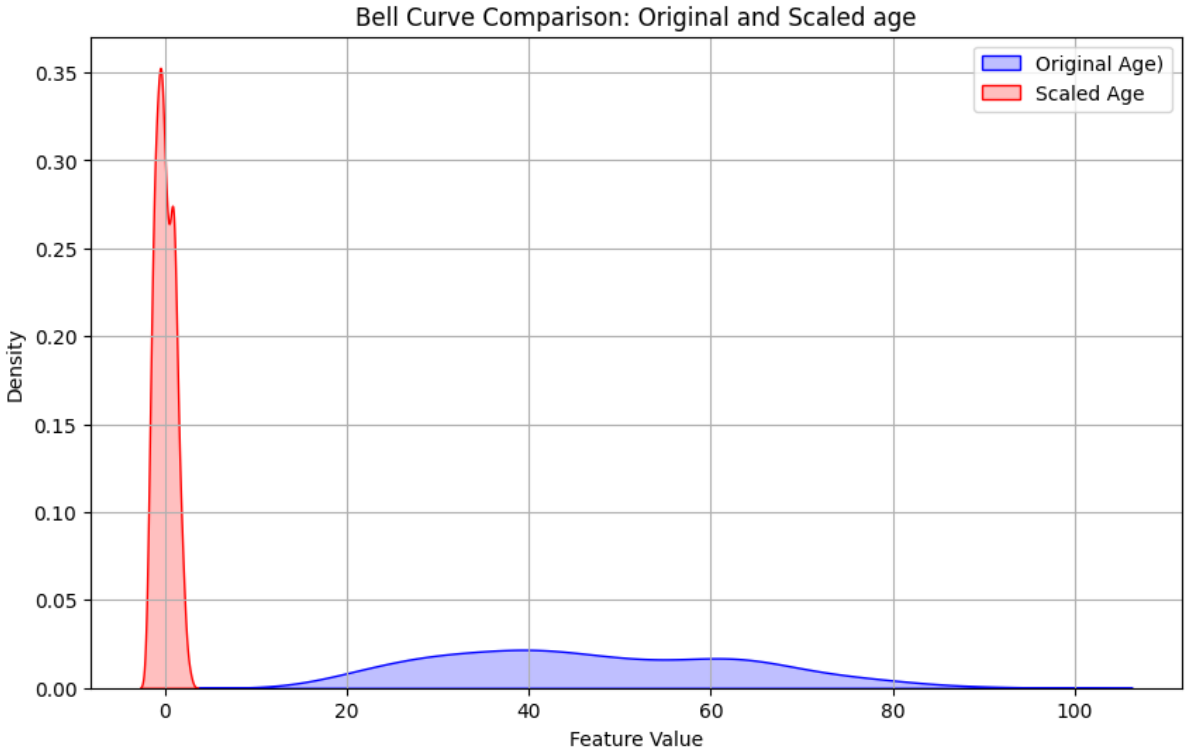
Η τυποποίηση (standardization) είναι μια διαδικασία κλιμάκωσης των δεδομένων που τα προσαρμόζει ώστε να έχουν μέση τιμή 0 και τυπική απόκλιση 1. Αυτή η μέθοδος είναι ιδιαίτερα χρήσιμη όταν δουλεύουμε με αλγορίθμους μηχανικής μάθησης που βασίζονται στις αποστάσεις μεταξύ των δεδομένων, όπως η λογιστική παλινδρόμηση, οι υποστηρικτικές διανυσματικές μηχανές (SVM) ή οι K-κοντινότεροι γείτονες (K-nearest neighbors).

Μερικά πλεονεκτήματα της τυποποίησης είναι:

- **Βελτίωση της απόδοσης του μοντέλου:** Τα δεδομένα με διαφορετικές κλίμακες μπορούν να προκαλέσουν προβλήματα στους αλγόριθμους, ειδικά όταν κάποια χαρακτηριστικά έχουν μεγάλες τιμές και άλλα πολύ μικρές. Η τυποποίηση διασφαλίζει ότι όλα τα χαρακτηριστικά έχουν την ίδια κλίμακα και δεν κυριαρχούν ορισμένα χαρακτηριστικά στη διαδικασία εκμάθησης.
- **Ταχύτερη σύγκλιση αλγορίθμων:** Σε αλγορίθμους βελτιστοποίησης, όπως η βαθμιαία καθοδική κλίση (gradient descent), η τυποποίηση επιταχύνει τη διαδικασία σύγκλισης, καθώς όλες οι μεταβλητές έχουν συγκρίσιμα μεγέθη.
- **Αποφυγή στρεβλώσεων λόγω κλίμακας:** Δίνει στα χαρακτηριστικά ίσο "βάρος" στη διαδικασία μάθησης, αποτρέποντας την υπερβολική επιρροή των χαρακτηριστικών με μεγάλες τιμές.

Η διαδικασία τυποποίησης μιας ποσοτικής μεταβλητής συμπεριλαμβάνει την αφαίρεση της μέσης τιμής από κάθε παρατήρηση και τη διαίρεση με τη διασπορά της μεταβλητής. Έτσι επιτυγχάνουμε οι ποσοτικές μεταβλητές να έχουν μέση τιμή 0 και τυπική απόκλιση 1. (Δεδομένου ότι ακολουθούν κανονική κατανομή).

$$x_{scaled} = \frac{x - \mu}{\sigma}$$



Εικόνα 17: Ηλικία πριν και μετά την τυποποίηση

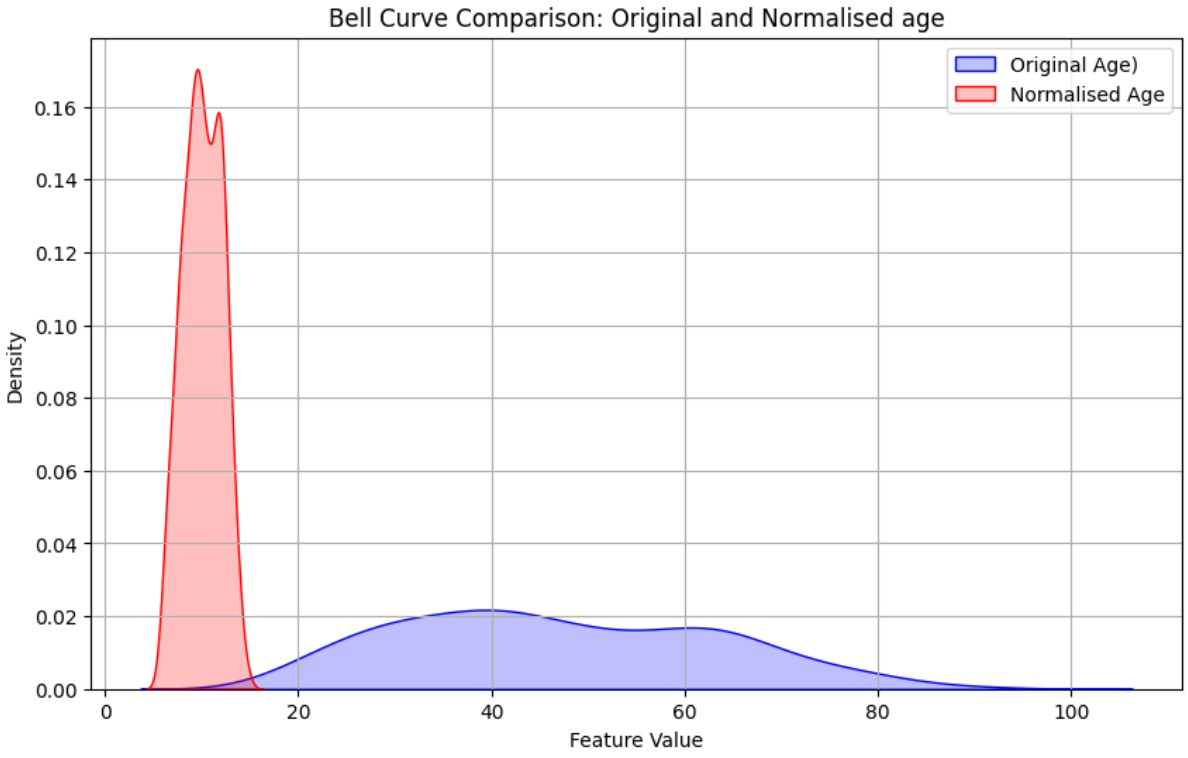
Στην παραπάνω εικόνα βλέπουμε τα αποτελέσματα της τυποποίησης της ηλικίας. Η μέση τιμή είναι 0 και η τυπική απόκλιση είναι 1, πολύ μικρότερη από την αρχική, έτσι, είναι πιο ψηλή και μαζεμένη η καμπύλη.

```
scaler = StandardScaler()
X_train_scaled = X_train.copy()
X_test_scaled = X_test.copy()

# Scale only the numerical columns in the training set and apply the same transformation to the test set
X_train_scaled[numerical_cols] = scaler.fit_transform(X_train[numerical_cols])
X_test_scaled[numerical_cols] = scaler.transform(X_test[numerical_cols])
```

4.3.3 Κανονικοποίηση (Box-Cox)

Σε περιπτώσεις όπου οι μεταβλητές δεν ακολουθούν την κανονική κατανομή ή τα δεδομένα είναι κατακερματισμένα μπορούμε να εφαρμόσουμε τον μετασχηματισμό Box-Cox για να λάβουμε μια καλύτερη κανονική κατανομή των δεδομένων. Στο συγκεκριμένο σετ δεδομένων δεν παρατηρήθηκαν περιπτώσεις που δεν ακολουθείται η κανονική κατανομή οπότε είναι λογικό να μην έχει σημαντικές αλλαγές στις αποδώσεις των μοντέλων ο συγκεκριμένος μετασχηματισμός, όμως σε άλλα είδη δεδομένων οι διαφορές θα ήταν πιο σημαντικές.

```
x_train_normal = pd.DataFrame()
x_test_normal = pd.DataFrame()

for col in X_train.columns:
    # Adding a small constant to avoid zero or negative values
    adjusted_train = X_train[col] + 1e-5  # Adjust as needed
    adjusted_test = X_test[col] + 1e-5  # Adjust as needed

    # Apply Box-Cox transformation
    x_train_normal[col], _ = stats.boxcox(adjusted_train)
    x_test_normal[col], _ = stats.boxcox(adjusted_test)
```



Εικόνα 18: Ηλικία πριν και μετά την κανονικοποίηση

4.3.4 Ανάλυση Κύριων Συνιστωσών (Principal Component Analysis)

Η Ανάλυση Κύριων Συνιστωσών (PCA) είναι μια τεχνική μείωσης διαστάσεων που μετασχηματίζει ένα σύνολο δεδομένων σε ένα νέο σύνολο μεταβλητών, που ονομάζονται κύριες συνιστώσες. Αυτές οι κύριες συνιστώσες στοχεύουν να συλλάβουν τα πιο σημαντικά μοτίβα ή τη διακύμανση στα δεδομένα, ενώ μειώνουν τον αριθμό των διαστάσεων. Η PCA κατασκευάζει τις κύριες συνιστώσες με τέτοιο τρόπο ώστε η πρώτη κύρια συνιστώσα (PC1) να εξηγεί το μεγαλύτερο ποσοστό της συνολικής διακύμανσης, η δεύτερη (PC2) να εξηγεί το επόμενο μεγαλύτερο ποσοστό, και ούτω καθεξής. Η συνολική πληροφορία για το πόσο "εξηγούν" οι κύριες συνιστώσες την αρχική διακύμανση ονομάζεται εξηγούμενη διακύμανση(explained variance).

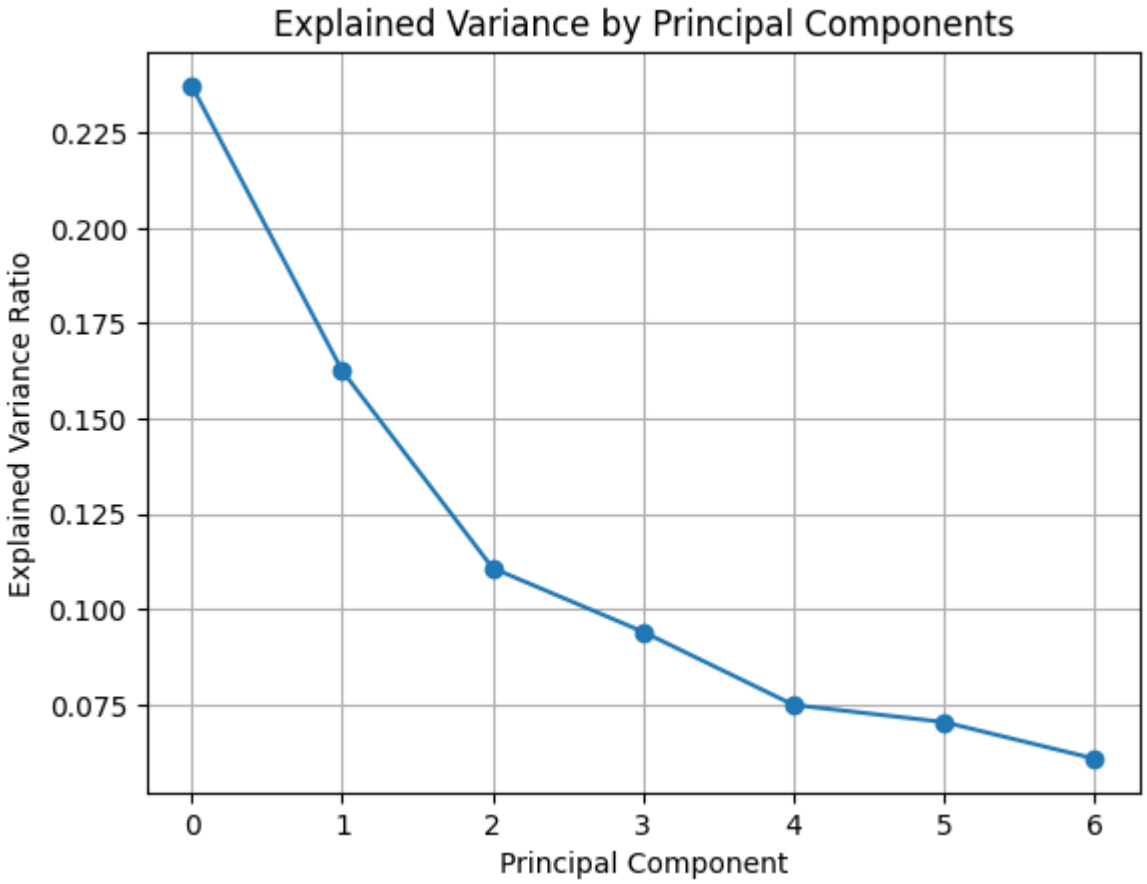
Το πλήθος των συνιστωσών που διαλέγουμε επηρεάζει την προβλεπτική ικανότητα του μοντέλου. Αν επιλέξουμε να μας επιστρέψει πολλές μεταβλητές τότε είναι πιθανό να μην υπάρχει σημαντική βελτίωση στην μείωση διαστάσεων. Από την άλλη, αν επιλέξουμε πολύ λίγες συνιστώσες μπορεί να χαθεί η προβλεπτική ικανότητα του μοντέλου. Είναι σημαντικό μέσω δοκιμών να βρεθεί ο κατάλληλος αριθμός συνιστωσών για να έχουμε τη μέγιστη εξηγούμενη διακύμανση με ταυτόχρονη σημαντική μείωση διαστάσεων.

Στα δικά μας δεδομένα επιλέξαμε να μειωθούν σε 7 συνιστώσες με συνολική εξηγούμενη διακύμανση 81%. Παρακάτω παρουσιάζεται ο κώδικας και η γραφική παράσταση της εξηγούμενης διακύμανσης κάθε συνιστώσας.

```
pca = PCA(n_components=7)  # For example, reduce to 7 components
x_test_pca = pca.fit_transform(X_test_scaled)
x_train_pca = pca.fit_transform(X_train_scaled)

# Explained variance can be used to understand the amount of information retained
print("Explained Variance: ", pca.explained_variance_ratio_)
print("Total Variance Explained: ", sum(pca.explained_variance_ratio_))
```

✓ 0.0s

```
Explained Variance:  [0.23724223 0.16252451 0.1107832  0.09406836 0.07488795 0.07039036
 0.06078779]
Total Variance Explained:  0.8106843946496078
```

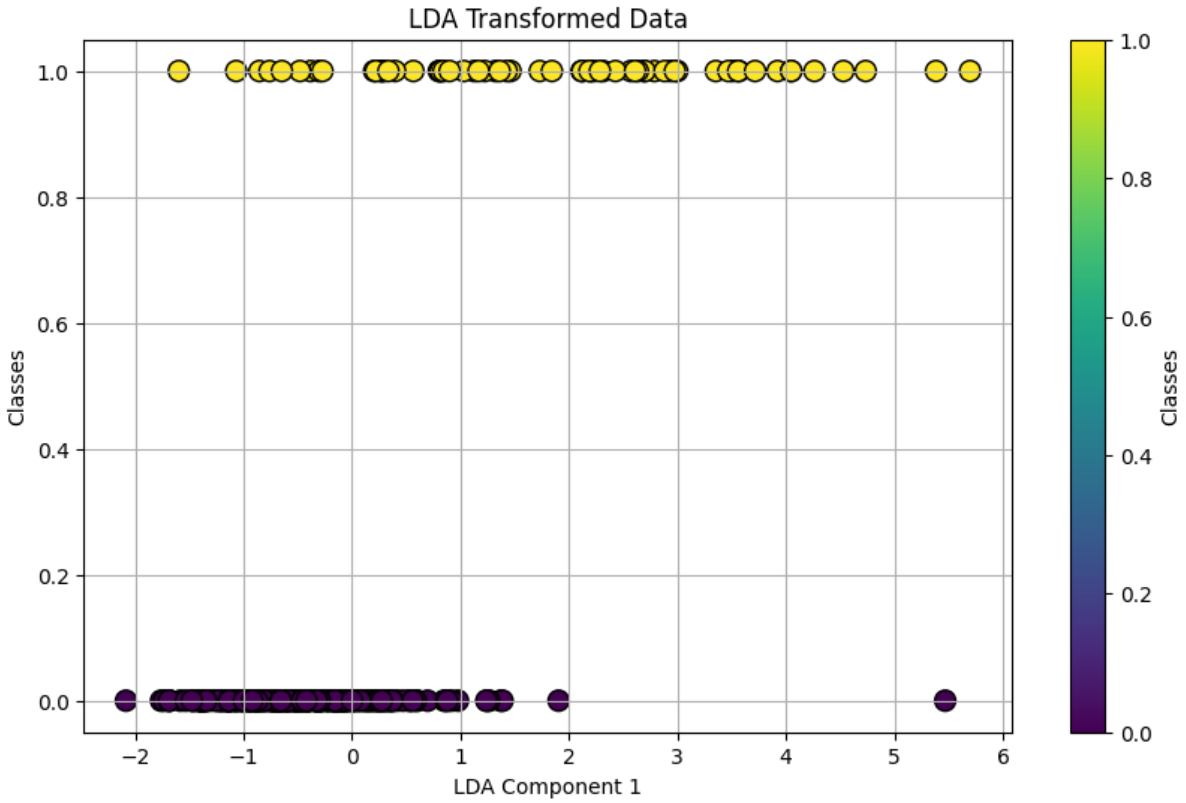


Εικόνα 19:Εξηγήσιμη διασπορά των κύριων συνιστωσών

4.3.5 Γραμμική Διακριτή Ανάλυση (Linear Discriminant Analysis)

Η Γραμμική Διακριτική Ανάλυση (LDA) είναι μια τεχνική μείωσης διαστάσεων που χρησιμοποιείται για την εύρεση των γραμμικών συνδυασμών χαρακτηριστικών που μεγιστοποιούν τον διαχωρισμό μεταξύ των διαφορετικών κατηγοριών σε ένα σύνολο δεδομένων. Σε αντίθεση με την Ανάλυση Κύριων Συνιστωσών (PCA), η οποία είναι μια μη επιβλεπόμενη τεχνική και στοχεύει στη μεγιστοποίηση της διασποράς των δεδομένων χωρίς να λαμβάνει υπόψη τις ετικέτες, η LDA είναι μια επιβλεπόμενη μέθοδος που χρησιμοποιεί τις ετικέτες των δεδομένων για να μεγιστοποιήσει τον διαχωρισμό των κατηγοριών.

```
lda_preprocessing = LinearDiscriminantAnalysis(n_components=1)

# Fit LDA on the training data and transform it
X_train_lda = lda_preprocessing.fit_transform(X_train, y_train)
X_test_lda = lda_preprocessing.transform(X_test)
```



Εικόνα 20:Μείωση σε μια διάσταση με LDA

4.4 Προσαρμογή Αλγορίθμων

Σε αυτή την ενότητα, θα παρουσιάσουμε την ανάπτυξη πολλαπλών αλγορίθμων μηχανικής μάθησης με στόχο την πρόβλεψη της πιθανότητας εμφάνισης διαβήτη. Η διαδικασία ανάπτυξης περιλαμβάνει την εκπαίδευση διαφόρων μοντέλων στο προεπεξεργασμένο σύνολο δεδομένων, την αξιολόγηση της απόδοσής τους, καθώς και την ανάδειξη βασικών πτυχών κάθε προσέγγισης. Οι αλγόριθμοι που εξετάζονται περιλαμβάνουν την Λογιστική Παλινδρόμηση, το SVM, τον Naïve Bayes, το Τυχαίο Δάσος (RF), τον XGBoost και τα Νευρωνικά Δίκτυα.

Για κάθε μοντέλο, θα εξηγήσουμε τη διαδικασία εκπαίδευσης, πρόβλεψης και αξιολόγησης, μαζί με τον αντίστοιχο κώδικα Python που επιδεικνύει τα βήματα που απαιτούνται για την εκπαίδευση του μοντέλου. Μέσω αυτών των μοντέλων, στοχεύουμε να

βρούμε την πιο αποτελεσματική προσέγγιση για την ακριβή πρόβλεψη του κινδύνου εμφάνισης διαβήτη, διατηρώντας παράλληλα την εξηγήσιμη και αποδοτική απόδοση.

❖ Λογιστική Παλινδρόμηση

```
# Initialize Logistic Regression
log_reg = LogisticRegression()

# Train and evaluate on raw data
log_reg.fit(X_train, y_train)
y_pred_lr = log_reg.predict(X_test)
accuracy_lr = accuracy_score(y_test, y_pred_lr)
precision_lr = precision_score(y_test, y_pred_lr, average='weighted')
recall_lr = recall_score(y_test, y_pred_lr, average='weighted')
f1_lr = f1_score(y_test, y_pred_lr, average='weighted')

# Train and evaluate on scaled data
log_reg.fit(X_train_scaled, y_train)
y_pred_lr_scaled = log_reg.predict(X_test_scaled)
accuracy_lr_scaled = accuracy_score(y_test, y_pred_lr_scaled)
precision_lr_scaled = precision_score(y_test, y_pred_lr_scaled, average='weighted')
recall_lr_scaled = recall_score(y_test, y_pred_lr_scaled, average='weighted')
f1_lr_scaled = f1_score(y_test, y_pred_lr_scaled, average='weighted')

# Train and evaluate on normalized data
log_reg.fit(x_train_normal, y_train)
y_pred_lr_normal = log_reg.predict(x_test_normal)
accuracy_lr_normal = accuracy_score(y_test, y_pred_lr_normal)
precision_lr_normal = precision_score(y_test, y_pred_lr_normal, average='weighted')
recall_lr_normal = recall_score(y_test, y_pred_lr_normal, average='weighted')
f1_lr_normal = f1_score(y_test, y_pred_lr_normal, average='weighted')

# Train and evaluate on PCA data
log_reg.fit(x_train_pca, y_train)
y_pred_lr_pca = log_reg.predict(x_test_pca)
accuracy_lr_pca = accuracy_score(y_test, y_pred_lr_pca)
precision_lr_pca = precision_score(y_test, y_pred_lr_pca, average='weighted')
recall_lr_pca = recall_score(y_test, y_pred_lr_pca, average='weighted')
f1_lr_pca = f1_score(y_test, y_pred_lr_pca, average='weighted')

# Train and evaluate on LDA data
log_reg.fit(X_train_lda, y_train)
y_pred_lr_lda = log_reg.predict(X_test_lda)
accuracy_lr_lda = accuracy_score(y_test, y_pred_lr_lda)
precision_lr_lda = precision_score(y_test, y_pred_lr_lda, average='weighted')
recall_lr_lda = recall_score(y_test, y_pred_lr_lda, average='weighted')
f1_lr_lda = f1_score(y_test, y_pred_lr_lda, average='weighted')
```

❖ **Support Vector Machine (SVM)**

```
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, precision_score, f1_score, recall_score

svm = SVC()

# Train and evaluate on raw data
svm.fit(X_train, y_train)
y_pred_svm = svm.predict(X_test)
accuracy_svm_raw = accuracy_score(y_test, y_pred_svm)
precision_svm_raw = precision_score(y_test, y_pred_svm, average='weighted')
recall_svm_raw = recall_score(y_test, y_pred_svm, average='weighted')
f1_svm_raw = f1_score(y_test, y_pred_svm, average='weighted')

# Train and evaluate on scaled data
svm.fit(X_train_scaled, y_train)
y_pred_svm_scaled = svm.predict(X_test_scaled)
accuracy_svm_scaled = accuracy_score(y_test, y_pred_svm_scaled)
precision_svm_scaled = precision_score(y_test, y_pred_svm_scaled, average='weighted')
recall_svm_scaled = recall_score(y_test, y_pred_svm_scaled, average='weighted')
f1_svm_scaled = f1_score(y_test, y_pred_svm_scaled, average='weighted')

# Train and evaluate on normalized data
svm.fit(x_train_normal, y_train)
y_pred_svm_normal = svm.predict(x_test_normal)
accuracy_svm_normal = accuracy_score(y_test, y_pred_svm_normal)
precision_svm_normal = precision_score(y_test, y_pred_svm_normal, average='weighted')
recall_svm_normal = recall_score(y_test, y_pred_svm_normal, average='weighted')
f1_svm_normal = f1_score(y_test, y_pred_svm_normal, average='weighted')

# Train and evaluate on PCA data
svm.fit(x_train_pca, y_train)
y_pred_svm_pca = svm.predict(x_test_pca)
accuracy_svm_pca = accuracy_score(y_test, y_pred_svm_pca)
precision_svm_pca = precision_score(y_test, y_pred_svm_pca, average='weighted')
recall_svm_pca = recall_score(y_test, y_pred_svm_pca, average='weighted')
f1_svm_pca = f1_score(y_test, y_pred_svm_pca, average='weighted')

# Train and evaluate on LDA data
svm.fit(X_train_lda, y_train)
y_pred_svm_lda = svm.predict(X_test_lda)
accuracy_svm_lda = accuracy_score(y_test, y_pred_svm_lda)
precision_svm_lda = precision_score(y_test, y_pred_svm_lda, average='weighted')
recall_svm_lda = recall_score(y_test, y_pred_svm_lda, average='weighted')
f1_svm_lda = f1_score(y_test, y_pred_svm_lda, average='weighted')
```

❖ **Naïve Bayes**

```
# Initialize Gaussian Naive Bayes
gnb = GaussianNB()

# Train and evaluate on raw data
gnb.fit(X_train, y_train)
y_pred_gnb = gnb.predict(X_test)
accuracy_gnb = accuracy_score(y_test, y_pred_gnb)
precision_gnb = precision_score(y_test, y_pred_gnb, average='weighted')
recall_gnb = recall_score(y_test, y_pred_gnb, average='weighted')
f1_gnb = f1_score(y_test, y_pred_gnb, average='weighted')

# Train and evaluate on scaled data
gnb.fit(X_train_scaled, y_train)
y_pred_gnb_scaled = gnb.predict(X_test_scaled)
accuracy_gnb_scaled = accuracy_score(y_test, y_pred_gnb_scaled)
precision_gnb_scaled = precision_score(y_test, y_pred_gnb_scaled, average='weighted')
recall_gnb_scaled = recall_score(y_test, y_pred_gnb_scaled, average='weighted')
f1_gnb_scaled = f1_score(y_test, y_pred_gnb_scaled, average='weighted')

# Train and evaluate on normalized data
gnb.fit(x_train_normal, y_train)
y_pred_gnb_normal = gnb.predict(x_test_normal)
accuracy_gnb_normal = accuracy_score(y_test, y_pred_gnb_normal)
precision_gnb_normal = precision_score(y_test, y_pred_gnb_normal, average='weighted')
recall_gnb_normal = recall_score(y_test, y_pred_gnb_normal, average='weighted')
f1_gnb_normal = f1_score(y_test, y_pred_gnb_normal, average='weighted')

# Train and evaluate on PCA data
gnb.fit(x_train_pca, y_train)
y_pred_gnb_pca = gnb.predict(x_test_pca)
accuracy_gnb_pca = accuracy_score(y_test, y_pred_gnb_pca)
precision_gnb_pca = precision_score(y_test, y_pred_gnb_pca, average='weighted')
recall_gnb_pca = recall_score(y_test, y_pred_gnb_pca, average='weighted')
f1_gnb_pca = f1_score(y_test, y_pred_gnb_pca, average='weighted')

# Train and evaluate on LDA data
gnb.fit(X_train_lda, y_train)
y_pred_gnb_lda = gnb.predict(X_test_lda)
accuracy_gnb_lda = accuracy_score(y_test, y_pred_gnb_lda)
precision_gnb_lda = precision_score(y_test, y_pred_gnb_lda, average='weighted')
recall_gnb_lda = recall_score(y_test, y_pred_gnb_lda, average='weighted')
f1_gnb_lda = f1_score(y_test, y_pred_gnb_lda, average='weighted')
```

❖ **Τυχαίο Δάσος (Random Forest)**

```
# Initialize RandomForestClassifier
clf = RandomForestClassifier(random_state=42)

# Train and evaluate on raw data
clf.fit(X_train, y_train)
y_pred_rf = clf.predict(X_test)
accuracy_rf_raw = accuracy_score(y_test, y_pred_rf)
precision_rf_raw = precision_score(y_test, y_pred_rf, average='weighted')
recall_rf_raw = recall_score(y_test, y_pred_rf, average='weighted')
f1_rf_raw = f1_score(y_test, y_pred_rf, average='weighted')

# Train and evaluate on scaled data
clf.fit(X_train_scaled, y_train)
y_pred_rf_scaled = clf.predict(X_test_scaled)
accuracy_rf_scaled = accuracy_score(y_test, y_pred_rf_scaled)
precision_rf_scaled = precision_score(y_test, y_pred_rf_scaled, average='weighted')
recall_rf_scaled = recall_score(y_test, y_pred_rf_scaled, average='weighted')
f1_rf_scaled = f1_score(y_test, y_pred_rf_scaled, average='weighted')

# Train and evaluate on normalized data
clf.fit(x_train_normal, y_train)
y_pred_rf_normal = clf.predict(x_test_normal)
accuracy_rf_normal = accuracy_score(y_test, y_pred_rf_normal)
precision_rf_normal = precision_score(y_test, y_pred_rf_normal, average='weighted')
recall_rf_normal = recall_score(y_test, y_pred_rf_normal, average='weighted')
f1_rf_normal = f1_score(y_test, y_pred_rf_normal, average='weighted')

# Train and evaluate on PCA data
clf.fit(x_train_pca, y_train)
y_pred_rf_pca = clf.predict(x_test_pca)
accuracy_rf_pca = accuracy_score(y_test, y_pred_rf_pca)
precision_rf_pca = precision_score(y_test, y_pred_rf_pca, average='weighted')
recall_rf_pca = recall_score(y_test, y_pred_rf_pca, average='weighted')
f1_rf_pca = f1_score(y_test, y_pred_rf_pca, average='weighted')

# Train and evaluate on LDA data
clf.fit(X_train_lda, y_train)
y_pred_rf_lda = clf.predict(X_test_lda)
accuracy_rf_lda = accuracy_score(y_test, y_pred_rf_lda)
precision_rf_lda = precision_score(y_test, y_pred_rf_lda, average='weighted')
recall_rf_lda = recall_score(y_test, y_pred_rf_lda, average='weighted')
f1_rf_lda = f1_score(y_test, y_pred_rf_lda, average='weighted')
```

❖ **XGBoost**

```
# Initialize XGBoost Classifier
xgb_clf = xgb.XGBClassifier()

# Train and evaluate on raw data
xgb_clf.fit(X_train, y_train)
y_pred_xgb = xgb_clf.predict(X_test)
accuracy_xgb = accuracy_score(y_test, y_pred_xgb)
precision_xgb = precision_score(y_test, y_pred_xgb, average='weighted')
recall_xgb = recall_score(y_test, y_pred_xgb, average='weighted')
f1_xgb = f1_score(y_test, y_pred_xgb, average='weighted')

# Train and evaluate on scaled data
xgb_clf.fit(X_train_scaled, y_train)
y_pred_xgb_scaled = xgb_clf.predict(X_test_scaled)
accuracy_xgb_scaled = accuracy_score(y_test, y_pred_xgb_scaled)
precision_xgb_scaled = precision_score(y_test, y_pred_xgb_scaled, average='weighted')
recall_xgb_scaled = recall_score(y_test, y_pred_xgb_scaled, average='weighted')
f1_xgb_scaled = f1_score(y_test, y_pred_xgb_scaled, average='weighted')

# Train and evaluate on normalized data
xgb_clf.fit(x_train_normal, y_train)
y_pred_xgb_normal = xgb_clf.predict(x_test_normal)
accuracy_xgb_normal = accuracy_score(y_test, y_pred_xgb_normal)
precision_xgb_normal = precision_score(y_test, y_pred_xgb_normal, average='weighted')
recall_xgb_normal = recall_score(y_test, y_pred_xgb_normal, average='weighted')
f1_xgb_normal = f1_score(y_test, y_pred_xgb_normal, average='weighted')

# Train and evaluate on PCA data
xgb_clf.fit(x_train_pca, y_train)
y_pred_xgb_pca = xgb_clf.predict(x_test_pca)
accuracy_xgb_pca = accuracy_score(y_test, y_pred_xgb_pca)
precision_xgb_pca = precision_score(y_test, y_pred_xgb_pca, average='weighted')
recall_xgb_pca = recall_score(y_test, y_pred_xgb_pca, average='weighted')
f1_xgb_pca = f1_score(y_test, y_pred_xgb_pca, average='weighted')

# Train and evaluate on LDA data
xgb_clf.fit(X_train_lda, y_train)
y_pred_xgb_lda = xgb_clf.predict(X_test_lda)
accuracy_xgb_lda = accuracy_score(y_test, y_pred_xgb_lda)
precision_xgb_lda = precision_score(y_test, y_pred_xgb_lda, average='weighted')
recall_xgb_lda = recall_score(y_test, y_pred_xgb_lda, average='weighted')
f1_xgb_lda = f1_score(y_test, y_pred_xgb_lda, average='weighted')
```

❖ **Νευρωνικά Δίκτυα**

```
from keras.models import Sequential
from keras.layers import Dense
from keras.optimizers import Adam
from sklearn.metrics import precision_score, f1_score
import numpy as np

# Define the models
model = Sequential()
model.add(Dense(12, input_dim=X_train.shape[1], activation='relu'))
model.add(Dense(8, activation='relu'))
model.add(Dense(1, activation='sigmoid'))

modelpca = Sequential()
modelpca.add(Dense(12, input_dim=x_train_pca.shape[1], activation='relu'))
modelpca.add(Dense(8, activation='relu'))
modelpca.add(Dense(1, activation='sigmoid'))

modellda = Sequential()
modellda.add(Dense(12, input_dim=X_train_lda.shape[1], activation='relu'))
modellda.add(Dense(8, activation='relu'))
modellda.add(Dense(1, activation='sigmoid'))

# Compile the models
model.compile(loss='binary_crossentropy', optimizer=Adam(), metrics=['accuracy'])
modelpca.compile(loss='binary_crossentropy', optimizer=Adam(), metrics=['accuracy'])
modellda.compile(loss='binary_crossentropy', optimizer=Adam(), metrics=['accuracy'])
```

Για τα Νευρωνικά Δίκτυα χρειάστηκε να δημιουργήσουμε 3 διαφορετικά μοντέλα λόγω των διαφορετικών διαστάσεων των δεδομένων. Τα δίκτυα περιέχουν μια κρυφή στρώση, η συνάρτηση ενεργοποίησης είναι η σιγμοειδής και η βελτιστοποίηση είναι με τη μέθοδο Adam. Στη συνέχεια τρέχω τα μοντέλα στα αντίστοιχα σετ δεδομένων.

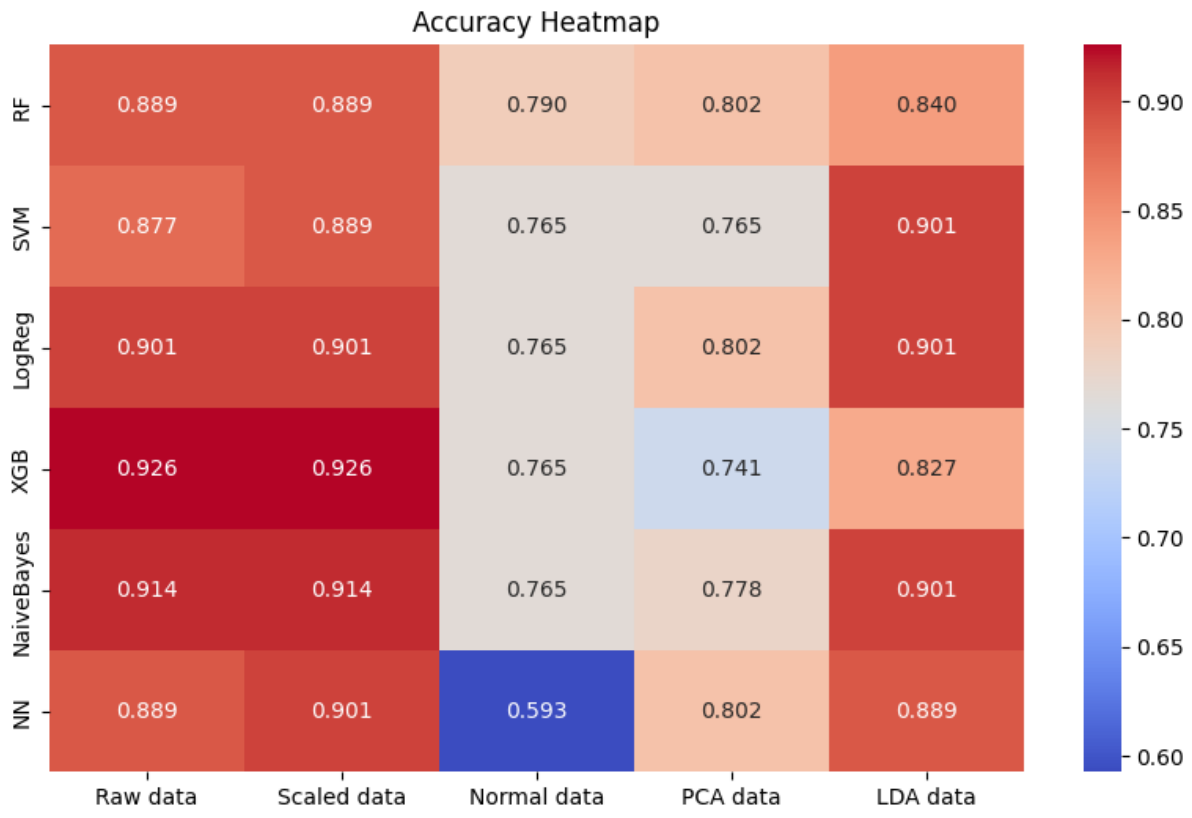
```
# Fit and evaluate the models
# Fit the model on raw data
model.fit(X_train, y_train, epochs=50, batch_size=10, verbose=0)
y_pred_nn_raw = (model.predict(X_test) > 0.5).astype(int)
accuracy_nn_raw = model.evaluate(X_test, y_test)
precision_nn_raw = precision_score(y_test, y_pred_nn_raw, average='weighted')
recall_nn_raw = recall_score(y_test, y_pred_nn_raw, average='weighted')
f1_nn_raw = f1_score(y_test, y_pred_nn_raw, average='weighted')

# Fit the model on scaled data
model.fit(X_train_scaled, y_train, epochs=50, batch_size=10, verbose=0)
y_pred_nn_scaled = (model.predict(X_test_scaled) > 0.5).astype(int)
accuracy_nn_scaled = model.evaluate(X_test_scaled, y_test)
precision_nn_scaled = precision_score(y_test, y_pred_nn_scaled, average='weighted')
recall_nn_scaled = recall_score(y_test, y_pred_nn_scaled, average='weighted')
f1_nn_scaled = f1_score(y_test, y_pred_nn_scaled, average='weighted')

# Fit the model on normalized data
model.fit(x_train_normal, y_train, epochs=50, batch_size=10, verbose=0)
y_pred_nn_normal = (model.predict(x_test_normal) > 0.5).astype(int)
accuracy_nn_normal = model.evaluate(x_test_normal, y_test)
precision_nn_normal = precision_score(y_test, y_pred_nn_normal, average='weighted')
recall_nn_normal = recall_score(y_test, y_pred_nn_normal, average='weighted')
f1_nn_normal = f1_score(y_test, y_pred_nn_normal, average='weighted')

# Fit the model on PCA data
modelpca.fit(x_train_pca, y_train, epochs=50, batch_size=10, verbose=0)
y_pred_nn_pca = (modelpca.predict(x_test_pca) > 0.5).astype(int)
accuracy_nn_pca = modelpca.evaluate(x_test_pca, y_test)
precision_nn_pca = precision_score(y_test, y_pred_nn_pca, average='weighted')
recall_nn_pca = recall_score(y_test, y_pred_nn_pca, average='weighted')
f1_nn_pca = f1_score(y_test, y_pred_nn_pca, average='weighted')

# Fit the model on LDA data
modellda.fit(X_train_lda, y_train, epochs=50, batch_size=10, verbose=0)
y_pred_nn_lda = (modellda.predict(X_test_lda) > 0.5).astype(int)
accuracy_nn_lda = modellda.evaluate(X_test_lda, y_test)
precision_nn_lda = precision_score(y_test, y_pred_nn_lda, average='weighted')
recall_nn_lda = recall_score(y_test, y_pred_nn_lda, average='weighted')
f1_nn_lda = f1_score(y_test, y_pred_nn_lda, average='weighted')
```

4.5 Αποτελέσματα

Αφού έχω φτιάξει όλα τα μοντέλα μηχανικής μάθησης και έχω αποθηκεύσει τις μετρικές απόδοσης για κάθε συνδυασμό προεπεξεργασίας και αλγορίθμου θα παραστήσω γραφικά τα αποτελέσματα για να έχω μια καλύτερη κατανόηση.

❖ **Ακρίβεια (Accuracy)**

```
accuracies = {
    "Raw data": [accuracy_rf_raw, accuracy_svm_raw, accuracy_lr, accuracy_xgb, accuracy_gnb, accuracy_nn_raw[1]] ,
    "Scaled data": [accuracy_rf_scaled, accuracy_svm_scaled, accuracy_lr_scaled, accuracy_xgb_scaled, accuracy_gnb_scaled, accuracy_nn_scaled[1]],
    "Normal data": [accuracy_rf_normal, accuracy_svm_normal, accuracy_lr_normal, accuracy_xgb_normal, accuracy_gnb_normal, accuracy_nn_normal[1]],
    "PCA data": [accuracy_rf_pca, accuracy_svm_pca, accuracy_lr_pca, accuracy_xgb_pca, accuracy_gnb_pca, accuracy_nn_pca[1]],
    "LDA data": [accuracy_rf_lda, accuracy_svm_lda, accuracy_lr_lda, accuracy_xgb_lda, accuracy_gnb_lda, accuracy_nn_lda[1]]
}

models = ['RF', 'SVM', 'LogReg', 'XGB', 'NaiveBayes', 'NN']

# Creating a DataFrame
accuracies_df = pd.DataFrame(accuracies, index=models)

# Creating the heatmap
plt.figure(figsize=(10, 6))
sns.heatmap(accuracies_df, annot=True, cmap='coolwarm', fmt=".3f")
plt.title('Accuracy Heatmap')
```



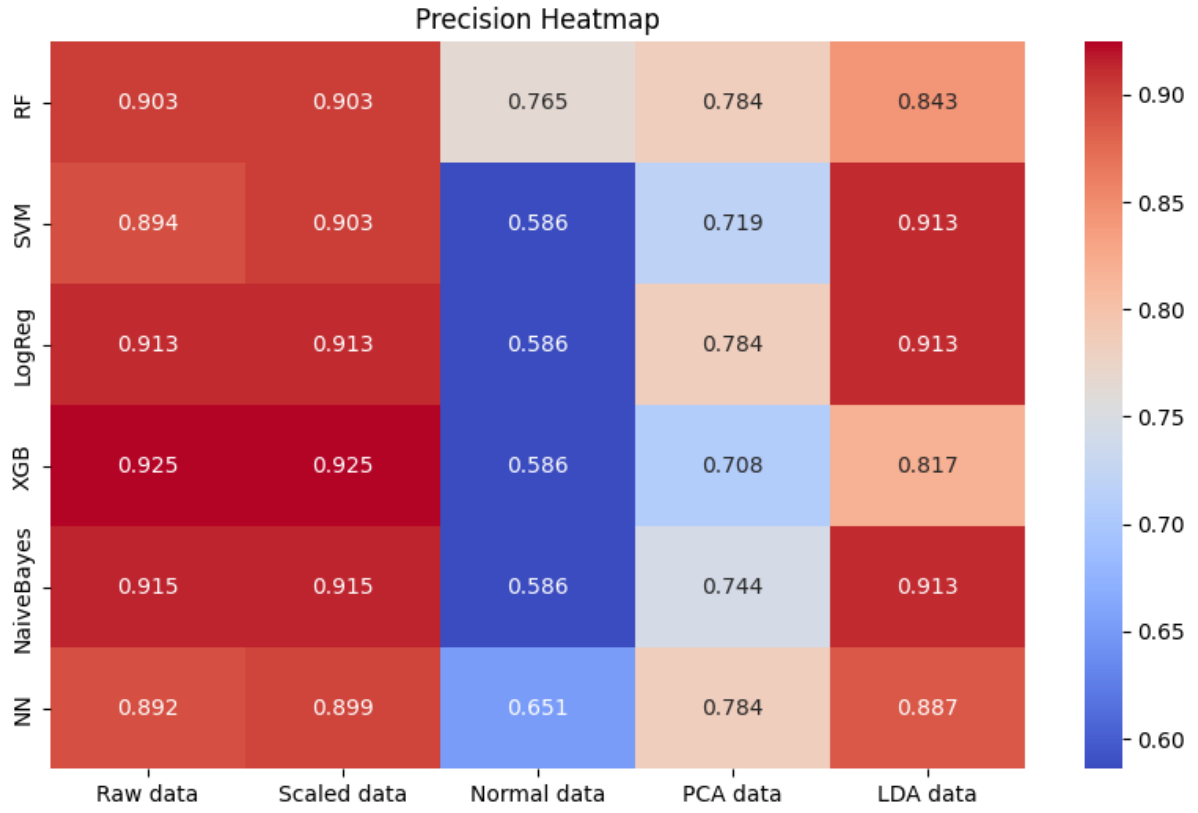
Εικόνα 21: Απεικόνιση της ακρίβειας των μοντέλων

Από την απεικόνιση των δεδομένων παρατηρούμε πως ο αλγόριθμος XGBoost έχει την καλύτερη ακρίβεια 92.6% στα αρχικά και τυποποιημένα δεδομένα. Αξίζει να

παρατηρήσουμε πως ενώ στο συγκεκριμένο αλγόριθμο δεν υπάρχει διαφορά στην προεπεξεργασία δεδομένων και μάλιστα κάποιες μέθοδοι ρίχνουν και την απόδοση, σε άλλους αλγορίθμους υπάρχει βελτίωση. Μια μέθοδος που σταθερά ρίχνει την ακρίβεια του μοντέλου είναι η κανονικοποίηση και είναι αναμενόμενο καθώς δεν είχαμε προβλήματα με την κανονικότητα των δεδομένων. Είναι σημαντικό να τονίσουμε πως δεν υπάρχει βέλτιστη λύση για κάθε περίπτωση και οφείλουμε να δοκιμάζουμε όλους τους συνδυασμούς ώστε να βρούμε τη βέλτιστη λύση για το σύνολο δεδομένων μας.

❖ Πιστότητα (Precision)

```
# Define precision scores
precisions = {
    "Raw data": [precision_rf_raw, precision_svm_raw, precision_lr, precision_xgb, precision_gnb, precision_nn_raw],
    "Scaled data": [precision_rf_scaled, precision_svm_scaled, precision_lr_scaled, precision_xgb_scaled, precision_gnb_scaled, precision_nn_scaled],
    "Normal data": [precision_rf_normal, precision_svm_normal, precision_lr_normal, precision_xgb_normal, precision_gnb_normal, precision_nn_normal],
    "PCA data": [precision_rf_pca, precision_svm_pca, precision_lr_pca, precision_xgb_pca, precision_gnb_pca, precision_nn_pca],
    "LDA data": [precision_rf_lda, precision_svm_lda, precision_lr_lda, precision_xgb_lda, precision_gnb_lda, precision_nn_lda]
}

# Creating a DataFrame for precision
precisions_df = pd.DataFrame(precisions, index=models)

# Creating the precision heatmap
plt.figure(figsize=(10, 6))
sns.heatmap(precisions_df, annot=True, cmap='coolwarm', fmt=".3f")
plt.title('Precision Heatmap')
plt.show()
```

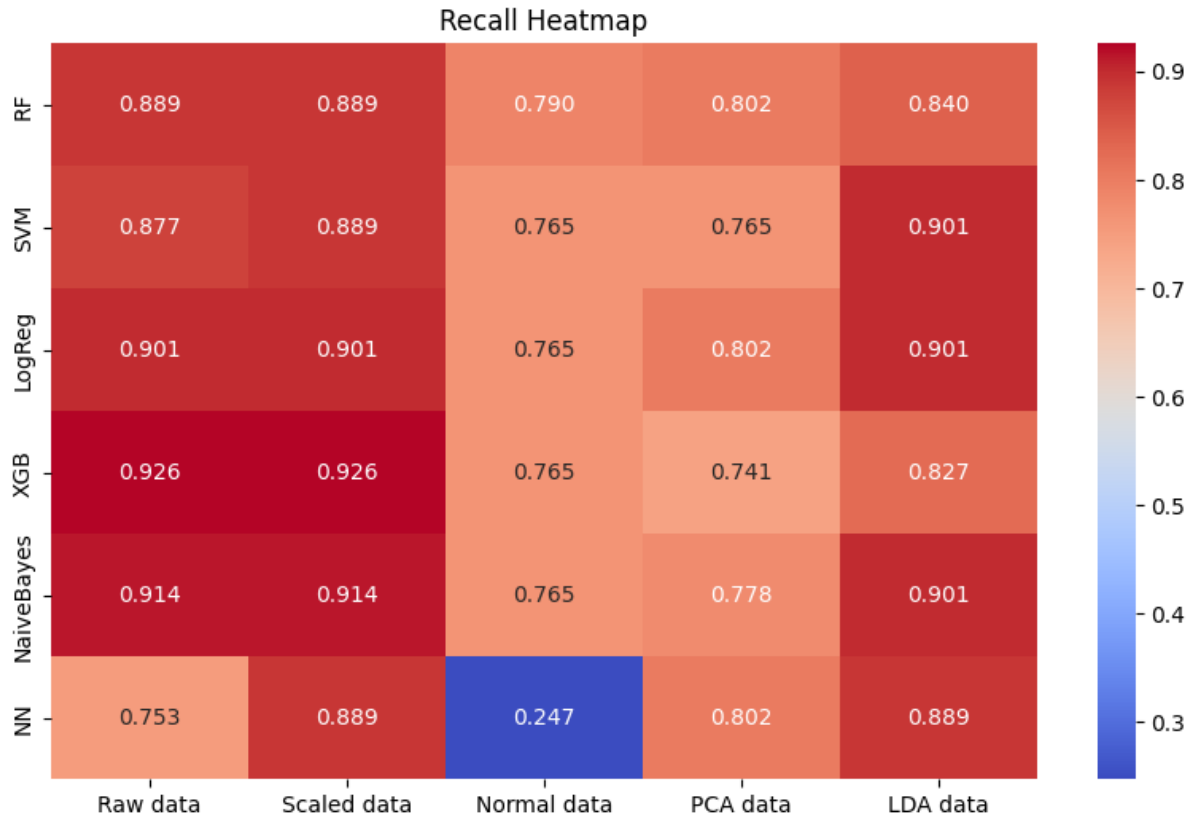


Εικόνα 22: Απεικόνιση της πιστότητας των μοντέλων

Παρατηρούμε παρόμοια αποτελέσματα με την ακρίβεια, πάλι βέλτιστο μοντέλο είναι ο XGBoost ενώ έχουμε πολύ υψηλή πιστότητα και από τους άλλους αλγορίθμους.

❖ **Ανάκληση (Recall)**

```
recalls = {
    "Raw data": [recall_rf_raw, recall_svm_raw, recall_lr, recall_xgb, recall_gnb, recall_nn_raw],
    "Scaled data": [recall_rf_scaled, recall_svm_scaled, recall_lr_scaled, recall_xgb_scaled, recall_gnb_scaled, recall_nn_scaled],
    "Normal data": [recall_rf_normal, recall_svm_normal, recall_lr_normal, recall_xgb_normal, recall_gnb_normal, recall_nn_normal],
    "PCA data": [recall_rf_pca, recall_svm_pca, recall_lr_pca, recall_xgb_pca, recall_gnb_pca, recall_nn_pca],
    "LDA data": [recall_rf_lda, recall_svm_lda, recall_lr_lda, recall_xgb_lda, recall_gnb_lda, recall_nn_lda]
}

# Creating a DataFrame for precision
precisions_df = pd.DataFrame(recalls, index=models)

# Creating the precision heatmap
plt.figure(figsize=(10, 6))
sns.heatmap(precisions_df, annot=True, cmap='coolwarm', fmt=".3f")
plt.title('Recall Heatmap')
plt.show()
```



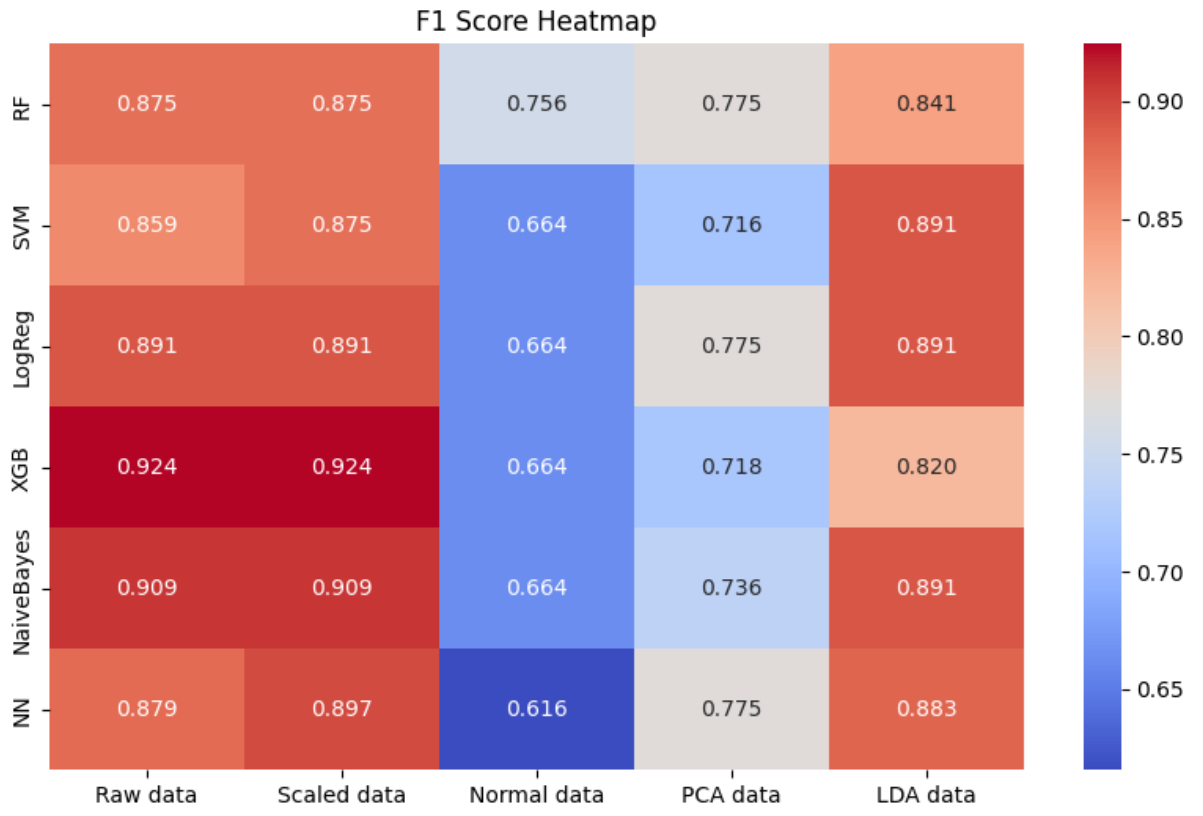
Εικόνα 23: Απεικόνιση της ανάκλησης των μοντέλων

Στην ανάκληση έχουμε πάλι τον ίδιο βέλτιστο συνδυασμό, ενώ παρατηρούμε τη χαμηλότερη τιμή του Νευρωνικού δικτύου με κανονικοποιημένα δεδομένα να είναι μόλις 24.7%. Είναι εντυπωσιακά χαμηλή τιμή και φαίνεται πως ο αλγόριθμος έχει πάρα πολλά ψευδή

αρνητικά. Δηλαδή πολύ συχνά ο αλγόριθμος προβλέπει ότι ο ασθενής δεν έχει διαβήτη ενώ στην πραγματικότητα έχει.

❖ F1 Score

```python
# Define F1 scores
f1_scores = {
    "Raw data": [f1_rf_raw, f1_svm_raw, f1_lr, f1_xgb, f1_gnb, f1_nn_raw],
    "Scaled data": [f1_rf_scaled, f1_svm_scaled, f1_lr_scaled, f1_xgb_scaled, f1_gnb_scaled, f1_nn_scaled],
    "Normal data": [f1_rf_normal, f1_svm_normal, f1_lr_normal, f1_xgb_normal, f1_gnb_normal, f1_nn_normal],
    "PCA data": [f1_rf_pca, f1_svm_pca, f1_lr_pca, f1_xgb_pca, f1_gnb_pca, f1_nn_pca],
    "LDA data": [f1_rf_lda, f1_svm_lda, f1_lr_lda, f1_xgb_lda, f1_gnb_lda, f1_nn_lda]
}

# Creating a DataFrame for F1 scores
f1_scores_df = pd.DataFrame(f1_scores, index=models)

# Creating the F1 score heatmap
plt.figure(figsize=(10, 6))
sns.heatmap(f1_scores_df, annot=True, cmap='coolwarm', fmt=".3f")
plt.title('F1 Score Heatmap')
plt.show()
```



Εικόνα 24: Απεικόνιση του δείκτη F1 των μοντέλων

Το F1 είναι μια συνάρτηση της πιστότητας και της ανάκλησης οπότε δεν περιμένουμε διαφορετικά αποτελέσματα μιας και οι δύο προηγούμενες μετρικές συμφωνούσαν. Από τις τέσσερεις διαφορετικές μετρικές έχουμε κοινή απόφαση πως ο αλγόριθμος XGBoost στα

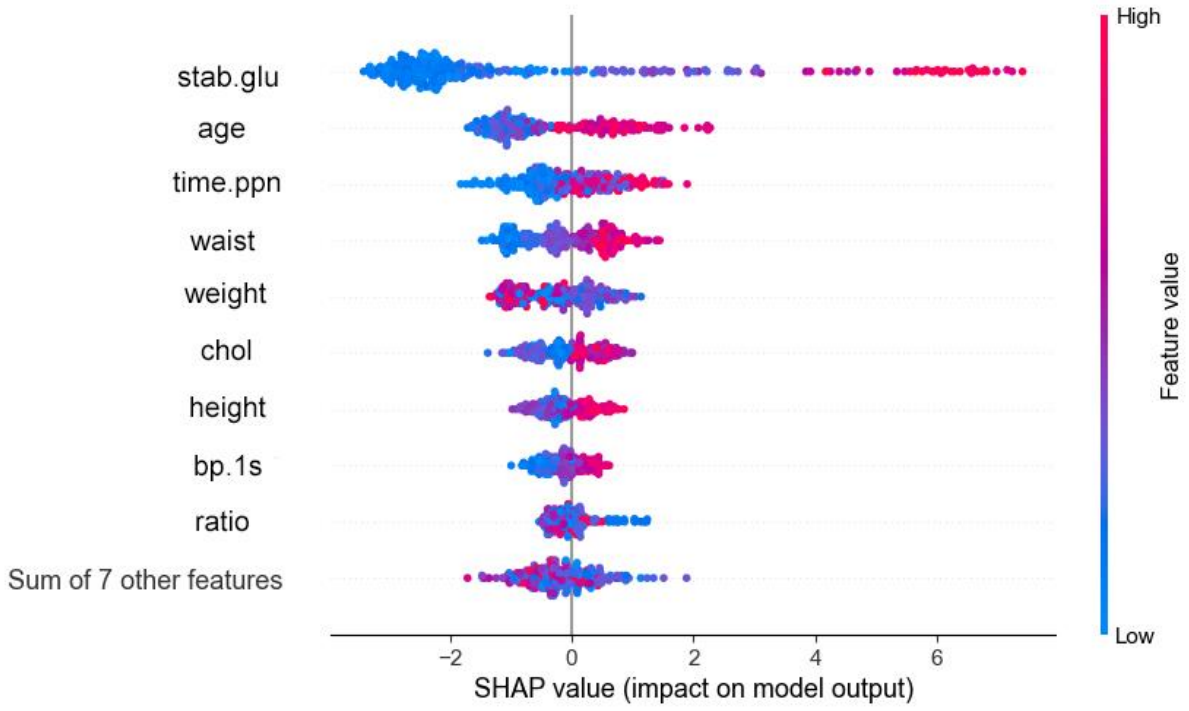
τυποποιημένα δεδομένα μας παρέχει με τα πιο αξιόπιστα αποτελέσματα και έχει την καλύτερη προβλεπτική ικανότητα.

4.6 Ερμηνεία Τελικού Μοντέλου

Μετά την ανάλυση των διαφόρων αλγορίθμων, βρήκαμε ότι το XGBoost στο τυποποιημένο (scaled) σύνολο δεδομένων ήταν ο καλύτερος αλγόριθμος για την πρόβλεψη της πιθανότητας εμφάνισης διαβήτη, επιδεικνύοντας την υψηλότερη ακρίβεια. Ωστόσο, για να κατανοήσουμε καλύτερα το μοντέλο και τη συμπεριφορά του, θέλουμε να εξετάσουμε ποια χαρακτηριστικά του συνόλου δεδομένων παίζουν τον πιο σημαντικό ρόλο στην πρόβλεψη του διαβήτη.

Για να επιτύχουμε αυτήν την ανάλυση, θα χρησιμοποιήσουμε τη βιβλιοθήκη SHAP (SHapley Additive exPlanations), η οποία μας επιτρέπει να εξηγήσουμε τις προβλέψεις των αλγορίθμων μηχανικής μάθησης, δίνοντάς μας εικόνα για τη συμβολή κάθε χαρακτηριστικού στην τελική πρόβλεψη. Ο SHAP παρέχει αναλύσεις εξήγησης χαρακτηριστικών βασισμένες στη θεωρία παιγνίων, δείχνοντάς μας την ακριβή επίδραση κάθε χαρακτηριστικού στο αποτέλεσμα του μοντέλου. Στην περίπτωσή μας, θα χρησιμοποιήσουμε το shap.TreeExplainer από τη βιβλιοθήκη SHAP, μιας και ο αλγόριθμος XGBoost έχει δομή δέντρου απόφασης, για να δούμε ποια χαρακτηριστικά συμβάλλουν περισσότερο στις προβλέψεις του XGBoost.

```
import shap
pred = xgb_clf_scaled.predict(X_train_scaled, output_margin=True)
# Create a TreeExplainer object
Xd = xgb.DMatrix(X_train_scaled,label=y_train)
Xdt = xgb.DMatrix(X_test_scaled,label=y_test)
explainer = shap.TreeExplainer(xgb_clf_scaled)
explanation = explainer(Xd)

shap_values = explanation.values
# make sure the SHAP values add up to marginal predictions
np.abs(shap_values.sum(axis=1) + explanation.base_values - pred).max()

shap.plots.beeswarm(explanation)
```



Εικόνα 25: Σημαντικότητα των μεταβλητών

Η γραφική απεικόνιση των SHAP τιμών που παρουσιάζεται, παρέχει μια ολοκληρωμένη ανάλυση του τρόπου με τον οποίο οι διάφορες μεταβλητές του συνόλου δεδομένων επηρεάζουν τις προβλέψεις που πραγματοποιούνται από το μοντέλο XGBoost για την πρόβλεψη του διαβήτη.

Στον **άξονα x**, οι SHAP τιμές αντιπροσωπεύουν την επίδραση κάθε χαρακτηριστικού στο αποτέλεσμα του μοντέλου. Θετικές SHAP τιμές (στην δεξιά πλευρά) δείχνουν ότι το χαρακτηριστικό αυξάνει την πρόβλεψη κινδύνου για διαβήτη, ενώ αρνητικές SHAP τιμές (στην αριστερή πλευρά) υποδεικνύουν ότι το χαρακτηριστικό μειώνει την πιθανότητα πρόβλεψης διαβήτη. Όσο πιο μακριά βρίσκεται μια τιμή από το μηδέν, τόσο μεγαλύτερη είναι η επίδραση του χαρακτηριστικού στην πρόβλεψη του μοντέλου. Σε αυτό το πλαίσιο, η κατανόηση των SHAP τιμών μας επιτρέπει να ερμηνεύσουμε τη συμπεριφορά του μοντέλου, ποσοτικοποιώντας την επιρροή κάθε χαρακτηριστικού στην τελική πρόβλεψη.

Στον **άξονα y**, οι μεταβλητές αναφέρονται με σειρά σημαντικότητας για το μοντέλο, με το πιο σημαντικό χαρακτηριστικό να εμφανίζεται στην κορυφή. Στην προκειμένη περίπτωση, η μεταβλητή stab.glu (σταθερή γλυκόζη) είναι ο πιο σημαντικός παράγοντας πρόβλεψης, παίζοντας καθοριστικό ρόλο στον υπολογισμό του κινδύνου για εμφάνιση διαβήτη. Όπως φαίνεται στο διάγραμμα, υψηλότερα επίπεδα γλυκόζης (κόκκινα σημεία) συνδέονται με υψηλότερη πιθανότητα διαβήτη, ενώ χαμηλότερα επίπεδα γλυκόζης (μπλε σημεία) μειώνουν την πιθανότητα εμφάνισης της νόσου. Αυτό είναι σύμφωνο με την

ιατρική γνώση, καθώς τα επίπεδα γλυκόζης αποτελούν γνωστό δείκτη κινδύνου για διαβήτη.

Άλλα σημαντικά χαρακτηριστικά περιλαμβάνουν την ηλικία, όπου οι μεγαλύτερης ηλικίας ασθενείς (κόκκινα σημεία) έχουν υψηλότερες SHAP τιμές, γεγονός που σημαίνει ότι η ηλικία συνδέεται θετικά με τον κίνδυνο διαβήτη. Αντίθετα, οι νεότεροι ασθενείς (μπλε σημεία) τείνουν να μειώνουν την πρόβλεψη κινδύνου. Οι μεταβλητές όπως η μέση και το βάρος συμβάλλουν επίσης σημαντικά, καθώς οι υψηλότερες τιμές αυτών των χαρακτηριστικών σχετίζονται με αυξημένο κίνδυνο. Αυτό είναι σύμφωνο με γνωστούς παράγοντες κινδύνου, όπου το αυξημένο βάρος και η περίμετρος μέσης συνδέονται με υψηλότερη συχνότητα εμφάνισης διαβήτη.

Το διάγραμμα δείχνει επίσης χαρακτηριστικά όπως η χοληστερόλη (chol) και η συστολική αρτηριακή πίεση (bp.1s), τα οποία έχουν πιο μετριοπαθή επίδραση σε σύγκριση με τη γλυκόζη ή την ηλικία, αλλά εξακολουθούν να συμβάλλουν στην πρόβλεψη συνολικά. Συγκεκριμένα, υψηλότερες τιμές χοληστερόλης και αρτηριακής πίεσης εμφανίζουν ελαφρά θετική επίδραση στην πρόβλεψη του κινδύνου για διαβήτη, αν και η συνολική τους συνεισφορά είναι λιγότερο έντονη σε σχέση με άλλα χαρακτηριστικά.

Στο κάτω μέρος του διαγράμματος παρατηρούμε μια σύνοψη λιγότερο σημαντικών χαρακτηριστικών κάτω από τον τίτλο Sum of 7 other features. Αυτές οι μεταβλητές εξακολουθούν να έχουν κάποιον ρόλο στις προβλέψεις του μοντέλου, αλλά η ατομική τους επίδραση είναι σχετικά μικρότερη σε σχέση με τα κυρίαρχα χαρακτηριστικά.

4.7 Dash

Σε αυτό το στάδιο, θα αναπτύξουμε μια διαδικτυακή διεπαφή για την πρόβλεψη του διαβήτη χρησιμοποιώντας τη γλώσσα προγραμματισμού Julia, με το πλαίσιο Dash για τη δημιουργία διαδραστικών στοιχείων. Αυτή η διεπαφή θα επιτρέπει στους χρήστες να εισάγουν παραμέτρους υγείας ασθενών, όπως ηλικία, επίπεδα χοληστερόλης, επίπεδα γλυκόζης και άλλες σχετικές μεταβλητές, και να λαμβάνουν σε πραγματικό χρόνο προβλέψεις για την πιθανότητα εμφάνισης διαβήτη.

Η διαδικασία ξεκινά με τη φόρτωση και την προετοιμασία των δεδομένων στη Julia, χρησιμοποιώντας το πακέτο CSV για την ανάγνωση των δεδομένων και το πακέτο Dataframes για τον χειρισμό τους. Το σύνολο δεδομένων εκπαίδευσης προεπεξεργάζεται και χρησιμοποιείται για την εκπαίδευση ενός μοντέλου XGBoost όπως διαλέξαμε πριν. Οι

υπερπαράμετροι του μοντέλου, όπως το max_depth, το eta και το eval_metric, επιλέγονται προσεκτικά για τη βελτιστοποίηση της απόδοσης στο δεδομένο σύνολο δεδομένων.

Μετά την εκπαίδευση του μοντέλου, ορίζεται μια συνάρτηση πρόβλεψης για την ταξινόμηση μελλοντικών εισόδων. Αυτή η συνάρτηση χρησιμοποιεί το εκπαιδευμένο μοντέλο XGBoost για να κάνει προβλέψεις με βάση τα δεδομένα υγείας που παρέχονται από τον χρήστη. Η συνάρτηση επιστρέφει την πιθανότητα εμφάνισης διαβήτη, μετατρέποντας την πιθανότητα σε δυαδική ταξινόμηση (διαβητικός ή μη) με βάση ένα προκαθορισμένο όριο.

```
using Pkg
using CSV
using DataFrames
using XGBoost
using Dash, DashCoreComponents, DashHtmlComponents

# Load data
X_train = CSV.read("X_train.csv", DataFrame)
y_train = CSV.read("y_train.csv", DataFrame)
X_test = CSV.read("X_test.csv", DataFrame)
y_test = CSV.read("y_test.csv", DataFrame)


# Convert y_train and y_test to vectors
y_train = convert(Vector{Float32}, y_train[:, 1])
y_test = convert(Vector{Float32}, y_test[:, 1])

# Convert DataFrames to matrices
X_train_matrix = Matrix(X_train)
X_test_matrix = Matrix(X_test)

# Training
dtrain = DMatrix(X_train_matrix, label=y_train)
param = Dict("max_depth" => 3, "eta" => 0.1, "objective" => "binary:logistic", "eval_metric" => "logloss")
num_round = 100
bst = xgboost(dtrain, num_round=num_round, params=param)


# Prediction function
function predict_diabetes(model, input_data)
    dtest = DMatrix(input_data)
    preds = XGBoost.predict(model, dtest)
    return preds[1] > 0.5  # Convert probabilities to binary output
end
```

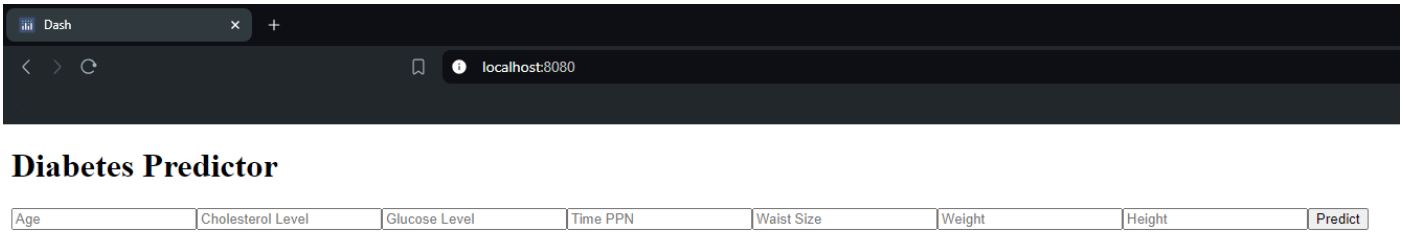
Το βασικό στοιχείο της διεπαφής είναι η εφαρμογή Dash. Αυτή η εφαρμογή περιλαμβάνει πεδία εισαγωγής για σχετικές ιατρικές μετρήσεις (π.χ., ηλικία, επίπεδο χοληστερόλης, επίπεδο γλυκόζης κ.λπ.) και ένα κουμπί για την έναρξη της διαδικασίας πρόβλεψης. Μετά την επιλογή του κουμπιού "Πρόβλεψη", τα δεδομένα εισαγωγής επεξεργάζονται και περνούν στο εκπαιδευμένο μοντέλο XGBoost, το οποίο επιστρέφει την

πρόβλεψη. Το προβλεπόμενο αποτέλεσμα εμφανίζεται σε πραγματικό χρόνο μέσα στην εφαρμογή.

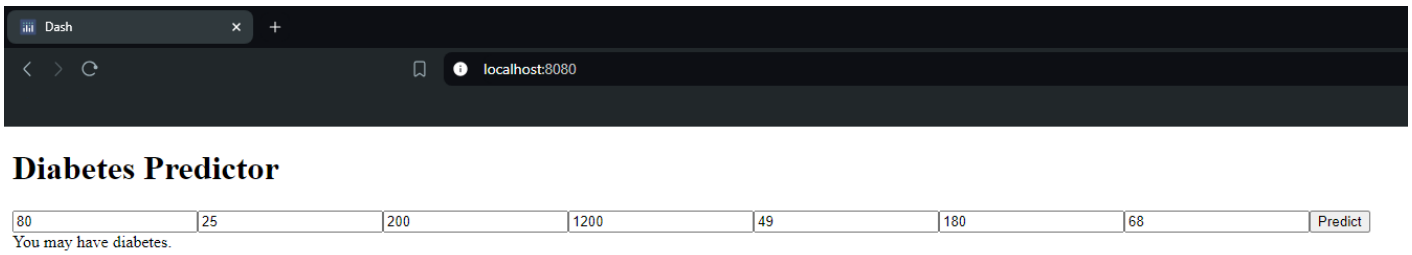
```
app = dash()

app.layout = html_div() do
    html_h1("Diabetes Predictor"),
    dcc_input(id="input-age", type="number", placeholder="Age"),
    dcc_input(id="input-chol", type="number", placeholder="Cholesterol Level"),
    dcc_input(id="input-glucose", type="number", placeholder="Glucose Level"),
    dcc_input(id="input-time_ppn", type="number", placeholder="Time PPN"),
    dcc_input(id="input-waist", type="number", placeholder="Waist Size"),
    dcc_input(id="input-weight", type="number", placeholder="Weight"),
    dcc_input(id="input-height", type="number", placeholder="Height"),
    html_button("Predict", id="predict-button", n_clicks=0),
    html_div(id="output-prediction")
end

callback!(app, Output("output-prediction", "children"),
          Input("predict-button", "n_clicks"),
          State("input-age", "value"),
          State("input-chol", "value"),
          State("input-glucose", "value"),
          State("input-time_ppn", "value"),
          State("input-waist", "value"),
          State("input-weight", "value"),
          State("input-height", "value")) do n_clicks, age, chol, glucose, time_ppn, waist, weight, height
    if !ismissing(age) && !ismissing(chol) && !ismissing(glucose) && !ismissing(time_ppn) &&
       !ismissing(waist) && !ismissing(weight) && !ismissing(height)
       input_data = [age, chol, glucose, time_ppn, waist, weight, height]
       input_data = reshape(input_data, 1, length(input_data))  # Reshape to 2D matrix
       prediction = predict_diabetes(bst, input_data)
       return prediction ? "You may have diabetes." : "You are unlikely to have diabetes."
    else
       return "Please enter all the values."
    end
end


run_server(app, "0.0.0.0", 8080)
```

Η εφαρμογή έχει σχεδιαστεί με γνώμονα την αλληλεπίδραση των χρηστών, καθιστώντας την προσβάσιμη και λειτουργική για μη τεχνικούς χρήστες, όπως επαγγελματίες υγείας. Με τη φιλοξενία της εφαρμογής τοπικά, διευκολύνεται η δοκιμή και η επανάληψη. Τα μελλοντικά σχέδια περιλαμβάνουν την επέκταση αυτής της ανάπτυξης σε μια πλατφόρμα ψηφιακού νέφους(cloud) για ευρύτερη πρόσβαση.

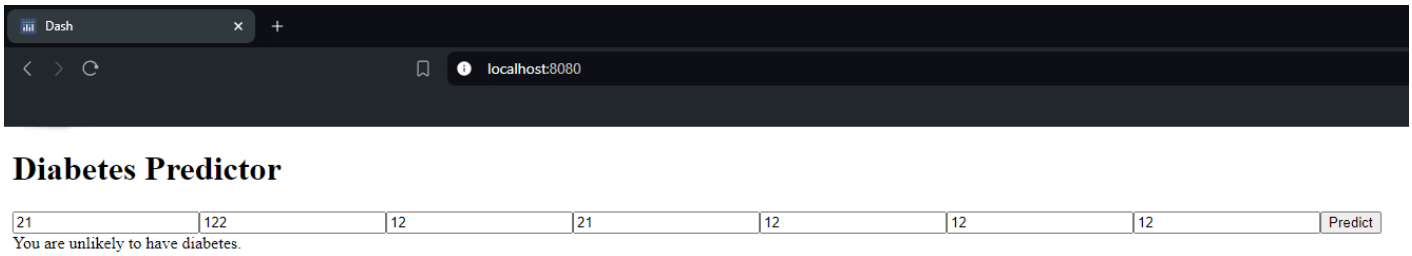
Παρακάτω παρουσιάζονται τρία παραδείγματα: το πρώτο αφορά την κατάσταση πριν από την εισαγωγή των στοιχείων, ενώ τα άλλα δύο αφορούν την πρόβλεψη για έναν διαβητικό και έναν μη διαβητικό ασθενή.



Εικόνα 26: Εμφάνιση της εφαρμογής Dash



Εικόνα 28: Θετική πρόβλεψη από την εφαρμογή



Εικόνα 27: Αρνητική πρόβλεψη από την εφαρμογή

5. ΠΡΟΒΛΗΜΑΤΑ ΤΗΣ ΜΗΧΑΝΙΚΗΣ ΜΑΘΗΣΗΣ ΣΤΟΝ ΤΟΜΕΑ ΤΗΣ ΥΓΕΙΑΣ

Η μηχανική μάθηση (ΜΜ) έχει αναδειχθεί ως ένα ισχυρό εργαλείο στον τομέα της υγείας, προσφέροντας τη δυνατότητα να φέρει επανάσταση στη διάγνωση, την πρόβλεψη ασθενειών και την εξατομικευμένη θεραπεία. Με την ικανότητά της να αναλύει μεγάλα ποσά ιατρικών δεδομένων, να εντοπίζει πολύπλοκα πρότυπα και να παρέχει σε πραγματικό χρόνο πληροφορίες, η Μηχανική Μάθηση μεταμορφώνει ήδη τον τομέα της υγειονομικής περίθαλψης, καθιστώντας τον πιο αποδοτικό και ακριβή. Από την ανίχνευση πρώιμων ενδείξεων ασθενειών μέχρι τη βελτιστοποίηση των θεραπευτικών σχεδίων, οι εφαρμογές της μηχανικής μάθησης είναι ευρείες και πολλά υποσχόμενες.

Παρά όμως τις προφανείς δυνατότητές της, η εφαρμογή της μηχανικής μάθησης στην υγεία συνοδεύεται από μια σειρά προκλήσεων που περιορίζουν την αποτελεσματικότητα και την ηθική χρήση αυτών των τεχνολογιών. Καθώς τα συστήματα υγείας υιοθετούν λύσεις βασισμένες στη ML, καλούνται να αντιμετωπίσουν διάφορα κρίσιμα ζητήματα. Αυτά τα ζητήματα μπορούν να ομαδοποιηθούν σε τρεις βασικές κατηγορίες:

- Δεδομένα,
- Ηθικά Ζητήματα
- Οικονομικά κόστη

Εικόνα 29: Προβλήματα της Τεχνητής Νοημοσύνης στον υγειονομικό τομέα:

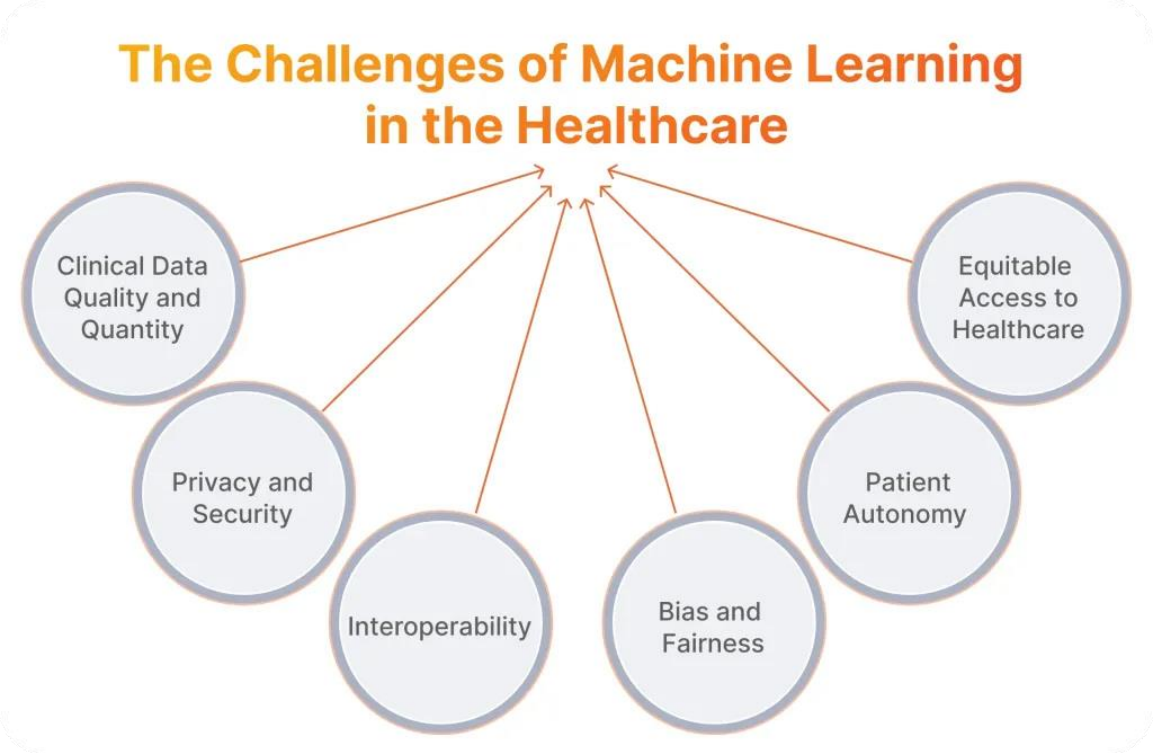
5.1 Δεδομένα

Μία από τις κυριότερες προκλήσεις είναι η ποιότητα και η διαθεσιμότητα των δεδομένων υγείας. Τα ιατρικά δεδομένα είναι συχνά ελλιπή, θορυβώδη ή ασυνεπή, με ελλείπουσες τιμές και σφάλματα στα αρχεία ασθενών, στα διαγνωστικά αποτελέσματα ή στο ιστορικό θεραπειών. Η απόδοση των μοντέλων ML επηρεάζεται άμεσα από την ποιότητα των δεδομένων και η εκπαίδευση μοντέλων σε δεδομένα χαμηλής ποιότητας μπορεί να οδηγήσει σε ανακριβείς προβλέψεις. Επιπλέον, η ιατρική συχνά στερείται μεγάλων, επισημασμένων συνόλων δεδομένων, που είναι κρίσιμα για την εκπαίδευση πολλών μοντέλων επιβλεπόμενης μάθησης. Η χειροκίνητη επισημείωση δεδομένων, ειδικά από ιατρικούς εμπειρογνώμονες, είναι δαπανηρή και χρονοβόρα. Επιπρόσθετα, τα δεδομένα υγείας είναι συχνά μη ισορροπημένα, όπου ασθένειες ή καταστάσεις που επηρεάζουν ένα μικρό ποσοστό του πληθυσμού μπορεί να οδηγήσουν σε προκατειλημμένα μοντέλα που αποδίδουν κακά για την μειοψηφία.

Ένα άλλο μεγάλο πρόβλημα είναι η ερμηνευσιμότητα. Πολλά από τα πιο προηγμένα μοντέλα μηχανικής μάθησης, ιδιαίτερα τα βαθιά νευρωνικά δίκτυα, θεωρούνται συχνά ως "μαύρα κουτιά". Αυτά τα μοντέλα κάνουν προβλέψεις που είναι δύσκολο να εξηγηθούν με τρόπο που να ευθυγραμμίζεται με την κλινική κατανόηση. Στον τομέα της υγείας, αυτή η έλλειψη ερμηνευσιμότητας είναι σοβαρό ζήτημα, καθώς οι γιατροί και οι επαγγελματίες υγείας πρέπει να κατανοήσουν γιατί ένα μοντέλο κάνει ορισμένες προβλέψεις ή προτάσεις, ειδικά όταν πρόκειται για αποφάσεις ζωής ή θανάτου. Οι κλινικοί ιατροί πρέπει να είναι σε θέση να εμπιστεύονται τις προβλέψεις του μοντέλου και να εξηγούν τη λογική του στους ασθενείς, αλλά τα μοντέλα που παράγουν εξαιρετικά ακριβή αποτελέσματα χωρίς διαφανή λογική μπορεί να προκαλέσουν δυσπιστία. Αυτή η πρόκληση επιδεινώνεται από το γεγονός ότι τα μοντέλα ML μπορεί να εντοπίσουν μη διαισθητικά χαρακτηριστικά (όπως ο ταχυδρομικός κώδικας ενός ασθενούς ή κοινωνικοοικονομικοί παράγοντες) που οδηγούν στις προβλέψεις, αλλά αυτά τα χαρακτηριστικά μπορεί να μην έχουν κλινικό νόημα.

Η γενίκευση των μοντέλων ML στην υγεία είναι ένα άλλο σημαντικό εμπόδιο. Τα μοντέλα που εκπαιδεύονται με δεδομένα από ένα συγκεκριμένο νοσοκομείο ή περιοχή μπορεί να μην γενικεύουν καλά όταν εφαρμόζονται σε διαφορετικούς πληθυσμούς ή περιβάλλοντα. Οι διαφορές στη δημογραφία των ασθενών, στην επικράτηση ασθενειών και στα πρωτόκολλα θεραπείας σε διαφορετικές περιοχές μπορούν να επηρεάσουν την απόδοση του μοντέλου. Επιπλέον, τα μοντέλα μηχανικής μάθησης, ιδιαίτερα εκείνα που είναι εξαιρετικά περίπλοκα, μπορεί εύκολα να υπερπροσαρμοστούν στα δεδομένα με τα οποία έχουν εκπαιδευτεί, γεγονός

που τα καθιστά λιγότερο αποτελεσματικά όταν αναπτυχθούν σε πραγματικά κλινικά περιβάλλοντα. Αυτό το πρόβλημα περιπλέκεται περαιτέρω από το γεγονός ότι τα δεδομένα υγείας είναι συχνά περιορισμένα σε μέγεθος και εύρος, γεγονός που καθιστά δύσκολη την κατασκευή ανθεκτικών, γενικεύσιμων μοντέλων.



Εικόνα 30:Άλλα προβλήματα

5.2 Ηθικά Ζητήματα

Οι ηθικές και νομικές ανησυχίες θέτουν επίσης σοβαρά εμπόδια στην εφαρμογή της μηχανικής μάθησης στην υγεία. Η προκατάληψη στα μοντέλα ML είναι ένα κρίσιμο ζήτημα, καθώς τα μοντέλα είναι τόσο καλά όσο τα δεδομένα με τα οποία εκπαιδεύονται. Αν ένα μοντέλο εκπαιδευτεί σε προκατειλημμένα ή μη αντιπροσωπευτικά δεδομένα, μπορεί να παράγει προκατειλημμένα αποτελέσματα, ιδιαίτερα για μειονοτικές ομάδες. Αυτό μπορεί να οδηγήσει σε άδικες προτάσεις θεραπείας ή ανακριβείς διαγνώσεις, κάτι που έχει σοβαρές ηθικές επιπτώσεις.

Επιπλέον, υπάρχουν άλυτα ερωτήματα σχετικά με την ευθύνη: εάν ένα μοντέλο μηχανικής μάθησης κάνει μια λανθασμένη διάγνωση ή προτείνει μια λανθασμένη θεραπεία, δεν είναι σαφές ποιος θα θεωρηθεί υπεύθυνος – οι προγραμματιστές του μοντέλου, οι κλινικοί ιατροί που το χρησιμοποιούν ή το ίδιο το ίδρυμα υγείας. Αυτή η νομική ασάφεια δημιουργεί εμπόδια στην ευρεία υιοθέτηση της Μηχανικής Μάθησης σε κλινικά περιβάλλοντα. Επιπλέον, το ζήτημα της ιδιωτικότητας των δεδομένων των ασθενών περιπλέκει τη χρήση της ΜΜ στην υγεία. Οι αυστηροί κανονισμοί όπως το HIPAA στις ΗΠΑ και το GDPR στην Ευρώπη περιορίζουν τη διαθεσιμότητα των δεδομένων ασθενών για την εκπαίδευση μοντέλων μηχανικής μάθησης, δημιουργώντας προκλήσεις για την ανταλλαγή δεδομένων μεταξύ των ιδρυμάτων και την εκτέλεση έρευνας μεγάλης κλίμακας.

5.3 Οικονομικά Κόστη

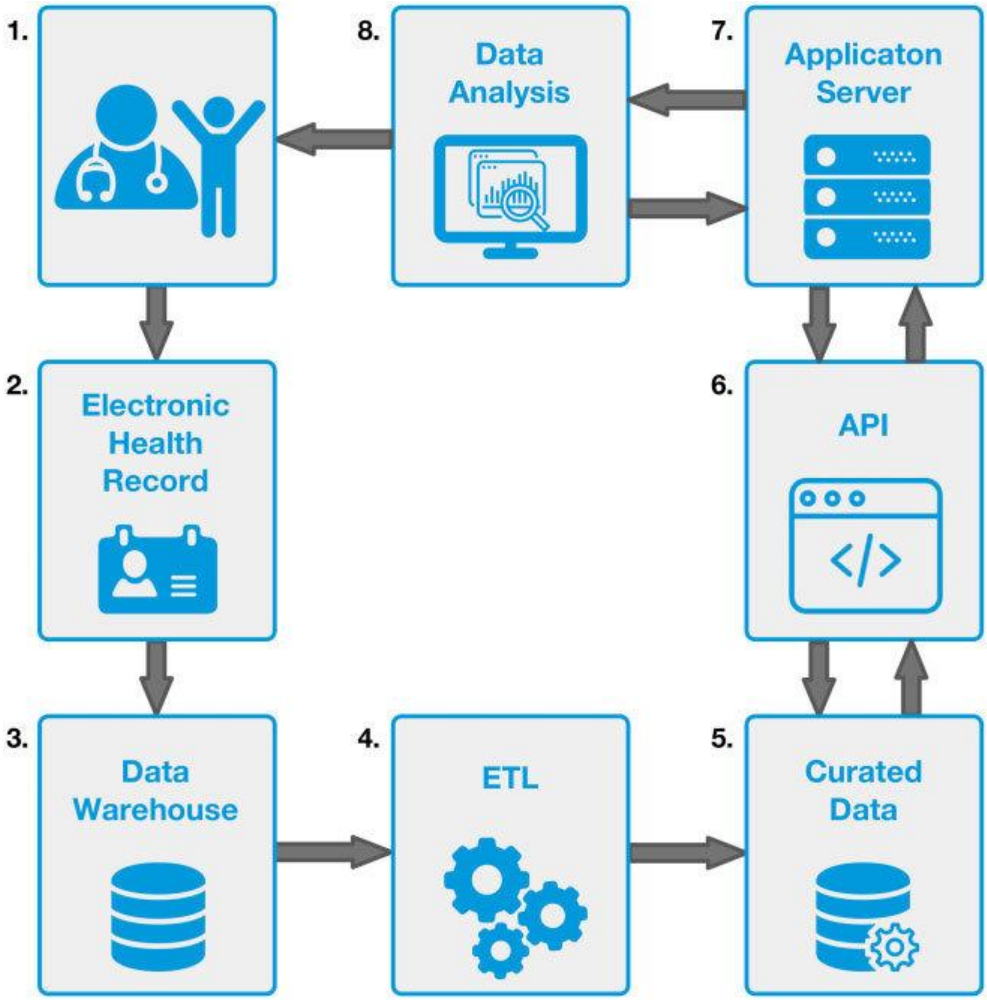
Σημαντικό κομμάτι των προκλήσεων που αντιμετωπίζει η Μηχανική Μάθηση στον τομέα της υγείας αφορά τα οικονομικά κόστη που συνδέονται τόσο με τη λειτουργία όσο και με την επίλυση άλλων προβλημάτων που προκύπτουν. Παρόλο που η ΜΜ υπόσχεται να βελτιώσει τη φροντίδα των ασθενών και να μειώσει το κόστος μακροπρόθεσμα, η αρχική επένδυση και τα συνεχή έξοδα που απαιτούνται είναι συχνά πολύ υψηλά για πολλά νοσοκομεία και ιδρύματα υγειονομικής περίθαλψης.

Ένα από τα βασικά κόστη είναι η ανάπτυξη των μοντέλων ML. Η δημιουργία ενός αποτελεσματικού συστήματος μηχανικής μάθησης απαιτεί όχι μόνο τεράστια ποσά δεδομένων αλλά και την πρόσληψη εξειδικευμένων επαγγελματιών με διεπιστημονική γνώση, που μπορούν να κατανοήσουν τόσο τα τεχνικά όσο και τα κλινικά δεδομένα. Οι ειδικοί αυτοί είναι σπάνιοι και πολύ ακριβοί. Επιπλέον, τα σύγχρονα μοντέλα, όπως τα βαθιά νευρωνικά δίκτυα, απαιτούν μεγάλη υπολογιστική ισχύ για την επεξεργασία των δεδομένων και την εκπαίδευση των μοντέλων, κάτι που προσθέτει σημαντικά στο συνολικό κόστος. Η υποδομή που χρειάζεται για τη φιλοξενία αυτών των αλγορίθμων περιλαμβάνει ακριβά συστήματα αποθήκευσης, εξυπηρετητές, και ειδικό λογισμικό, γεγονός που καθιστά την αρχική επένδυση πολύ υψηλή, ιδιαίτερα για μικρότερα νοσοκομεία ή κλινικές.

Επιπρόσθετα, η συνεχής συντήρηση και αναβάθμιση των μοντέλων ΜΜ είναι δαπανηρή. Τα μοντέλα χρειάζονται συνεχή ενημέρωση και επανεκπαίδευση, καθώς τα ιατρικά δεδομένα και οι μέθοδοι θεραπείας εξελίσσονται. Αυτό απαιτεί περαιτέρω επενδύσεις σε υπολογιστική υποδομή και ανθρώπινο δυναμικό. Επιπλέον, τα συστήματα αυτά πρέπει να είναι εξοπλισμένα για την

αντιμετώπιση του "model drift", δηλαδή της σταδιακής μείωσης της ακρίβειας ενός μοντέλου όταν τα δεδομένα που χρησιμοποιούνται αλλάζουν με τον καιρό. Για να παραμένουν ακριβή και ασφαλή στη χρήση, τα μοντέλα πρέπει να επανεξετάζονται και να επαναξιολογούνται συστηματικά, μια διαδικασία που απαιτεί πρόσθετους πόρους.

Εκτός από το κόστος ανάπτυξης και συντήρησης, υπάρχει το ζήτημα του κόστους για την επίλυση των προβλημάτων που προκύπτουν από την ίδια τη χρήση της ΜΜ. Ένα παράδειγμα είναι η ανάγκη για εξασφάλιση της ποιότητας των δεδομένων. Καθώς τα ιατρικά δεδομένα είναι συχνά ελλιπή ή ασυνεπή, είναι απαραίτητο να γίνουν προσεκτικές διαδικασίες προεπεξεργασίας δεδομένων, όπως η καθαριότητα, η ομαλοποίηση και η ανασυγκρότηση των πληροφοριών, κάτι που μπορεί να αυξήσει το λειτουργικό κόστος. Χωρίς την κατάλληλη επεξεργασία, τα δεδομένα μπορεί να οδηγήσουν σε αναξιόπιστα αποτελέσματα από τα μοντέλα ΜΜ, και οι διορθώσεις αυτών των προβλημάτων προσθέτουν επιπλέον οικονομική επιβάρυνση.



Τελικά, κάθε πρόκληση που ανακύπτει στην εφαρμογή της μηχανικής μάθησης στην υγεία, είτε είναι τεχνική είτε ηθική είτε σχετίζεται με τα δεδομένα, συνδέεται στενά με το κόστος. Χωρίς την απαραίτητη χρηματοδότηση για την αντιμετώπιση αυτών των προκλήσεων, τα πλεονεκτήματα της μηχανικής μάθησης μπορεί να αποδειχθούν απρόσιτα για πολλά ιδρύματα υγείας.

6. ΣΥΜΠΕΡΑΣΜΑ ΚΑΙ ΜΕΛΛΟΝΤΙΚΕΣ ΔΟΥΛΕΙΕΣ

Στην παρούσα διατριβή, εξετάσαμε και υλοποιήσαμε διάφορες τεχνικές μηχανικής μάθησης με στόχο την πρόβλεψη του κινδύνου εμφάνισης διαβήτη, χρησιμοποιώντας σύνολα δεδομένων που περιέχουν κλινικά και δημογραφικά χαρακτηριστικά των ασθενών. Η διαδικασία ξεκίνησε με την προεπεξεργασία των δεδομένων, η οποία περιλάμβανε την καθαριότητα των δεδομένων, την αφαίρεση των ελλιπών ή μη χρήσιμων χαρακτηριστικών, και την εφαρμογή μεθόδων τυποποίησης και κανονικοποίησης για τη βελτιστοποίηση των αλγορίθμων μηχανικής μάθησης.

Στη συνέχεια, υλοποιήσαμε και συγκρίναμε επιβλεπόμενους αλγορίθμους μηχανικής μάθησης, όπως η Λογιστική Παλινδρόμηση (Logistic Regression), τα Νευρωνικά Δίκτυα (Neural Networks), και το Τυχαίο Δάσος (Random Forest), για την ταξινόμηση των ασθενών σε διαβητικούς ή μη. Οι αλγόριθμοι αυτοί εκτιμούν την πιθανότητα να έχει ένας ασθενής διαβήτη με βάση τα χαρακτηριστικά εισόδου, όπως τα επίπεδα γλυκόζης, την αρτηριακή πίεση, και το σωματικό βάρος. Κάθε αλγόριθμος αξιολογήθηκε χρησιμοποιώντας μετρικές όπως η ακρίβεια, η πιστότητα (precision), η ανάκληση (recall) και ο δείκτης F1, εξασφαλίζοντας την ισορροπία μεταξύ της ακρίβειας και της ικανότητας πρόβλεψης σε νέα, άγνωστα δεδομένα.

Αφού βρήκαμε το καλύτερο μοντέλο βάσει ακρίβειας και άλλων μετρικών απόδοσης, προχωρήσαμε στην ανάλυση της ερμηνευσιμότητας του μοντέλου με τη βοήθεια της βιβλιοθήκης SHAP. Συγκεκριμένα, εξηγήσαμε ποιες μεταβλητές έπαιξαν τον πιο σημαντικό ρόλο στις προβλέψεις του αλγορίθμου, όπως τα επίπεδα γλυκοζυλιωμένης αιμοσφαιρίνης, η ηλικία , και οι δείκτες σωματικού βάρους. Αυτή η ανάλυση έδωσε τη δυνατότητα στους επαγγελματίες υγείας να κατανοήσουν πώς και γιατί ο αλγόριθμος κατέληξε σε συγκεκριμένες προβλέψεις, προσφέροντας ένα πολύτιμο εργαλείο για τη λήψη τεκμηριωμένων αποφάσεων.

Τέλος, δημιουργήσαμε μια διαδραστική εφαρμογή στη Julia, η οποία επιτρέπει στους χρήστες να εισάγουν κλινικά δεδομένα και να λαμβάνουν προβλέψεις για τον κίνδυνο εμφάνισης διαβήτη. Η εφαρμογή αυτή συνδυάζει την ακρίβεια του επιλεγμένου μοντέλου με μια εύχρηστη διεπαφή, καθιστώντας την πρακτική και επεκτάσιμη για μελλοντική χρήση τόσο στην κλινική πρακτική όσο και σε ερευνητικά περιβάλλοντα. Με την ανάπτυξη αυτής της εφαρμογής, δώσαμε τη δυνατότητα στους χρήστες να εκμεταλλευτούν τα οφέλη της μηχανικής μάθησης σε πραγματικό χρόνο και να αξιολογήσουν τον κίνδυνο εμφάνισης διαβήτη γρήγορα και αποτελεσματικά.

Ωστόσο, η έρευνα στον τομέα της μηχανικής μάθησης στον διαβήτη είναι ήδη αρκετά προχωρημένη σε σχέση με άλλους, λιγότερο ερευνημένους ιατρικούς τομείς. Για παράδειγμα, υπάρχουν κλάδοι της ιατρικής όπως οι σπάνιες νόσοι, οι οποίες δεν έχουν μελετηθεί επαρκώς με τη χρήση τεχνικών μηχανικής μάθησης, αλλά όπου τα μοντέλα αυτά μπορούν να προσφέρουν πολύτιμες λύσεις. Η έλλειψη μεγάλων συνόλων δεδομένων ή ετικετοποιημένων δεδομένων σε αυτούς τους τομείς μπορεί να καθιστά την επιβλεπόμενη μάθηση λιγότερο εφαρμόσιμη, γεγονός που καθιστά τις τεχνικές μη επιβλεπόμενης μάθησης ή ημι-επιβλεπόμενης μάθησης πιο κατάλληλες. Η μελλοντική έρευνα θα μπορούσε να επικεντρωθεί στην εφαρμογή αυτών των μεθόδων σε περιοχές της ιατρικής που δεν είναι τόσο καλά ερευνημένες, επιτρέποντας την ανακάλυψη νέων γνώσεων και την ανάπτυξη καλύτερων διαγνωστικών εργαλείων.

Ένα κρίσιμο ζήτημα για τη μελλοντική ανάπτυξη των αλγορίθμων μηχανικής μάθησης στην ιατρική είναι η ανάγκη για επεξηγήσιμη τεχνητή νοημοσύνη (Explainable AI - XAI). Η επεξηγήσιμη τεχνητή νοημοσύνη μπορεί να προσφέρει διαφάνεια στον τρόπο λειτουργίας των αλγορίθμων, επιτρέποντας στους ιατρούς να κατανοήσουν πώς και γιατί λαμβάνονται συγκεκριμένες αποφάσεις από ένα μοντέλο. Ειδικότερα, η ανάπτυξη αλγορίθμων που όχι μόνο παρέχουν ακριβείς προβλέψεις, αλλά μπορούν να εξηγήσουν με σαφήνεια τα βήματα που ακολουθούν και τα χαρακτηριστικά που επηρεάζουν τις προβλέψεις, θα ενισχύσει την εμπιστοσύνη και την αποδοχή της τεχνητής νοημοσύνης στην ιατρική κοινότητα. Μελλοντικές μελέτες θα πρέπει να δώσουν έμφαση στην ενσωμάτωση επεξηγήσιμων μοντέλων που μπορούν να ερμηνεύσουν καλύτερα τις διαδικασίες των αλγορίθμων, καθιστώντας τα εργαλεία μηχανικής μάθησης όχι μόνο ισχυρά αλλά και διαφανή για τους τελικούς χρήστες.

Βιβλιογραφία

[1] https://www.kaggle.com/datasets/imtkaggleteam/diabetes

[2] https://www.geeksforgeeks.org/support-vector-machine-algorithm/

[3] A. Alanazi, *Using Machine Learning for healthcare challenges and opportunities*, (2022) https://www.sciencedirect.com/science/article/pii/S2352914822000739

[4] S.Brown, *Machine Learning Explained*, (2021) https://mitsloan.mit.edu/ideas-made-to-matter/machine-learning-explained

[5] S.R.Gallego, *A survey on data preprocessing for data stream mining: Current status and future directions*, (2017) https://www.sciencedirect.com/science/article/abs/pii/S0925231217302631?casa_token=8BHb1EK0q_8AAAAA:5K2QnAKHKTL3vIsyDZ7l_YWyLPlNA_515qAz4-8k-jwMJFGLcn_Q5CU7Ef9ok_9QMXEyQ0Qj2hE

[6] I. Kavakiotis, *Machine Learning and Data Mining Methods in Diabetes Research*, (2017), https://www.sciencedirect.com/science/article/pii/S2001037016300733#bb0070

[7] J. Kent, *Over 80% of Health Execs have Artificial Intelligence Plans in Place*, (2020) https://www.techtarget.com/healthtechanalytics/news/366591450/Over-80-of-Health-Execs-Have-Artificial-Intelligence-Plans-in-Place

[8] S. Leleko, *10 Real-World Machine Learning Projects in Healthcare*, (2024) https://spd.tech/machine-learning/machine-learning-in-healthcare/

[9] B. Mahesh, *Machine Learning Algorithms – A review*, (2019) https://www.researchgate.net/profile/Batta-Mahesh/publication/344717762_Machine_Learning_Algorithms_-A_Review/links/5f8b2365299bf1b53e2d243a/Machine-Learning-Algorithms-A-Review.pdf?eid=5082902844932096t

[10] B. Siwicki, 86% of Healthcare companies use some form of AI, (2017) https://www.healthcareitnews.com/news/86-healthcare-companies-use-some-form-ai

Πηγές Εικόνων

[1] https://medlineplus.gov/genetics/condition/type-2-diabetes/

[2] https://towardsdatascience.com/the-balance-accuracy-vs-interpretability-1b3861408062

[3] https://easyai.tech/en/ai-definition/gradient-descent/

[4] https://dev.to/petercour/machine-learning-classification-vs-regression-1gn.

[5] https://medium.com/analytics-vidhya/kernel-support-vector-machines-from-scratch-483ebd4175c

[6] https://www.linkedin.com/pulse/understanding-logistic-regression-model-laymans-words-omkar-sutar

[7] https://medium.com/@denizgunay/random-forest-af5bde5d7e1e

[8] https://www.researchgate.net/figure/Flow-diagram-of-gradient-boosting-machine-learning-method-The-ensemble-classifiers_fig1_351542039

[9] https://medium.com/mit-6-s089-intro-to-quantum-computing/quantum-neural-networks-7b5bc469d984

[10] https://medium.com/aimonks/linear-discriminant-analysis-lda-in-machine-learning-example-concept-and-applications-37f27e7c7e98

[11] https://www.biorender.com/template/principal-component-analysis-pca-transformation

[12] https://www.allerin.com/blog/5-ai-implementation-challenges-in-healthcare

[13] https://spd.tech/machine-learning/machine-learning-in-healthcare/

[14] https://www.researchgate.net/figure/Dataflow-The-learning-health-system-data-flow-1-Clinician-and-patient-encounters-are_fig2_347205458