# ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ

## ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΕΠΙΚΟΙΝΩΝΙΩΝ

## ΤΜΗΜΑ ΨΗΦΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

# ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ

## "ΠΛΗΡΟΦΟΡΙΑΚΑ ΣΥΣΤΗΜΑΤΑ & ΥΠΗΡΕΣΙΕΣ"

**Τίτλος: Πρόβλεψη διαβήτη χρησιμοποιώντας Data mining αλγορίθμους**


**Όνομα Επώνυμο**
Γιαννέλου Αθανασία (ΑΜ: me2235)


Υποβάλλεται
για την εκπλήρωση των προϋποθέσεων λήψης
Μεταπτυχιακού Διπλώματος
στην ειδίκευση «ΜΔΑ/ΠΠΣ/ΠΔ»
του ΠΜΣ "Πληροφοριακά Συστήματα & Υπηρεσίες"
στο
**ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ**
**Φεβρουάριος 2025**


**Επιβλέπων/Επιβλέπουσα:** ΦΙΛΙΠΠΑΚΗΣ ΜΙΧΑΗΛ
**Ακαδημαϊκή Θέση:** Καθηγητής

## ΣΕΛΙΔΑ ΕΓΚΥΡΟΤΗΤΑΣ

**Ονοματεπώνυμο Φοιτητή/Φοιτήτριας:** Γιαννέλου Αθανασία

**Τίτλος Μεταπτυχιακής Διπλωματικής Εργασίας:** Πρόβλεψη διαβήτη χρησιμοποιώντας Data mining αλγορίθμους

*Η παρούσα Μεταπτυχιακή Διπλωματική Εργασία υποβάλλεται ως μερική εκπλήρωση των απαιτήσεων του Προγράμματος Μεταπτυχιακών Σπουδών "Πληροφοριακά Συστήματα & Υπηρεσίες" του Τμήματος Ψηφιακών Συστημάτων του Πανεπιστημίου Πειραιώς και εγκρίθηκε στις …… [ημερομηνία έγκρισης] από τα μέλη της Εξεταστικής Επιτροπής.*

**Εξεταστική Επιτροπή**

Επιβλέπων/ουσα (Τμήμα Ψηφιακών Συστημάτων, Πανεπιστήμιο Πειραιώς)……[ονοματεπώνυμο, βαθμίδα]

Μέλος Εξεταστικής Επιτροπής: ……[ονοματεπώνυμο, βαθμίδα]

Μέλος Εξεταστικής Επιτροπής: ……[ονοματεπώνυμο, βαθμίδα]

## ΥΠΕΥΘΥΝΗ ΔΗΛΩΣΗ ΑΥΘΕΝΤΙΚΟΤΗΤΑΣ

*Ο/Η Athanasia Giannelou., γνωρίζοντας τις συνέπειες της λογοκλοπής, δηλώνω υπεύθυνα ότι η παρούσα εργασία με τίτλο* ***Diabetes Prediction*** *αποτελεί προϊόν αυστηρά προσωπικής εργασίας και όλες οι πηγές που έχω χρησιμοποιήσει, έχουν δηλωθεί κατάλληλα στις βιβλιογραφικές παραπομπές και αναφορές. Τα σημεία όπου έχω χρησιμοποιήσει ιδέες, κείμενο ή/και πηγές άλλων συγγραφέων, αναφέρονται ευδιάκριτα στο κείμενο με την κατάλληλη παραπομπή και η σχετική αναφορά περιλαμβάνεται στο τμήμα των βιβλιογραφικών αναφορών με πλήρη περιγραφή.*
Επιπλέον *δηλώνω υπεύθυνα ότι η συγκεκριμένη Μεταπτυχιακή Διπλωματική Εργασία έχει συγγραφεί από εμένα προσωπικά και δεν έχει υποβληθεί ούτε έχει αξιολογηθεί στο πλαίσιο κάποιου άλλου μεταπτυχιακού ή προπτυχιακού τίτλου σπουδών, στην Ελλάδα ή στο εξωτερικό. Παράβαση της ανωτέρω ακαδημαϊκής μου ευθύνης αποτελεί ουσιώδη λόγο για την ανάκληση του πτυχίου μου. Σε κάθε περίπτωση, αναληθούς ή ανακριβούς δηλώσεως, υπόκειμαι στις συνέπειες που προβλέπονται τις διατάξεις που προβλέπει η Ελληνική και Κοινοτική Νομοθεσία περί πνευματικής ιδιοκτησίας.*

**Ο/Η ΔΗΛΩΝ/ΟΥΣΑ**

**Ονοματεπώνυμο: Γιαννέλου Αθανασία**
**Αριθμός Μητρώου: me2235**
**Υπογραφή:**

# Πίνακας Περιεχομένου

## Table of Figures

# 1. Περίληψη

Οι τεχνολογικές εξελίξεις και η εξέλιξη στον τομέα της ανάλυσης και επεξεργασίας δεδομένων συνέβαλαν καθοριστικά στην επανάσταση στις πρακτικές υγειονομικής περίθαλψης, ιδίως στην ενίσχυση των ικανοτήτων στην πρόληψη ασθενειών, στην ακριβή διάγνωση και στις αποτελεσματικές θεραπείες. Η ενσωμάτωση τεχνικών μηχανικής μάθησης έχει συμβάλει σημαντικά στη βελτίωση της διαγνωστικής διαδικασίας για ασθένειες όπως ο διαβήτης, μια σημαντική ανησυχία για την παγκόσμια υγεία. Αυτή η ερευνητική εργασία αξιοποιεί αυτά τα τεχνολογικά βήματα για τη διερεύνηση και την αξιολόγηση μεθοδολογιών που στοχεύουν στην προώθηση της ακρίβειας της διάγνωσης του διαβήτη. Ερευνά την αποτελεσματικότητα διαφόρων αλγορίθμων μηχανικής μάθησης μέσω μιας συγκριτικής ανάλυσης για να εντοπίσει την προσέγγιση που αποδίδει την υψηλότερη προγνωστική ακρίβεια για αυτήν την κατάσταση. Η επίτευξη αυτού του στόχου περιλάμβανε τη χρήση ενός συνόλου δεδομένων για τον διαβήτη ως πεδίο δοκιμών για αυτούς τους αλγόριθμους, επιτρέποντας τη λεπτομερή αξιολόγηση και την αντιπαράθεση των αποτελεσμάτων. Μια τέτοια εις βάθος ανάλυση δεδομένων διευκόλυνε μια λεπτή κατανόηση και προσδιορισμό της βέλτιστης στρατηγικής για μια ακριβή και αξιόπιστη διάγνωση του διαβήτη [1].

Η μελέτη υπογράμμισε τον κρίσιμο ρόλο της επιλογής και της σύγκρισης διαφορετικών στρατηγικών μηχανικής μάθησης, αξιολογώντας σχολαστικά την αποτελεσματικότητα καθεμιάς στην ακριβή πρόβλεψη του διαβήτη. Η επιλογή ενός κατάλληλου συνόλου δεδομένων αναδείχθηκε ως κρίσιμο στοιχείο για την επιτυχία της έρευνας, υπογραμμίζοντας πώς η ποιότητα και η καταλληλόλητα των δεδομένων επηρεάζουν άμεσα την απόδοση των αλγορίθμων που εφαρμόζονται. Μέσω αυτής της αναλυτικής διαδικασίας, η έρευνα παρέχει βαθιές γνώσεις για την επιλογή της καταλληλότερης τεχνολογικής τεχνικής για τη διάγνωση του διαβήτη με υψηλή αξιοπιστία και ακρίβεια.

Αυτή η διατριβή υπογραμμίζει την επιτακτική ανάγκη της ύφανσης των τεχνολογικών καινοτομιών αιχμής στον ιστό της ιατρικής έρευνας και πρακτικής. Θέτει μια ισχυρή βάση για μελλοντικές ανακαλύψεις στην έγκαιρη ανίχνευση, πρόληψη και διαχείριση του διαβήτη, με πρωταρχικό στόχο τη βελτίωση της ποιότητας ζωής όσων επηρεάζονται από τη νόσο. Υπογραμμίζοντας τη σημασία της τεχνολογικής προόδου στην υγειονομική περίθαλψη, το έγγραφο υποστηρίζει τη συνεχή εστίαση στην καινοτομία ως καταλύτη για τη βελτίωση των διαγνωστικών μεθόδων και των τρόπων θεραπείας. Με αυτόν τον τρόπο, όχι μόνο συμβάλλει στον ακαδημαϊκό και πρακτικό λόγο για τη διαχείριση του διαβήτη, αλλά χρησιμεύει επίσης ως φάρος για μελλοντική έρευνα με στόχο την αντιμετώπιση άλλων διαδεδομένων ζητημάτων υγείας μέσω τεχνολογικής εφευρετικότητας. Αυτή η διερεύνηση της δυνατότητας της μηχανικής μάθησης να βελτιώσει τη διάγνωση του διαβήτη αποτελεί παράδειγμα των ευρύτερων επιπτώσεων της τεχνολογίας στον μετασχηματισμό της υγειονομικής περίθαλψης, δίνοντας έμφαση σε μια μακροπρόθεσμη προσέγγιση στην ιατρική έρευνα και τη φροντίδα των ασθενών.

Η αδιάκοπη πρόοδος στον τομέα της τεχνολογίας και η εξέλιξη στην ανάλυση δεδομένων έχουν συμβάλλει σημαντικά στην ανάπτυξη της ιατρικής επιστήμης, παρέχοντας

πρωτοποριακές λύσεις για την εντοπισμό και αποτελεσματική αντιμετώπιση πληθώρας ασθενειών. Ειδικά στον τομέα της ακριβέστερης διάγνωσης νοσημάτων, η τεχνητή νοημοσύνη και ιδίως η μηχανική μάθηση έχουν δείξει εντυπωσιακά αποτελέσματα, βελτιώνοντας τη διαδικασία αναγνώρισης και αντιμετώπισης διαφόρων παθήσεων, όπως ο διαβήτης.

Αυτή η μεταπτυχιακή διατριβή επικεντρώνεται στην εκμετάλλευση των τελευταίων επιτευγμάτων στο πεδίο της τεχνητής νοημοσύνης για τη βελτίωση των διαγνωστικών διαδικασιών του διαβήτη. Αναλύοντας και συγκρίνοντας διάφορους αλγορίθμους μηχανικής μάθησης, η διατριβή αναζητά την πιο αποδοτική στρατηγική για την πρόγνωση της πάθησης με την υψηλότερη δυνατή ακρίβεια. Η επιτυχής αναγνώριση και έγκαιρη θεραπεία του διαβήτη εξαρτάται κρίσιμα από την επιλογή και την αποδοτικότητα των αλγορίθμων που εφαρμόζονται στην επεξεργασία των δεδομένων.

Καθ' όλη τη διάρκεια της έρευνας, υπήρξε ένας σημαντικό επίκεντρο στην ανάπτυξη μοντέλων μηχανικής μάθησης, προσαρμοσμένων στις προκλήσεις που παρουσιάζει η ασθένεια του διαβήτη. Η επιλογή των κατάλληλων μοντέλων διενεργήθηκε με βάση μια ενδελεχή ανάλυση των διαθέσιμων δεδομένων, επιδιώκοντας να ανακαλύψει την πλέον αποτελεσματική τεχνική για την προσδιορισμό της πάθησης.

Η έρευνα αυτή καταδεικνύει το θετικό αντίκτυπο της συνεργασίας τεχνολογικών καινοτομιών με τον ιατρικό κλάδο, ανοίγοντας νέους ορίζοντες για την προληπτική φροντίδα και αποτελεσματική αντιμετώπιση του διαβήτη και άλλων ασθενειών. Επισημαίνει τη σημαντικότητα της ενσωμάτωσης προηγμένων τεχνολογικών διαδικασιών στην ιατρική πρακτική για τη βελτίωση της ακρίβειας των διαγνώσεων και της αποτελεσματικότητας των θεραπειών, ενισχύοντας την ποιότητα ζωής των ασθενών.

**English Version**

Technological advancements and developments in data analysis and processing have played a pivotal role in revolutionizing healthcare practices, particularly in enhancing capabilities for disease prevention, accurate diagnosis, and effective treatments. The integration of machine learning techniques has significantly contributed to improving the diagnostic process for diseases like diabetes, a major global health concern. This research leverages these technological strides to explore and evaluate methodologies aimed at advancing the accuracy of diabetes diagnosis. It investigates the effectiveness of various machine learning algorithms through comparative analysis to identify the approach that yields the highest predictive accuracy for this condition. Achieving this objective involved utilizing a diabetes dataset as a testbed for these algorithms, allowing for a detailed assessment and comparison of the results. Such an in-depth data analysis facilitated a nuanced understanding and identification of the optimal strategy for a precise and reliable diabetes diagnosis.

The study highlighted the crucial role of selecting and comparing different machine learning strategies, meticulously evaluating each one’s effectiveness in accurately predicting diabetes. The choice of an appropriate dataset emerged as a critical element

for the research's success, emphasizing how data quality and relevance directly impact the performance of the applied algorithms. Through this analytical process, the research provides profound insights into selecting the most suitable technological technique for diagnosing diabetes with high reliability and precision.

This thesis underscores the imperative of weaving cutting-edge technological innovations into the fabric of medical research and practice. It lays a strong foundation for future breakthroughs in early detection, prevention, and management of diabetes, with the primary goal of improving the quality of life for those affected by the disease. Highlighting the importance of technological advancement in healthcare, the paper advocates for continued focus on innovation as a catalyst for improving diagnostic methods and treatment approaches. In doing so, it not only contributes to the academic and practical discourse on diabetes management but also serves as a beacon for future research aimed at addressing other widespread health issues through technological ingenuity. This exploration of the potential of machine learning to enhance diabetes diagnosis exemplifies the broader impact of technology on transforming healthcare, emphasizing a long-term approach to medical research and patient care.

The continuous progress in the field of technology and developments in data analysis have significantly contributed to the advancement of medical science, providing groundbreaking solutions for identifying and effectively addressing a wide range of diseases. Specifically, in the area of more accurate disease diagnosis, artificial intelligence and especially machine learning have shown impressive results, enhancing the process of recognizing and managing various conditions like diabetes.

This master's thesis focuses on leveraging the latest advancements in artificial intelligence to improve the diagnostic processes of diabetes. By analyzing and comparing various machine learning algorithms, the thesis seeks to identify the most efficient strategy for predicting the condition with the highest possible accuracy. The successful identification and timely treatment of diabetes critically depend on the selection and effectiveness of the algorithms applied in data processing.

Throughout the research, there was significant emphasis on developing machine learning models tailored to the challenges posed by the disease of diabetes. The selection of appropriate models was based on a thorough analysis of available data, aiming to uncover the most effective technique for determining the condition.

This research demonstrates the positive impact of combining technological innovations with the medical field, opening new horizons for preventive care and effective management of diabetes and other diseases. It emphasizes the importance of integrating advanced technological processes into medical practice to improve the accuracy of diagnoses and the effectiveness of treatments, ultimately enhancing patients' quality of life.

## 2. Εισαγωγή

Ο διαβήτης αποτελεί μία από τις πιο διαδεδομένες χρόνιες παθήσεις παγκοσμίως, με

σημαντικό αντίκτυπο τόσο στην ατομική υγεία όσο και στα συστήματα υγείας των χωρών. Η ασθένεια χαρακτηρίζεται από την ανεπάρκεια του οργανισμού να παράγει ή να χρησιμοποιήσει επαρκώς την ινσουλίνη, η οποία είναι απαραίτητη για τη μεταφορά της γλυκόζης από το αίμα στα κύτταρα. Ως αποτέλεσμα, η γλυκόζη συσσωρεύεται στο αίμα, οδηγώντας σε υψηλά επίπεδα σακχάρου.

### 2.1 Τύποι Διαβήτη

Η ασθένεια του διαβήτη έχει αρκετές διαφοροποιήσεις [2]**:**

- **Διαβήτης τύπου 1**: Πρόκειται για αυτοάνοση πάθηση όπου το πάγκρεας δεν παράγει ινσουλίνη. Συνήθως διαγιγνώσκεται σε παιδιά και νέους ενήλικες, αλλά μπορεί να εμφανιστεί σε οποιαδήποτε ηλικία.
- **Διαβήτης τύπου 2**: Αυτός είναι ο πιο κοινός τύπος διαβήτη και συνήθως εμφανίζεται σε ενήλικες. Χαρακτηρίζεται από την ανεπάρκεια του οργανισμού να χρησιμοποιήσει επαρκώς την ινσουλίνη που παράγει.
- **Εγκυμοσυνικός διαβήτης**: Εμφανίζεται κατά τη διάρκεια της εγκυμοσύνης και συνήθως εξαφανίζεται μετά τη γέννηση. Ωστόσο, οι γυναίκες που αναπτύσσουν εγκυμοσυνικό διαβήτη έχουν αυξημένο κίνδυνο για την εμφάνιση διαβήτη τύπου 2 στο μέλλον.
- **Διαβήτης της Νεανικής Ηλικίας (MODY)**: Ο MODY είναι μια μορφή διαβήτη που οφείλεται σε μετάλλαξη συγκεκριμένου γονιδίου και παρουσιάζεται συνήθως σε νέους κάτω των 25 ετών. Παρότι είναι γενετικής φύσεως, οι θεραπείες του μπορεί να μην απαιτούν ινσουλίνη.
- **Νεογνικός Διαβήτης:** Αυτή η μορφή διαβήτη εμφανίζεται σε βρέφη και συνήθως είναι παροδική, αν και μπορεί να είναι και μόνιμη. Σχετίζεται με γενετικές ανωμαλίες και απαιτεί έλεγχο και θεραπευτική αντιμετώπιση από πολύ νωρίς.
- **Σύνδρομο Wolfram:** Είναι μια σπάνια κληρονομική διαταραχή που σχετίζεται με τον διαβήτη τύπου 1 και άλλες σοβαρές ανωμαλίες όπως νευρολογικές και οφθαλμολογικές.
- **Σύνδρομο Alström:** Πρόκειται για μία ακόμα σπάνια γενετική διαταραχή που προκαλεί συμπτώματα όπως διαβήτη, προβλήματα όρασης και ακοής, καθώς και παχυσαρκία.
- **Latent Autoimmune diabetes in Adults (LADA):** Ο Αργοπρόβλητος Αυτοάνοσος Διαβήτης στους Ενήλικες (LADA) είναι μια μορφή διαβήτη που μοιάζει κλινικά με τον διαβήτη τύπου 1, αλλά εμφανίζεται συνήθως σε μεγαλύτερες ηλικίες. Στην πάθηση αυτή, το πάγκρεας παύει σταδιακά να παράγει ινσουλίνη, συχνά προκαλώντας σύγχυση με τον διαβήτη τύπου 2, εξ ου και η ονομασία "αργοπρόβλητος". Οι ασθενείς με LADA δεν είναι συνήθως υπέρβαροι, όπως είναι συνηθισμένο στον διαβήτη τύπου 2, και μπορεί να ανταποκρίνονται στην αρχή στα αντιδιαβητικά φάρμακα, αλλά συνήθως χρειάζονται ινσουλίνη νωρίτερα στην πορεία της νόσου. Η διαγνωστική διαδικασία συχνά απαιτεί μέτρηση των αυτό-αντισωμάτων για να επιβεβαιωθεί η αυτό-άνοση φύση της νόσου.
- **Διαβήτης Τύπου 3c:** Ο διαβήτης τύπου 3c, επίσης γνωστός ως παγκρεατογενής διαβήτης, είναι μια μορφή της νόσου που προκαλείται από βλάβη ή πάθηση του παγκρέατος, όπως παγκρεατίτιδα, κυστική ίνωση, ηπατική νόσος ή επέμβαση στο

πάγκρεας. Αυτή η κατάσταση επηρεάζει τόσο την παραγωγή της ινσουλίνης όσο και των πεπτικών ενζύμων, οδηγώντας σε σακχαρώδη διαβήτη.

- **Διαβήτης Που Προκαλείται από Στεροειδή:** Ο στεροειδής διαβήτης είναι μια διαταραχή στον μεταβολισμό της γλυκόζης που εμφανίζεται ως απόκριση στη χρήση εξωγενών στεροειδών ορμονών, όπως πρεδνιζόνη. Τα στεροειδή αυξάνουν την αντίσταση στην ινσουλίνη και μπορούν να προκαλέσουν υψηλά επίπεδα σακχάρου σε άτομα με υπάρχουσα ή κρυφή προδιάθεση για διαβήτη.
- **Διαβήτης Κυστικής Ίνωσης:** Ο διαβήτης κυστικής ίνωσης είναι ένας τύπος διαβήτη που συχνά παρουσιάζεται σε ασθενείς με κυστική ίνωση, μια γενετική διαταραχή που προκαλεί παχύρρευστες βλέννες, επηρεάζοντας πολλά όργανα, συμπεριλαμβανομένου του παγκρέατος. Η βλάβη στο πάγκρεας μειώνει την παραγωγή της ινσουλίνης, οδηγώντας σε διαβήτη.

Στην Ελλάδα, ο διαβήτης αποτελεί σημαντική δημόσια υγειονομική πρόκληση, με το ποσοστό της νόσου να είναι συγκρίσιμο με αυτό πολλών δυτικών χωρών. Η υψηλή διάδοση της νόσου σχετίζεται με παράγοντες όπως οι αλλαγές στον τρόπο ζωής, η αυξανόμενη παχυσαρκία και η γήρανση του πληθυσμού. Η πρόληψη και η διαχείριση της νόσου απαιτούν συνεχή ενημέρωση και εκπαίδευση του πληθυσμού, καθώς και την εφαρμογή στοχευμένων δημόσιων υγειονομικών πολιτικών [3].

# 3. Η Συμβολή του Machine Learning

Το Machine Learning (ML) προσφέρει μοναδικές δυνατότητες στην πρόληψη και πρόβλεψη του διαβήτη. Με την ανάλυση μεγάλων όγκων δεδομένων από ιατρικά αρχεία, δημοσιογραφικές πηγές και ατομικά ιστορικά, τα ML μοντέλα μπορούν να αναγνωρίσουν πρότυπα και να προβλέψουν την εμφάνιση του διαβήτη σε άτομα υψηλού κινδύνου πριν ακόμα εμφανιστούν συμπτώματα. Αυτό επιτρέπει την έγκαιρη παρέμβαση και την υιοθέτηση προληπτικών μέτρων, όπως η αλλαγή του τρόπου ζωής και η διατροφή, με στόχο τη μείωση του κινδύνου εμφάνισης της νόσου. Επιπλέον, το ML μπορεί να βελτιώσει την προσωπική ιατρική, παρέχοντας πιο στοχευμένες και εξατομικευμένες θεραπευτικές προσεγγίσεις βασισμένες στα ατομικά χαρακτηριστικά και την ιστορική ανταπόκριση του κάθε ασθενούς. Η ενσωμάτωση του ML στα συστήματα υγείας μπορεί να οδηγήσει σε πιο αποτελεσματική και οικονομικά αποδοτική διαχείριση του διαβήτη, βελτιώνοντας την ποιότητα ζωής των ασθενών και μειώνοντας τον βάρος που επωμίζονται τα συστήματα υγείας.

## 3.1 Διαφορετικές μέθοδοι μηχανικής μάθησης

Μπορεί κανείς να προσεγγίσει το πρόβλημα με διάφορες μεθοδολογίες όπως το supervised learning, unsupervised learning, reinforcment learning καθώς και άλλες τεχνικές κάθε μια από τις οποίες προσφέρει διαφορετικούς αλγόριθμους και approaches για τη μοντελοποίηση και επίλυση του προβλήματος [4] [5] [6].

### 3.1.1 Supervised Learning

Το Supervised Learning είναι ένα υποσύνολο του Machine Learning όπου το μοντέλο εκπαιδεύεται σε ένα επισημειωμένο σύνολο δεδομένων, πράγμα που σημαίνει ότι κάθε παράδειγμα εκπαίδευσης συνοδεύεται από ένα output label. Αυτή η προσέγγιση επιτρέπει στο μοντέλο να μάθει την αντιστοίχιση από τις εισόδους στις εξόδους, κάτι που μπορεί να είναι ιδιαίτερα χρήσιμο σε διάφορες εργασίες προβλεπτικής μοντελοποίησης, συμπεριλαμβανομένων των εφαρμογών υγειονομικής περίθαλψης όπως η πρόβλεψη του διαβήτη.

Στο πλαίσιο της πρόβλεψης του διαβήτη, οι αλγόριθμοι Supervised Learning μπορούν να αναλύσουν ιστορικά ιατρικά δεδομένα, όπου κάθε εγγραφή περιλαμβάνει διάφορα χαρακτηριστικά όπως η ηλικία, το φύλο, ο δείκτης μάζας σώματος (BMI), τα επίπεδα γλυκόζης, η αρτηριακή πίεση και η παρουσία ή απουσία διαβήτη. Μαθαίνοντας από αυτά τα δεδομένα, το μοντέλο μπορεί να αναγνωρίσει μοτίβα και σχέσεις μεταξύ των χαρακτηριστικών και της πιθανότητας διαβήτη. Αυτό επιτρέπει στους επαγγελματίες υγείας να αξιολογήσουν τον κίνδυνο διαβήτη σε άτομα με βάση τα ιατρικά τους προφίλ, οδηγώντας σε πρώιμες παρεμβάσεις και εξατομικευμένα σχέδια θεραπείας.
Κοινοί αλγόριθμοι Supervised Learning που χρησιμοποιούνται στην πρόβλεψη του διαβήτη περιλαμβάνουν:

- το logistic regression
- τα δέντρα αποφάσεων
- τα support vector machines (SVM)
- τα νευρωτικά δίκτυα.

Κάθε ένας από αυτούς τους αλγόριθμους έχει τα δυνατά του σημεία και επιλέγεται με βάση τα συγκεκριμένα χαρακτηριστικά των δεδομένων και τις απαιτήσεις της εργασίας πρόβλεψης.

- Το **logistic regression** χρησιμοποιείται συχνά για προβλήματα δυαδικής ταξινόμησης, όπως η πρόβλεψη εάν ένα άτομο έχει διαβήτη (ναι ή όχι). Λειτουργεί καλά με δεδομένα που μπορούν να διαχωριστούν γραμμικά και είναι εύκολη στην υλοποίηση και στην ερμηνεία, καθιστώντας την δημοφιλή επιλογή για ιατρικές προβλέψεις.
- Τα **δέντρα αποφάσεων** είναι άλλη μια κοινή επιλογή, ικανά να χειριστούν και αριθμητικά και κατηγορικά δεδομένα. Διαχωρίζουν τα δεδομένα σε υποσύνολα με βάση την τιμή των εισαγωγικών χαρακτηριστικών, κάνοντας αποφάσεις σε κάθε κόμβο που οδηγούν στην τελική πρόβλεψη. Τα δέντρα αποφάσεων είναι διαισθητικά και εύκολα στην οπτικοποίηση.
- Τα **Support Vector Machines (SVM)** είναι ισχυρά για εργασίες ταξινόμησης, συμπεριλαμβανομένης της πρόβλεψης διαβήτη. Τα SVM βρίσκουν το υπερεπίπεδο που διαχωρίζει καλύτερα τα δεδομένα σε κλάσεις, μεγιστοποιώντας το περιθώριο μεταξύ των πλησιέστερων σημείων των κλάσεων (support vectors). Τα SVM είναι αποτελεσματικά σε χώρους υψηλής διάστασης και όταν οι κλάσεις δεν είναι γραμμικά διαχωρίσιμες.
- Τα **Neural Networks**, ιδιαίτερα τα μοντέλα Deep Learning, έχουν αποκτήσει δημοτικότητα στις προβλέψεις υγειονομικής περίθαλψης λόγω της ικανότητάς τους

να χειρίζονται μεγάλα και περίπλοκα σύνολα δεδομένων. Αποτελούνται από πολλαπλά επίπεδα διασυνδεδεμένων κόμβων (νευρώνων) που μπορούν να μάθουν μη γραμμικές σχέσεις στα δεδομένα. Τα Neural Networks απαιτούν σημαντικούς υπολογιστικούς πόρους και δεδομένα αλλά μπορούν να επιτύχουν υψηλή ακρίβεια σε εργασίες όπως η πρόβλεψη διαβήτη.

Στα Supervised Learning Algorithms για την πρόβλεψη του διαβήτη, είναι κρίσιμη η προεπεξεργασία των δεδομένων, ο χειρισμός των τιμών που λείπουν και η επιλογή των σχετικών χαρακτηριστικών ώστε να διασφαλίσουμε την απόδοση του μοντέλου. Επιπλέον, η αξιολόγηση του μοντέλου χρησιμοποιώντας κατάλληλες μετρικές όπως η ακρίβεια, η ακριβότητα, η ανάκληση και η περιοχή κάτω από την καμπύλη ROC (AUC-ROC) είναι ουσιώδης για την εκτίμηση των προβλεπτικών του ικανοτήτων.

Αξιοποιώντας τους αλγορίθμους supervised learning, τα συστήματα υγειονομικής περίθαλψης μπορούν να ενισχύσουν την ικανότητά τους να προβλέπουν τον διαβήτη, επιτρέποντας την πρώιμη διάγνωση και τα προσωποποιημένα σχέδια θεραπείας. Αυτό δεν βελτιώνει μόνο τα αποτελέσματα για τους ασθενείς αλλά μειώνει επίσης το συνολικό βάρος του διαβήτη στα συστήματα υγειονομικής περίθαλψης [7].

### 3.1.2 Unsupervised Learning

Το Unsupervised Learning είναι μια σημαντική κατηγορία του Machine Learning που δεν χρησιμοποιεί επισημασμένα δεδομένα για την εκπαίδευση των μοντέλων. Αντίθετα, ο αλγόριθμος unsupervised learning αναλύουν και προσπαθούν να κατανοήσουν τα δεδομένα χωρίς προκαθορισμένες απαντήσεις ή ετικέτες, αναζητώντας κρυφές δομές ή μοτίβα στα δεδομένα. Αυτό μπορεί να είναι ιδιαίτερα χρήσιμο σε περιπτώσεις όπου οι ετικέτες δεν είναι διαθέσιμες ή είναι δύσκολο να τις λάβουμε, όπως συμβαίνει συχνά στην ιατρική έρευνα και την πρόβλεψη ασθενειών, όπως ο διαβήτης.

Η εφαρμογή του unsupervised learning στην πρόβλεψη του διαβήτη μπορεί να παρέχει πολύτιμες πληροφορίες σχετικά με τη δομή και τις σχέσεις μεταξύ διαφόρων βιολογικών και ιατρικών δεδομένων. Για παράδειγμα, η τεχνική της ομαδοποίησης (clustering) μπορεί να χρησιμοποιηθεί για να ανακαλύψει ομάδες ασθενών με παρόμοια χαρακτηριστικά ή μοτίβα στα δεδομένα τους, που μπορεί να υποδηλώνουν κοινούς παράγοντες κινδύνου ή τύπους διαβήτη. Αυτό μπορεί να βοηθήσει τους ειδικούς υγείας να αναγνωρίσουν υποομάδες ασθενών που ενδέχεται να χρειάζονται διαφοροποιημένες προσεγγίσεις στη διαχείριση ή τη θεραπεία του διαβήτη.

Επίσης, η μείωση διαστάσεων (dimensionality reduction) είναι μια άλλη σημαντική τεχνική στο unsupervised learning, η οποία μπορεί να βοηθήσει στην ανάλυση και την ερμηνεία μεγάλων και πολύπλοκων δεδομενοσυνόλων, όπως αυτά που συναντώνται στην ιατρική έρευνα για τον διαβήτη. Αλγόριθμοι όπως η ανάλυση κύριων συνιστωσών (PCA) και η t-distributed Stochastic Neighbor Embedding (t-SNE) επιτρέπουν την απλούστευση των δεδομένων, διατηρώντας ταυτόχρονα τις βασικές πληροφορίες, γεγονός που καθιστά ευκολότερη την επισκόπηση και την ανακάλυψη δομών ή μοτίβων.

Η χρήση τεχνικών unsupervised learning στην έρευνα για τον διαβήτη μπορεί να οδηγήσει στην ανακάλυψη νέων βιοδεικτών ή παραγόντων κινδύνου που δεν έχουν αναγνωριστεί μέχρι σήμερα. Αυτό είναι ιδιαίτερα σημαντικό στον διαβήτη, όπου η πρόληψη και η πρώιμη διάγνωση μπορούν να έχουν σημαντικό αντίκτυπο στη διαχείριση και τη θεραπεία της νόσου.

Παρά τις προκλήσεις που συνδέονται με την ερμηνεία των αποτελεσμάτων του unsupervised learning και την ανάγκη για μεγάλα και ποιοτικά δεδομένα, οι δυνατότητες που προσφέρει στην ιατρική έρευνα και την πρόβλεψη ασθενειών είναι αξιοσημείωτες. Η συνεχιζόμενη εξέλιξη στις τεχνολογίες Machine Learning και η αυξανόμενη διαθεσιμότητα ιατρικών δεδομένων ανοίγουν νέους δρόμους για τη βελτίωση της πρόληψης, της διάγνωσης και της θεραπείας του διαβήτη, με το unsupervised learning να παίζει κεντρικό ρόλο σε αυτή τη διαδικασία [8].

### 3.1.3 Reinforcement Learning

Το Reinforcement Learning (RL) είναι μια προηγμένη κατηγορία του Machine Learning που επικεντρώνεται στην εκμάθηση βέλτιστων στρατηγικών δράσης μέσω δοκιμών και αξιολόγησης των αποτελεσμάτων. Στην πραγματικότητα, τα μοντέλα RL "εκπαιδεύονται" μέσω ενός συνεχούς κύκλου από αποφάσεις και αντιδράσεις, ενισχύοντας συμπεριφορές που οδηγούν σε θετικές επιβραβεύσεις και αποφεύγοντας εκείνες που οδηγούν σε αρνητικές. Αυτή η διαδικασία μάθησης από την εμπειρία κάνει το RL ιδιαίτερα ελκυστικό για περίπλοκες και δυναμικά προβλήματα, όπως είναι η πρόβλεψη και η διαχείριση του διαβήτη [9].

Η εφαρμογή του Reinforcement Learning στην πρόβλεψη και διαχείριση του διαβήτη μπορεί να αποτελέσει μια πρωτοποριακή προσέγγιση, δεδομένου ότι η νόσος αυτή απαιτεί συνεχή παρακολούθηση και προσαρμογή των θεραπευτικών πρακτικών. Τα μοντέλα RL μπορούν να αξιοποιήσουν τεράστιες ποσότητες δεδομένων από διάφορες πηγές, όπως ιατρικές εξετάσεις, αναλύσεις αίματος, συνήθειες διατροφής και φυσικής δραστηριότητας, για να κατανοήσουν καλύτερα την αλληλεπίδραση μεταξύ διαφόρων παραγόντων και την επίδρασή τους στην εξέλιξη του διαβήτη.

Ένας τομέας όπου το RL μπορεί να αποδειχθεί ιδιαίτερα ωφέλιμο είναι η βελτιστοποίηση των σχεδίων θεραπείας. Μέσω της συνεχούς διαδικασίας δοκιμής και λάθους, ένα σύστημα RL μπορεί να "μάθει" να προσδιορίζει ποιες θεραπείες και συνδυασμοί θεραπειών είναι πιο αποτελεσματικοί για διάφορους τύπους ασθενών, λαμβάνοντας υπόψη τις μοναδικές τους βιολογικές και ζωτικές παραμέτρους.

Επιπλέον, το RL μπορεί να συμβάλει στην ανάπτυξη προσωποποιημένων προγραμμάτων παρακολούθησης και πρόληψης για τον διαβήτη, αξιοποιώντας την ικανότητά του να προσαρμόζεται σε συνεχώς αλλαγμένες συνθήκες και να βελτιστοποιεί τις αποφάσεις με βάση την ανατροφοδότηση. Αυτό σημαίνει ότι τα συστήματα βασισμένα στο RL μπορούν να παρέχουν πιο δυναμικές και ευέλικτες λύσεις που προσαρμόζονται στις μεταβαλλόμενες ανάγκες των ασθενών με διαβήτη, βελτιστοποιώντας την πρόληψη και την παρέμβαση.

Το RL μπορεί επίσης να προσφέρει σημαντικές δυνατότητες στην πρόβλεψη και αντιμετώπιση επιπλοκών του διαβήτη, όπως η διαβητική νευροπάθεια και η διαβητική ρετινοπάθεια. Μέσω της ανάλυσης των μοτίβων εξέλιξης της νόσου και της απόκρισης των ασθενών σε διάφορες θεραπείες, τα συστήματα RL μπορούν να αναγνωρίσουν τις πιο αποτελεσματικές στρατηγικές για την πρόληψη ή την καθυστέρηση της εμφάνισης των επιπλοκών.

Παρά τις προκλήσεις που συνεπάγεται η εφαρμογή του RL σε πραγματικά ιατρικά προβλήματα, όπως η ανάγκη για μεγάλες ποσότητες υψηλής ποιότητας δεδομένων και η πολυπλοκότητα της ανάπτυξης και της εκπαίδευσης των μοντέλων, το δυναμικό του στη βελτίωση της διαχείρισης και της πρόληψης του διαβήτη είναι τεράστιο. Με την κατάλληλη εφαρμογή και τη συνεχή τεχνολογική πρόοδο, το Reinforcement Learning έχει τη δυνατότητα να επανασχεδιάσει το μέλλον της ιατρικής περίθαλψης για τους ασθενείς με διαβήτη, προσφέροντας πιο προσαρμοσμένες και αποτελεσματικές θεραπείες.

### 3.1.4 Semi-supervised learning

Το semi-supervised learning αντιπροσωπεύει μια υβριδική προσέγγιση στο φάσμα των μεθοδολογιών μηχανικής μάθησης που συνδυάζει τα έξυπνα τα χαρακτηριστικά της εποπτευόμενης και της μη εποπτευόμενης μάθησης. Αυτή η μέθοδος αξιοποιεί ένα μικρό σύνολο δεδομένων με labels μαζί με έναν τεράστιο όγκο δεδομένων χωρίς labels για το training. Η ουσία του semi-supervised learning έγκειται στην ικανότητά της να ξεπερνά τους περιορισμούς που θέτει η σπανιότητα των δεδομένων με labels, η οποία είναι συχνά το σημείο συμφόρησης στην ανάπτυξη λύσεων machine learning λόγω του υψηλού κόστους και των εκτεταμένων απαιτήσεων χρόνου που σχετίζονται με τον σχολιασμό δεδομένων. Η βάση των μοντέλων machine learning βασίζεται στην ικανότητά τους να μαθαίνουν από δεδομένα.

Στην εποπτευόμενη μάθηση, τα μοντέλα εκπαιδεύονται σε ένα σύνολο δεδομένων όπου κάθε παράδειγμα γίνεται labelled με τη σωστή απάντηση, επιτρέποντας στο μοντέλο να μάθει τη σχέση μεταξύ των εισροών και των επιθυμητών εξόδων. Ωστόσο, η διαδικασία χειροκίνητης επισήμανσης δεδομένων μπορεί να είναι απαγορευτικά δαπανηρή και χρονοβόρα, ιδιαίτερα για εργασίες που απαιτούν ειδικές γνώσεις. Το semi-supervised learning, αντίθετα, δεν χρησιμοποιεί δεδομένα με labels, αλλά εστιάζει στον εντοπισμό προτύπων ή δομών εντός των δεδομένων. Ενώ η μάθηση χωρίς επίβλεψη μπορεί να αποκαλύψει κρυφές πληροφορίες μέσα σε σύνολα δεδομένων, στερείται της άμεσης εφαρμογής των εποπτευόμενων μεθόδων σε προγνωστικές εργασίες.

Το semi-supervised learning αναδεικνύεται ως στρατηγική λύση σε αυτές τις προκλήσεις απαιτώντας μόνο ένα ελάχιστο υποσύνολο επισημασμένων δεδομένων σε συνδυασμό με ένα μεγαλύτερο σύνολο δεδομένων χωρίς labels. Αυτή η προσέγγιση βασίζεται στην υπόθεση ότι τα δεδομένα χωρίς labels, όταν αξιοποιηθούν σωστά, μπορούν να βελτιώσουν σημαντικά τη διαδικασία μάθησης, παρέχοντας πρόσθετο πλαίσιο και διευκολύνοντας μια πιο ολοκληρωμένη κατανόηση της διανομής δεδομένων. Η μεθοδολογία πίσω από το semi-supervised περιλαμβάνει πολλές καινοτόμες τεχνικές

που έχουν σχεδιαστεί για την παρέκταση της γνώσης που αποκτάται από τα επισημασμένα δεδομένα στο μη επισημασμένο σύνολο δεδομένων.

Μια κοινή στρατηγική είναι η αυτοεκπαίδευση, όπου το μοντέλο που αρχικά εκπαιδεύτηκε στα επισημασμένα δεδομένα χρησιμοποιείται για την πρόβλεψη ετικετών για τα δεδομένα χωρίς labels. Στη συνέχεια, οι πιο σίγουρες προβλέψεις προστίθενται στο σύνολο εκπαίδευσης σαν να ήταν αληθινές ετικέτες και το μοντέλο επανεκπαιδεύεται με αυτό το επαυξημένο σύνολο δεδομένων. Μια άλλη προσέγγιση, η συνεκπαίδευση, περιλαμβάνει την εκπαίδευση δύο χωριστών μοντέλων σε διαφορετικές απόψεις των δεδομένων, χρησιμοποιώντας κάθε μοντέλο για την επισήμανση δεδομένων για το άλλο, εμπλουτίζοντας έτσι τη διαδικασία εκπαίδευσης.

Η χρησιμότητα του semi-supervised learning είναι εμφανής σε διάφορους τομείς όπου τα δεδομένα είναι άφθονα, αλλά οι εμφανίσεις με labels είναι σπάνιες ή δαπανηρές για την απόκτησή τους. Για παράδειγμα, στον τομέα της ιατρικής απεικόνισης, η επισήμανση απαιτεί από ειδικούς ακτινολόγους να σχολιάζουν σχολαστικά τις εικόνες, μια διαδικασία που είναι τόσο χρονοβόρα όσο και δαπανηρή. Το semi-supervised learning μπορεί να αξιοποιήσει ένα μικρό σύνολο ετικετών εικόνων για τη βελτίωση των διαγνωστικών μοντέλων, χρησιμοποιώντας τις τεράστιες ποσότητες διαθέσιμων ιατρικών εικόνων χωρίς labels. Ομοίως, στο natural language processing, τα semi-supervised ML methods επιτρέπουν την ανάπτυξη ισχυρών μοντέλων για εργασίες όπως η ανάλυση συναισθήματος ή η μετάφραση γλώσσας χωρίς την ανάγκη εκτεταμένων συνόλων δεδομένων με label. Τα πλεονεκτήματα του semi-supervised learning εκτείνονται πέρα από την απλή αποτελεσματική χρήση δεδομένων χωρίς labels. Μπορεί να οδηγήσει σε μοντέλα που είναι πιο γενικεύσιμα και ανθεκτικά σε αόρατα δεδομένα. Με την εκπαίδευση σε ένα σύνολο δεδομένων που περιλαμβάνει μια ευρεία αναπαράσταση του χώρου εισόδου (μέσω των δεδομένων χωρίς labels), τα μοντέλα μπορούν να συλλάβουν καλύτερα την υποκείμενη κατανομή δεδομένων, μειώνοντας τον κίνδυνο υπερβολικής προσαρμογής στα επισημασμένα δεδομένα και ενισχύοντας την ικανότητα του μοντέλου να αποδίδει καλά σε νέα δεδομένα [10] .

Παρά αυτά τα οφέλη, το semi-supervised learning αντιμετωπίζει και αρκετές προκλήσεις. Ένα πρωταρχικό μέλημα είναι η αξιοπιστία των pseudo-labels - οι ετικέτες που το μοντέλο αποδίδει στα δεδομένα χωρίς labels. Εάν αυτές οι ψευδο-ετικέτες είναι ανακριβείς, μπορεί να οδηγήσουν σε ενίσχυση σφαλμάτων κατά τη διαδικασία εκμάθησης, υποβαθμίζοντας ενδεχομένως την απόδοση του μοντέλου. Ως εκ τούτου, οι μηχανισμοί για την αξιολόγηση και τη διασφάλιση της ποιότητας των ψευδο-ετικέτες είναι κρίσιμα συστατικά των ημι-εποπτευόμενων συστημάτων μάθησης. Επιπλέον, η επιτυχία του semi-supervised learning εξαρτάται από την υπόθεση ότι τα επισημασμένα και τα μη επισημασμένα δεδομένα προέρχονται από την ίδια κατανομή. Σημαντικές διαφορές μεταξύ αυτών των κατανομών μπορεί να υπονομεύσουν την αποτελεσματικότητα του training, απαιτώντας προσεκτική επιλογή και προεπεξεργασία δεδομένων για να διασφαλιστεί η συνέπεια.

Συμπερασματικά, το semi-supervised learning προσφέρει ένα ισχυρό πλαίσιο για το data manipulation τόσο με labels όσο και unlabeled, παρουσιάζοντας μια οικονομικά αποδοτική λύση για την αντιμετώπιση των προκλήσεων που σχετίζονται με labelling

process. Με την ενσωμάτωση των πληροφοριών που προέρχονται από δεδομένα χωρίς labels, το semi-supervised learning μπορεί να βελτιώσει την απόδοση, τη γενίκευση και την ευρωστία του μοντέλου. Καθώς τα δεδομένα συνεχίζουν να αυξάνονται σε όγκο και πολυπλοκότητα, ο ρόλος του semi-supervised learning στην απελευθέρωση των δυνατοτήτων των μοντέλων machine learning γίνεται ολοένα και πιο απαραίτητος, προσφέροντας μια ρεαλιστική οδό για αξιοποίηση.

### 3.1.5 Self-supervised learning

Το self-supervised learning, ένα αναπτυσσόμενο υποσύνολο τεχνικών μάθησης χωρίς επίβλεψη, έχει αναδειχθεί ως μια ισχυρή μέθοδος που επιτρέπει στις μηχανές να κατανοούν και να ερμηνεύουν τον κόσμο γύρω τους με ελάχιστη ανθρώπινη παρέμβαση. Σε αντίθεση με την παραδοσιακή εποπτευόμενη μάθηση, η οποία βασίζεται σε μεγάλο βαθμό σε σύνολα δεδομένων με επισήμανση ανθρώπου, το self-supervised learning αξιοποιεί έξυπνα την εγγενή δομή των ίδιων των δεδομένων για να δημιουργήσει τα δικά της εποπτικά σήματα. Αυτή η προσέγγιση διευκολύνει τη δημιουργία βοηθητικών εργασιών, όπως η πρόβλεψη μέρους των δεδομένων από ένα άλλο μέρος, για να μάθουμε πλούσιες και ουσιαστικές αναπαραστάσεις των δεδομένων. Η ουσία του self-supervised learning έγκειται στην ικανότητά της να εκμεταλλεύεται το εγγενές πλαίσιο μέσα στα δεδομένα, ξεκλειδώνοντας μια πληθώρα εφαρμογών και δυνατοτήτων σε διάφορους τομείς, συμπεριλαμβανομένης της επεξεργασίας φυσικής γλώσσας, της όρασης υπολογιστή και όχι μόνο.

Στον πυρήνα της, το self-supervised learning επιδιώκει να μοντελοποιήσει την υποκείμενη κατανομή των δεδομένων μαθαίνοντας να προβλέπει οποιοδήποτε μέρος των δεδομένων από οποιοδήποτε άλλο μέρος. Αυτό επιτυγχάνεται με το σχεδιασμό εργασιών προσχήματος—βοηθητικούς στόχους που το μοντέλο στοχεύει να λύσει, μαθαίνοντας έτσι πολύτιμα χαρακτηριστικά και αναπαραστάσεις στη διαδικασία. Αυτές οι εργασίες δεν απαιτούν εξωτερικές ετικέτες, αλλά παράγουν ετικέτες από τα ίδια τα δεδομένα. Για παράδειγμα, στο πλαίσιο της επεξεργασίας φυσικής γλώσσας, μια κοινή εργασία προσχήματος μπορεί να περιλαμβάνει την πρόβλεψη της επόμενης λέξης σε μια πρόταση που δίνονται στις προηγούμενες λέξεις ή τη συμπλήρωση μιας λέξης που λείπει δεδομένου του περιβάλλοντός της. Ομοίως, στην όραση υπολογιστή, ένα μοντέλο μπορεί να εκπαιδευτεί για να προβλέψει ένα κομμάτι που λείπει μιας εικόνας ή να ανακατασκευάσει μια εικόνα από μια κατεστραμμένη έκδοση. Μέσω αυτών των εργασιών, τα μοντέλα μαθαίνουν να αποτυπώνουν τα βασικά χαρακτηριστικά και δομές μέσα στα δεδομένα, οδηγώντας σε μια βαθύτερη κατανόηση των υποκείμενων προτύπων τους [11].

Ένα από τα βασικά πλεονεκτήματα του self-supervised learning είναι η ευελιξία και η αποτελεσματικότητά της στην αξιοποίηση μεγάλου όγκου δεδομένων χωρίς labels. Ο ψηφιακός κόσμος είναι γεμάτος με τεράστιες ποσότητες δεδομένων, τα περισσότερα από τα οποία είναι χωρίς labels. Η μη αυτόματη επισήμανση αυτών των δεδομένων για εποπτευόμενη μάθηση δεν είναι μόνο μη πρακτική αλλά και απαγορευτικά δαπανηρή και χρονοβόρα. Το self-supervised learning παρακάμπτει αυτό το σημείο συμφόρησης δημιουργώντας τις δικές της ετικέτες, εκμεταλλευόμενο έτσι τα δεδομένα χωρίς labels με

οικονομικά αποδοτικό τρόπο. Αυτή η δυνατότητα επιτρέπει την ανάπτυξη πιο ισχυρών και γενικεύσιμων μοντέλων που μπορούν να μάθουν από ένα ευρύτερο φάσμα δεδομένων χωρίς τους περιορισμούς των συνόλων δεδομένων με μη αυτόματο σχολιασμό.

Επιπλέον, το self-supervised learning έχει τη δυνατότητα να μάθει πιο γενικευμένες και μεταβιβάσιμες αναπαραστάσεις. Δεδομένου ότι τα μαθησιακά χαρακτηριστικά προέρχονται από τα ίδια τα δεδομένα, χωρίς να βασίζονται σε συγκεκριμένες ετικέτες που δημιουργούνται από τον άνθρωπο, οι αναπαραστάσεις τείνουν να καταγράφουν θεμελιώδεις πτυχές των δεδομένων που ισχύουν σε ένα ευρύ φάσμα εργασιών. Αυτή η ιδιότητα καθιστά το multi-instance learning ιδιαίτερα ελκυστική για τη μεταφορά μάθησης, όπου ένα μοντέλο που εκπαιδεύεται σε μια εργασία προσαρμόζεται για μια άλλη σχετική εργασία. Οι αναπαραστάσεις που μαθαίνονται μέσω multi-instance μεθόδων μπορούν να χρησιμεύσουν ως ένα πλούσιο, μεταβιβάσιμο θεμέλιο, βελτιώνοντας σημαντικά την απόδοση σε εργασίες με ελάχιστη λεπτομέρεια και δεδομένα με labels.

Παρά τα πλεονεκτήματά της, το self-supervised learning παρουσιάζει το δικό της σύνολο προκλήσεων και προβληματισμών. Ο σχεδιασμός αποτελεσματικών εργασιών προσχήματος είναι ζωτικής σημασίας, καθώς η ποιότητα και η συνάφεια αυτών των εργασιών επηρεάζουν άμεσα την ικανότητα του μοντέλου να μαθαίνει ουσιαστικές αναπαραστάσεις. Οι εργασίες πρέπει να είναι αρκετά απαιτητικές για να ενθαρρύνουν το μοντέλο να συλλάβει σημαντικά χαρακτηριστικά των δεδομένων, αλλά να είναι ακόμη επιλύσιμα με βάση τις παρούσες πληροφορίες. Αυτή η λεπτή ισορροπία απαιτεί προσεκτική εξέταση και γνώσεις σε συγκεκριμένο τομέα για τον εντοπισμό εργασιών που οδηγούν σε χρήσιμα μαθησιακά αποτελέσματα.

Επιπλέον, η αξιολόγηση της απόδοσης των multi-instance μοντέλων μπορεί να είναι πιο περίπλοκη από ό,τι σε περιβάλλοντα εποπτευόμενης μάθησης. Δεδομένου ότι οι κύριοι στόχοι μάθησης δεν αντιστοιχούν άμεσα σε συγκεκριμένες εργασίες του πραγματικού κόσμου, η αξιολόγηση της ποιότητας των μαθησιακών αναπαραστάσεων συχνά περιλαμβάνει την εφαρμογή του μοντέλου σε εργασίες και τη μέτρηση της απόδοσής του. Αυτή η έμμεση μέθοδος αξιολόγησης καθιστά δυσκολότερη την άμεση σύγκριση της αποτελεσματικότητας διαφορετικών προσεγγίσεων αυτοεποπτεύουσας μάθησης και την κατανόηση των ειδικών χαρακτηριστικών των δεδομένων που έχει συλλάβει το μοντέλο.

Το self-supervised learning διασταυρώνεται επίσης με άλλα παραδείγματα machine Learning και συμπληρώνει. Για παράδειγμα, μπορεί να συνδυαστεί με supervised learning για την προετοιμασία των βαρών μοντέλων πριν από τη λεπτομερή ρύθμιση σε ένα επισημασμένο σύνολο δεδομένων για μια συγκεκριμένη εργασία. Αυτή η υβριδική προσέγγιση μπορεί να οδηγήσει σε σημαντικές βελτιώσεις στην απόδοση, ειδικά σε σενάρια όπου τα δεδομένα με labels είναι σπάνια. Επιπλέον, το self-supervised learning μπορεί να ενισχύσει το reinforcement learning παρέχοντας έναν μηχανισμό στους πράκτορες να μάθουν χρήσιμες αναπαραστάσεις του περιβάλλοντός τους, βελτιώνοντας έτσι τις ικανότητές τους στη λήψη αποφάσεων.

Το self-supervised learning αντιπροσωπεύει μια αλλαγή παραδείγματος στον τομέα της

μηχανικής μάθησης, προσφέροντας μια νέα προσέγγιση για την αξιοποίηση των τεράστιων ποσοτήτων unlabeled data τα που είναι διαθέσιμα σήμερα. Παράγοντας τα δικά του σήματα εποπτείας από τα ίδια τα δεδομένα, επιτρέπει την ανάπτυξη μοντέλων που μπορούν να μάθουν πλούσιες και μεταβιβάσιμες αναπαραστάσεις, ανοίγοντας το δρόμο για προόδους σε πολλούς τομείς. Καθώς η έρευνα σε αυτόν τον τομέα συνεχίζει να εξελίσσεται, το self-supervised learning μπορεί να παίξει καθοριστικό ρόλο στο ξεκλείδωμα νέων συνόρων στην τεχνητή νοημοσύνη, προσφέροντας μια ματιά σε ένα μέλλον όπου οι μηχανές θα μπορούν να μαθαίνουν πιο αυτόνομα.

### 3.1.6 Multi-Instance Learning

Το multi-instance learning, ένα αναπτυσσόμενο υποσύνολο τεχνικών μάθησης χωρίς επίβλεψη, έχει αναδειχθεί ως μια ισχυρή μέθοδος που επιτρέπει στις μηχανές να κατανοούν και να ερμηνεύουν τον κόσμο γύρω τους με ελάχιστη ανθρώπινη παρέμβαση. Σε αντίθεση με την παραδοσιακή εποπτευόμενη μάθηση, η οποία βασίζεται σε μεγάλο βαθμό σε σύνολα δεδομένων με επισήμανση ανθρώπου, το multi-instance learning αξιοποιεί έξυπνα την εγγενή δομή των ίδιων των δεδομένων για να δημιουργήσει τα δικά της εποπτικά σήματα. Αυτή η προσέγγιση διευκολύνει τη δημιουργία βοηθητικών εργασιών, όπως η πρόβλεψη μέρους των δεδομένων από ένα άλλο μέρος, για να μάθουμε πλούσιες και ουσιαστικές αναπαραστάσεις των δεδομένων. Η ουσία του multi-instance learning έγκειται στην ικανότητά της να εκμεταλλεύεται το εγγενές πλαίσιο μέσα στα δεδομένα, ξεκλειδώνοντας μια πληθώρα εφαρμογών και δυνατοτήτων σε διάφορους τομείς, συμπεριλαμβανομένης της επεξεργασίας φυσικής γλώσσας, της όρασης υπολογιστή και όχι μόνο [12].

Στον πυρήνα της, το multi-instance learning επιδιώκει να μοντελοποιήσει την υποκείμενη κατανομή των δεδομένων μαθαίνοντας να προβλέπει οποιοδήποτε μέρος των δεδομένων από οποιοδήποτε άλλο μέρος. Αυτό επιτυγχάνεται με το σχεδιασμό εργασιών προσχήματος—βοηθητικούς στόχους που το μοντέλο στοχεύει να λύσει, μαθαίνοντας έτσι πολύτιμα χαρακτηριστικά και αναπαραστάσεις στη διαδικασία. Αυτές οι εργασίες δεν απαιτούν εξωτερικές ετικέτες, αλλά παράγουν ετικέτες από τα ίδια τα δεδομένα. Για παράδειγμα, στο πλαίσιο της επεξεργασίας φυσικής γλώσσας, μια κοινή εργασία προσχήματος μπορεί να περιλαμβάνει την πρόβλεψη της επόμενης λέξης σε μια πρόταση που δίνονται στις προηγούμενες λέξεις ή τη συμπλήρωση μιας λέξης που λείπει δεδομένου του περιβάλλοντός της. Ομοίως, στην όραση υπολογιστή, ένα μοντέλο μπορεί να εκπαιδευτεί για να προβλέψει ένα κομμάτι που λείπει μιας εικόνας ή να ανακατασκευάσει μια εικόνα από μια κατεστραμμένη έκδοση. Μέσω αυτών των εργασιών, τα μοντέλα μαθαίνουν να αποτυπώνουν τα βασικά χαρακτηριστικά και δομές μέσα στα δεδομένα, οδηγώντας σε μια βαθύτερη κατανόηση των υποκείμενων προτύπων τους.

Ένα από τα βασικά πλεονεκτήματα του multi-instance learning είναι η ευελιξία και η αποτελεσματικότητά της στην αξιοποίηση μεγάλου όγκου δεδομένων χωρίς ετικέτα. Ο ψηφιακός κόσμος είναι γεμάτος με τεράστιες ποσότητες δεδομένων, τα περισσότερα από τα οποία είναι χωρίς ετικέτα. Η μη αυτόματη επισήμανση αυτών των δεδομένων για εποπτευόμενη μάθηση δεν είναι μόνο μη πρακτική αλλά και απαγορευτικά δαπανηρή και

χρονοβόρα. Το multi-instance learning παρακάμπτει αυτό το σημείο συμφόρησης δημιουργώντας τις δικές της ετικέτες, εκμεταλλευόμενο έτσι τα δεδομένα χωρίς ετικέτα με οικονομικά αποδοτικό τρόπο. Αυτή η δυνατότητα επιτρέπει την ανάπτυξη πιο ισχυρών και γενικεύσιμων μοντέλων που μπορούν να μάθουν από ένα ευρύτερο φάσμα δεδομένων χωρίς τους περιορισμούς των συνόλων δεδομένων με μη αυτόματο σχολιασμό.

Επιπλέον, το multi-instance learning έχει τη δυνατότητα να μάθει πιο γενικευμένες και μεταβιβάσιμες αναπαραστάσεις. Δεδομένου ότι τα μαθησιακά χαρακτηριστικά προέρχονται από τα ίδια τα δεδομένα, χωρίς να βασίζονται σε συγκεκριμένες ετικέτες που δημιουργούνται από τον άνθρωπο, οι αναπαραστάσεις τείνουν να καταγράφουν θεμελιώδεις πτυχές των δεδομένων που ισχύουν σε ένα ευρύ φάσμα εργασιών. Αυτή η ιδιότητα καθιστά το multi-instance learning ιδιαίτερα ελκυστική για τη μεταφορά μάθησης, όπου ένα μοντέλο που εκπαιδεύεται σε μια εργασία προσαρμόζεται για μια άλλη σχετική εργασία. Οι αναπαραστάσεις που μαθαίνονται μέσω multi-instance μεθόδων μπορούν να χρησιμεύσουν ως ένα πλούσιο, μεταβιβάσιμο θεμέλιο, βελτιώνοντας σημαντικά την απόδοση σε εργασίες με ελάχιστη λεπτομέρεια και δεδομένα με ετικέτα.

Παρά τα πλεονεκτήματά της, το multi-instance learning παρουσιάζει το δικό της σύνολο προκλήσεων και προβληματισμών. Ο σχεδιασμός αποτελεσματικών εργασιών προσχήματος είναι ζωτικής σημασίας, καθώς η ποιότητα και η συνάφεια αυτών των εργασιών επηρεάζουν άμεσα την ικανότητα του μοντέλου να μαθαίνει ουσιαστικές αναπαραστάσεις. Οι εργασίες πρέπει να είναι αρκετά απαιτητικές για να ενθαρρύνουν το μοντέλο να συλλάβει σημαντικά χαρακτηριστικά των δεδομένων, αλλά να είναι ακόμη επιλύσιμα με βάση τις παρούσες πληροφορίες. Αυτή η λεπτή ισορροπία απαιτεί προσεκτική εξέταση και γνώσεις σε συγκεκριμένο τομέα για τον εντοπισμό εργασιών που οδηγούν σε χρήσιμα μαθησιακά αποτελέσματα.

Επιπλέον, η αξιολόγηση της απόδοσης των multi-instance μοντέλων μπορεί να είναι πιο περίπλοκη από ό,τι σε περιβάλλοντα εποπτευόμενης μάθησης. Δεδομένου ότι οι κύριοι στόχοι μάθησης δεν αντιστοιχούν άμεσα σε συγκεκριμένες εργασίες του πραγματικού κόσμου, η αξιολόγηση της ποιότητας των μαθησιακών αναπαραστάσεων συχνά περιλαμβάνει την εφαρμογή του μοντέλου σε εργασίες και τη μέτρηση της απόδοσής του. Αυτή η έμμεση μέθοδος αξιολόγησης καθιστά δυσκολότερη την άμεση σύγκριση της αποτελεσματικότητας διαφορετικών προσεγγίσεων αυτοεποπτεύουσας μάθησης και την κατανόηση των ειδικών χαρακτηριστικών των δεδομένων που έχει συλλάβει το μοντέλο.

Το multi-instance learning διασταυρώνεται επίσης με άλλα παραδείγματα μηχανικής μάθησης και συμπληρώνει. Για παράδειγμα, μπορεί να συνδυαστεί με εποπτευόμενη μάθηση χρησιμοποιώντας αυτο-εποπτευόμενη προκατάρτιση για την προετοιμασία των βαρών μοντέλων πριν από τη λεπτομερή ρύθμιση σε ένα επισημασμένο σύνολο δεδομένων για μια συγκεκριμένη εργασία. Αυτή η υβριδική προσέγγιση μπορεί να οδηγήσει σε σημαντικές βελτιώσεις στην απόδοση, ειδικά σε σενάρια όπου τα δεδομένα με ετικέτα είναι σπάνια. Επιπλέον, το multi-instance learning μπορεί να ενισχύσει το reinforcement learning παρέχοντας έναν μηχανισμό στους πράκτορες να μάθουν χρήσιμες αναπαραστάσεις του περιβάλλοντός τους, βελτιώνοντας έτσι τις ικανότητές τους

στη λήψη αποφάσεων.

Εν κατακλείδι, το multi-instance learning προσφέρει ένα ισχυρό πλαίσιο για την αντιμετώπιση supervised learning operations όπου το labelling είναι ανέφικτο ή μη πρακτικό. Συσχετίζοντας ετικέτες με ομάδες παρουσιών, το MIL επιτρέπει την εξαγωγή σημαντικών μοτίβων και τη δημιουργία προβλέψεων σε περιβάλλοντα όπου είναι διαθέσιμοι μόνο coarse-grained annotations. Καθώς τα δεδομένα συνεχίζουν να αυξάνονται τόσο σε όγκο όσο και σε πολυπλοκότητα, η ευελιξία και η προσαρμοστικότητα της εκμάθησης πολλαπλών περιπτώσεων τα καθιστούν ένα ανεκτίμητο εργαλείο στο οπλοστάσιο της μηχανικής εκμάθησης, με πιθανές εφαρμογές σε ένα ευρύ φάσμα τομέων, από την υγειονομική περίθαλψη έως την επεξεργασία φυσικής γλώσσας.

### 3.2 Few shot learning

Η μάθηση με λίγες λήψεις αποτελεί ένα αξιοσημείωτο παράδειγμα στη σφαίρα της μηχανικής μάθησης, ειδικά σχεδιασμένο για να αντιμετωπίσει την πρόκληση της μάθησης από έναν ελάχιστο όγκο δεδομένων. Αυτή η προσέγγιση είναι ολοένα και πιο κρίσιμη σε έναν κόσμο όπου τα δεδομένα είναι άφθονα, ωστόσο η ικανότητα επισήμανσης και αποτελεσματικής χρήσης αυτών των δεδομένων μπορεί να είναι τόσο απαιτητική όσο και με ένταση πόρων. Η μάθηση με λίγες λήψεις στοχεύει στη δημιουργία μοντέλων ικανών να γενικεύουν καλά από πολύ λίγα παραδείγματα, συχνά μόλις μία ή λίγες περιπτώσεις ανά τάξη. Αυτό το δοκίμιο διερευνά την έννοια της μάθησης με λίγες λήψεις, τη σημασία της, τις μεθοδολογίες, τις εφαρμογές, τις προκλήσεις και τη μελλοντική κατεύθυνση αυτού του συναρπαστικού τομέα.

Τα παραδοσιακά μοντέλα μηχανικής μάθησης ευδοκιμούν σε μεγάλα σύνολα δεδομένων. Όσο περισσότερα δεδομένα εκπαιδεύονται αυτά τα μοντέλα, τόσο καλύτερη είναι η απόδοση τους. Αυτή η φύση που απαιτεί δεδομένα δημιουργεί πρόβλημα σε σενάρια όπου η συλλογή ή η επισήμανση τεράστιων ποσοτήτων δεδομένων είναι ανέφικτη, δαπανηρή ή αδύνατη. Το Few-shot Learning αντιμετωπίζει αυτό το ζήτημα αναπτύσσοντας αλγόριθμους που μπορούν να μάθουν αποτελεσματικά από περιορισμένα δεδομένα. Οι «βολές» αναφέρονται στον αριθμό των παραδειγμάτων ανά τάξη στα οποία έχει πρόσβαση το μοντέλο κατά τη διάρκεια της εκπαίδευσης. Ανάλογα με τη διαθεσιμότητα, αυτό μπορεί να κατηγοριοποιηθεί σε μάθηση μίας λήψης, όπου το μοντέλο μαθαίνει από ένα μεμονωμένο παράδειγμα ανά τάξη, ή σε μάθηση με λίγες λήψεις, όπου μαθαίνει από έναν μικρό αριθμό παραδειγμάτων.

Η σημασία της μάθησης με λίγες λήψεις έγκειται στην ικανότητά της να μιμείται την ανθρώπινη ικανότητα μάθησης. Οι άνθρωποι μπορούν συχνά να μάθουν από πολύ λίγα παραδείγματα λόγω της προηγούμενης γνώσης και της ικανότητάς μας να γενικεύουμε. Με στόχο την αναπαραγωγή αυτής της αποτελεσματικότητας, τα μοντέλα μάθησης λίγων βολών μπορούν να εφαρμοστούν σε πολλά σενάρια όπου τα δεδομένα είναι σπάνια ή ακριβά στη λήψη τους, όπως στη διάγνωση σπάνιων ασθενειών, στις ρομποτικές αλληλεπιδράσεις ή στην αναγνώριση νέων προϊόντων **[13]**.

- **Methodologies in Few-shot Learning:** Έχουν αναπτυχθεί διάφορες

μεθοδολογίες για την προσέγγιση της μάθησης με λίγα πλάνο, η καθεμία με τη μοναδική της στρατηγική για να ξεπεραστεί η πρόκληση της σπανιότητας δεδομένων:

- **Metric learning**: Οι προσεγγίσεις μετρικής εκμάθησης, όπως τα Σιαμέζικα δίκτυα ή τα δίκτυα αντιστοίχισης, στοχεύουν στην εκμάθηση μιας συνάρτησης ομοιότητας που μπορεί να συγκρίνει και να μετρήσει την απόσταση μεταξύ των παρουσιών σε έναν εκμαθημένο χώρο ενσωμάτωσης. Η ιδέα είναι ότι τα στιγμιότυπα της ίδιας κλάσης θα συγκεντρώνονται στενά μεταξύ τους, ενώ τα στιγμιότυπα από διαφορετικές κλάσεις θα απέχουν πολύ μεταξύ τους. Κατά τη διάρκεια της εξαγωγής συμπερασμάτων, το μοντέλο προβλέπει την κλάση ενός στιγμιότυπου συγκρίνοντας την απόστασή του με παραδείγματα λίγων λήψεων.

- **Model-Agnostic Meta-Learning** (MAML): Το MAML και παρόμοιες προσεγγίσεις μετα-μάθησης επικεντρώνονται στην εκμάθηση μιας προετοιμασίας μοντέλου που μπορεί να προσαρμοστεί γρήγορα σε νέες εργασίες με ελάχιστα δεδομένα. Το μοντέλο εκπαιδεύεται σε μια ποικιλία εργασιών, μαθαίνοντας να ενημερώνει αποτελεσματικά τις παραμέτρους του με λίγα μόνο βήματα κλίσης όταν αντιμετωπίζει μια νέα εργασία.
- **Transfer learning**: Αν και δεν είναι αποκλειστική για τη μάθηση με λίγες λήψεις, η μάθηση με μεταφορά μπορεί να είναι ιδιαίτερα αποτελεσματική με την προεκπαίδευση ενός μοντέλου σε ένα μεγάλο σύνολο δεδομένων και, στη συνέχεια, τη λεπτομέρεια σε ένα πολύ μικρότερο σύνολο δεδομένων που σχετίζεται με τη συγκεκριμένη εργασία λίγων βολών. Αυτό αξιοποιεί τα γενικά χαρακτηριστικά που αποκτήθηκαν κατά την προ-προπόνηση για να προσαρμοστούν στις συγκεκριμένες απαιτήσεις της νέας εργασίας.

### 3.2.1 Εφαρμογές Few-shot Learning

Η μάθηση Few-shot έχει βρει εφαρμογές σε διάφορους τομείς, αποδεικνύοντας την ευελιξία και την αποτελεσματικότητά της:

- Στην Υγειονομική περίθαλψη: Στην ιατρική απεικόνιση, η μάθηση με λίγες λήψεις βοηθά στη διάγνωση σπάνιων καταστάσεων όπου μόνο λίγα παραδείγματα μπορεί να είναι διαθέσιμα για εκπαίδευση.
- Στην Ρομποτική: Τα ρομπότ εξοπλισμένα με αλγόριθμους εκμάθησης λίγων βολών μπορούν να προσαρμοστούν γρήγορα σε νέες εργασίες, όπως η αναγνώριση νέων αντικειμένων ή η εκτέλεση νέων ενεργειών με ελάχιστα παραδείγματα.
- Στην Επεξεργασία Φυσικής Γλώσσας (NLP): Η εκμάθηση με λίγες λήψεις επιτρέπει στα μοντέλα να κατανοούν και να δημιουργούν κείμενο σε γλώσσες ή διαλέκτους με περιορισμένα διαθέσιμα δεδομένα.

### 3.2.2 Προκλήσεις στο Few-shot Learning

Παρά τις δυνατότητές της, η μάθηση με λίγες λήψεις αντιμετωπίζει πολλές προκλήσεις:
Ποιότητα δεδομένων: Τα περιορισμένα δεδομένα πρέπει να είναι άκρως αντιπροσωπευτικά και καλής ποιότητας, καθώς το μοντέλο έχει λιγότερα παραδείγματα

για να μάθει.
Overfitting: Με τόσο μικρά σύνολα δεδομένων, υπάρχει υψηλός κίνδυνος overfitting. Τα μοντέλα μπορεί να απομνημονεύουν τα λίγα παραδείγματα αντί να μαθαίνουν γενικεύσιμα μοτίβα.
Αξιολόγηση και συγκριτική αξιολόγηση: Η τυποποίηση της αξιολόγησης των μοντέλων μάθησης με λίγες λήψεις είναι πρόκληση, καθώς ο ορισμός των «λίγων» μπορεί να ποικίλλει και οι εργασίες μπορεί να είναι εξαιρετικά εξειδικευμένες.

Το μέλλον της ελάχιστης μάθησης έγκειται στην ενίσχυση της ικανότητας των μοντέλων να γενικεύουν από περιορισμένα δεδομένα και στην εξερεύνηση νέων αρχιτεκτονικών και μεθοδολογιών που μπορούν να μειώσουν περαιτέρω την εξάρτηση από μεγάλα σύνολα δεδομένων. Η ενσωμάτωση πιο προηγμένων μορφών προγενέστερης γνώσης, η εξερεύνηση της μη εποπτευόμενης και ημι-εποπτευόμενης μάθησης και η ανάπτυξη πιο ισχυρών σημείων αναφοράς αξιολόγησης αποτελούν βασικούς τομείς της συνεχιζόμενης και μελλοντικής έρευνας.
Η μάθηση με λίγες λήψεις αντιπροσωπεύει ένα σημαντικό βήμα προς τη δημιουργία πιο αποτελεσματικών και προσαρμόσιμων μοντέλων μηχανικής εκμάθησης που μπορούν να μάθουν από περιορισμένα δεδομένα. Εστιάζοντας σε μεθοδολογίες που επιτρέπουν γρήγορη μάθηση και γενίκευση, η μάθηση με λίγες λήψεις ανοίγει νέες δυνατότητες σε τομείς όπου τα δεδομένα είναι σπάνια ή η συλλογή περισσότερων δεδομένων δεν είναι πρακτική. Καθώς η έρευνα προχωρά, η ικανότητα των μοντέλων να μαθαίνουν από μερικά παραδείγματα όχι μόνο θα γίνει πιο περίπλοκη αλλά και πιο ενσωματωμένη στην επίλυση προβλημάτων του πραγματικού κόσμου σε διάφορους τομείς.

### 3.3 Zero shots learning

Το Zero-shot learning αντιπροσωπεύει μια μετασχηματιστική προσέγγιση στο τοπίο της μηχανικής μάθησης, με στόχο την υπέρβαση μιας από τις θεμελιώδεις προκλήσεις που αντιμετωπίζουν τα παραδοσιακά μοντέλα: την ικανότητα αναγνώρισης και ταξινόμησης περιπτώσεων τάξεων που δεν υπήρχαν κατά τη φάση της εκπαίδευσης. Αυτό το δοκίμιο εμβαθύνει στην έννοια της μηδενικής μάθησης, τις βασικές αρχές, τις μεθοδολογίες, τη σημασία, τις εφαρμογές, τις προκλήσεις και τις μελλοντικές κατευθύνσεις, αποκαλύπτοντας πώς ανοίγει το δρόμο για πιο προσαρμοστικά και έξυπνα συστήματα ικανά να αντιμετωπίσουν τη δυναμική φύση του πραγματικού κόσμος [14].

Η έννοια και σημασία του Zero-shot learning είναι η εξής: Στον πυρήνα της, το Zero-shot learning επιδιώκει να επιτρέψει στα μοντέλα να κάνουν ακριβείς προβλέψεις για τάξεις που δεν έχουν ξαναδεί. Αυτό έρχεται σε πλήρη αντίθεση με τα συμβατικά παραδείγματα μηχανικής μάθησης, τα οποία συνήθως απαιτούν έκθεση σε παραδείγματα όλων των τάξεων κατά τη διάρκεια της εκπαιδευτικής διαδικασίας για να επιτευχθεί αξιόπιστη ταξινόμηση. Η σημασία της μηδενικής μάθησης έγκειται στη δυνατότητά της να μειώσει δραστικά την εξάρτηση από μεγάλα, επισημασμένα σύνολα δεδομένων για κάθε κατηγορία ενδιαφέροντος, αντιμετωπίζοντας έτσι τα ζητήματα επεκτασιμότητας που σχετίζονται με την απόκτηση ολοκληρωμένων σχολιασμένων δεδομένων για κάθε πιθανή κατηγορία.

Η μηδενική μάθηση βασίζεται στη μεταφορά γνώσης από τάξεις που φαίνονται σε αόρατες. Βασίζεται στην υπόθεση ότι ενώ οι συγκεκριμένες περιπτώσεις ορισμένων κλάσεων μπορεί να μην υπάρχουν στα δεδομένα εκπαίδευσης, υπάρχει μια κοινή αναπαράσταση υψηλότερου επιπέδου ή χώρος χαρακτηριστικών σε όλες τις κλάσεις. Με την εκμάθηση αυτών των κοινών χαρακτηριστικών κατά τη διάρκεια της εκπαίδευσης, ένα μοντέλο μπορεί να αξιοποιήσει αυτή τη γνώση για να βγάλει συμπεράσματα σχετικά με μη ορατές κλάσεις με βάση τη σχέση τους με τα γνωστά χαρακτηριστικά.

Οι μεθοδολογίες που χρησιμοποιούνται στη μηδενική μάθηση έχουν σχεδιαστεί για να εκμεταλλεύονται τις σχέσεις μεταξύ των κλάσεων και των ιδιοτήτων τους. Αυτά μπορούν γενικά να κατηγοριοποιηθούν στις ακόλουθες προσεγγίσεις:

- Μέθοδοι που βασίζονται σε χαρακτηριστικά: Αυτές οι μέθοδοι βασίζονται στον καθορισμό κλάσεων μέσω ενός συνόλου υψηλού επιπέδου, περιγραφικών χαρακτηριστικών. Κατά τη διάρκεια της εκπαίδευσης, το μοντέλο μαθαίνει τη συσχέτιση μεταξύ των χαρακτηριστικών και των κλάσεων που βλέπει. Για εξαγωγή συμπερασμάτων σε μη ορατές κλάσεις, χρησιμοποιεί τα γνωστά χαρακτηριστικά αυτών των κλάσεων για να κάνει προβλέψεις.
- Μέθοδοι σημασιολογικής ενσωμάτωσης: Οι τεχνικές σημασιολογικής ενσωμάτωσης αντιστοιχίζουν τόσο τις ορατές όσο και τις μη ορατές τάξεις σε έναν κοινό σημασιολογικό χώρο, συχνά αξιοποιώντας εξωτερικές πηγές γνώσης, όπως σώματα κειμένου ή οντολογίες. Το μοντέλο μαθαίνει να προβάλλει χαρακτηριστικά εισόδου σε αυτόν τον χώρο, όπου η ταξινόμηση πραγματοποιείται με βάση την εγγύτητα στις αναπαραστάσεις της τάξης.
- Υβριδικές μέθοδοι: Οι υβριδικές προσεγγίσεις συνδυάζουν χαρακτηριστικά, σημασιολογικές ενσωματώσεις και μερικές φορές πρόσθετες πηγές πληροφοριών για να γεφυρώσουν πιο αποτελεσματικά το χάσμα μεταξύ ορατών και μη ορατών κλάσεων.

### 3.3.1 Εφαρμογές Zero-shot Learning

Οι επιπτώσεις της μηδενικής μάθησης είναι βαθιές σε διάφορους τομείς, ειδικά σε τομείς όπου η ποικιλομορφία των τάξεων είναι τεράστια και η διαθεσιμότητα των δεδομένων με ετικέτα είναι περιορισμένη. Η εκμάθηση μηδενικής λήψης είναι ζωτικής σημασίας στις εργασίες ταξινόμησης εικόνων και αναγνώρισης αντικειμένων, όπου μπορεί να αναγνωρίσει αντικείμενα που δεν φάνηκαν ποτέ κατά τη διάρκεια της εκπαίδευσης, μια κρίσιμη ικανότητα για αυτόνομα οχήματα και συστήματα επιτήρησης. Επιπλέον, η μηδενική εκμάθηση διευκολύνει τα γλωσσικά μοντέλα να εκτελούν εργασίες σε δεδομένα σε γλώσσες ή τομείς για τους οποίους δεν έχουν εκπαιδευτεί ρητά, ενισχύοντας πολυγλωσσικές εφαρμογές και εφαρμογές μεταξύ τομέων. Τέλος, για την αναγνώριση των ειδών, η μάθηση με μηδενική λήψη επιτρέπει την αναγνώριση σπάνιων ή απειλούμενων ειδών στις προσπάθειες παρακολούθησης της άγριας ζωής, όπου η απόκτηση ολοκληρωμένων συνόλων δεδομένων με ετικέτα είναι δύσκολη.

### 3.3.2 Προκλήσεις στο Zero-shot learning

Παρά τις δυνατότητές της, το Zero-shot learning παρουσιάζει προκλήσεις, οι οποίες αναφέροντα παρακάτω:

- Επιλογή και αναπαράσταση χαρακτηριστικών: Η επιτυχία των μεθόδων που βασίζονται σε χαρακτηριστικά βασίζεται σε μεγάλο βαθμό στην επιλογή και την αναπαράσταση χαρακτηριστικών, τα οποία πρέπει να είναι διακριτικά και περιγραφικά για όλες τις κλάσεις.
- Σημασιολογικό χάσμα: Η ασυμφωνία μεταξύ των χαρακτηριστικών χαμηλού επιπέδου που αποκτήθηκαν από τα δεδομένα και των σημασιολογικών πληροφοριών υψηλού επιπέδου ή των πληροφοριών χαρακτηριστικών αποτελεί σημαντική πρόκληση, που συχνά οδηγεί σε ένα σημασιολογικό χάσμα που μπορεί να επηρεάσει την απόδοση.
- Bias προς τις προβλεπόμενες κλάσεις: Τα μοντέλα τείνουν να είναι biased προς τις ορατές κλάσεις, γεγονός που καθιστά δύσκολη την επίτευξη υψηλής ακρίβειας για μη ορατές κλάσεις, ειδικά όταν η κατανομή των κλάσεων που εμφανίζονται και που δεν εμφανίζονται είναι σημαντικά διαφορετική.

Το μέλλον του zero-shot learning προσανατολίζεται στην αντιμετώπιση των σημερινών περιορισμών της και στην εξερεύνηση νέων μεθοδολογιών που μπορούν να ενισχύσουν περαιτέρω την εφαρμογή και την αποτελεσματικότητά της:

- Βελτιωμένη ιδιότητα και σημασιολογική αναπαράσταση: Η ανάπτυξη πιο εξελιγμένων μεθόδων για την επιλογή χαρακτηριστικών και τη σημασιολογική αναπαράσταση είναι ζωτικής σημασίας για την αποτελεσματικότερη γεφύρωση του χάσματος μεταξύ ορατών και μη ορατών κλάσεων.
- Cross-modal Zero-shot Learning: Η αξιοποίηση πληροφοριών από πολλαπλούς τρόπους (π.χ. κείμενο και εικόνες) υπόσχεται τη βελτίωση της κατανόησης και της ταξινόμησης μη ορατών τάξεων παρέχοντας πλουσιότερες πληροφορίες σχετικά με το context.
- Προσαρμογή και Εκμάθηση Μεταβίβασης Τομέα: Η ενσωμάτωση της μάθησης μηδενικής λήψης με τις τεχνικές προσαρμογής και μεταφοράς μάθησης τομέα μπορεί να βοηθήσει στον μετριασμό της προκατάληψης προς τις προβλεπόμενες τάξεις και στη βελτίωση των δυνατοτήτων γενίκευσης των μοντέλων.
- Θέματα ηθικής και δικαιοσύνης: Όπως συμβαίνει με όλες τις τεχνολογίες τεχνητής νοημοσύνης, η διασφάλιση ότι τα μοντέλα μάθησης μηδενικής βολής αναπτύσσονται και αναπτύσσονται με ηθικό και δίκαιο τρόπο είναι πρωταρχικής σημασίας, ιδιαίτερα όταν εφαρμόζεται σε ευαίσθητες εφαρμογές.

Το zero-shot learning ενσωματώνει ένα σημαντικό άλμα προς τη δημιουργία μοντέλων μηχανικής μάθησης που μπορούν να προσαρμοστούν σε νέα, αόρατα σενάρια χωρίς την ανάγκη εξαντλητικών συνόλων δεδομένων με ετικέτα. Αξιοποιώντας τη γνώση σχετικά με την εργασία και συνάγοντας ιδιότητες άγνωστων κλάσεων, το zero-shot learning όχι μόνο αμφισβητεί τα παραδοσιακά παραδείγματα της μηχανικής μάθησης αλλά επίσης ανοίγει νέες δυνατότητες για εφαρμογές σε ένα ευρύ φάσμα τομέων.

## 3.4 Ensemble learning

Το ensemble learning είναι μια ισχυρή τεχνική στη μηχανική μάθηση που περιλαμβάνει το συνδυασμό πολλαπλών μοντέλων για τη βελτίωση της συνολικής απόδοσης και της ικανότητας πρόβλεψης πέρα από αυτό που θα μπορούσε να επιτύχει οποιοδήποτε μοντέλο. Παρακάτω αναλύουμε αυτό το ML technique, τις διάφορες μεθοδολογίες του, όπως το bagging, boosting και stacking, το σκεπτικό πίσω από την αποτελεσματικότητά του, τις εφαρμογές του, τις προκλήσεις και τις μελλοντικές κατευθύνσεις.

Το ensemble learning λειτουργεί με την προϋπόθεση ότι ο συνδυασμός των προβλέψεων από πολλαπλά μοντέλα μπορεί να βελτιώσει σημαντικά την ακρίβεια και την ευρωστία της συνολικής πρόβλεψης. Αυτή η προσέγγιση βασίζεται στη θεωρία wisdom of crowds, η οποία υποδηλώνει ότι η συλλογική πρόβλεψη μιας ομάδας είναι συχνά πιο ακριβής από αυτή των μεμονωμένων μελών της. Στη μηχανική μάθηση, αυτό μεταφράζεται σε μείωση της διακύμανσης, της μεροληψίας ή και των δύο, οδηγώντας σε βελτιωμένα αποτελέσματα πρόβλεψης.

Η βασική λογική για την εκμάθηση συνόλου είναι η εκμετάλλευση της διαφορετικότητας μεταξύ των μοντέλων για την επίτευξη καλύτερης γενίκευσης σε αόρατα δεδομένα. Τα μεμονωμένα μοντέλα μπορεί να έχουν τα μοναδικά πλεονεκτήματα και τις αδυναμίες τους σε διαφορετικές περιοχές του χώρου εισόδου. Συνδυάζοντας αυτά τα μοντέλα, οι μέθοδοι συνόλου στοχεύουν στην εξισορρόπηση των επιμέρους σφαλμάτων τους, επιτυγχάνοντας έτσι μεγαλύτερη ακρίβεια και σταθερότητα. Αυτή η προσέγγιση είναι ιδιαίτερα αποτελεσματική στη μείωση του overfitting, καθώς είναι λιγότερο πιθανό για πολλά μοντέλα να γίνουν overfit τα δεδομένα με τον ίδιο ακριβώς τρόπο.
Το ensemble learning περιλαμβάνει διάφορες μεθοδολογίες, η καθεμία με τη μοναδική της στρατηγική για το συνδυασμό μοντέλων. Το Bagging περιλαμβάνει εκπαίδευση πολλαπλών παρουσιών ενός μοντέλου σε διαφορετικά υποσύνολα των δεδομένων εκπαίδευσης, που συνήθως sampled με αντικατάσταση (δείγματα bootstrap). Η τελική πρόβλεψη προκύπτει με τον υπολογισμό του μέσου όρου των προβλέψεων (για εργασίες παλινδρόμησης) ή με ψηφοφορία (για εργασίες ταξινόμησης). Το Random Forest είναι ένα από τα πιο δημοφιλή σύνολα bagging, το οποίο συνδυάζει πολλαπλά δέντρα απόφασης για να παράγει ένα πιο στιβαρό και ακριβές μοντέλο.
Η ενίσχυση είναι μια διαδοχική διαδικασία όπου κάθε μοντέλο προσπαθεί να διορθώσει τα λάθη των προκατόχων του. Τα μοντέλα προστίθενται διαδοχικά και τα δεδομένα εκπαίδευσης σταθμίζονται εκ νέου για να επικεντρωθούν σε περιπτώσεις που είχαν ταξινομηθεί εσφαλμένα από προηγούμενα μοντέλα. Η τελική πρόβλεψη γίνεται με βάση ένα σταθμισμένο άθροισμα των προβλέψεων από όλα τα μοντέλα. Αλγόριθμοι όπως ο AdaBoost και ο Gradient Boosting είναι γνωστές εφαρμογές της ενίσχυσης.

- Στοίβαξη (Γενίκευση στοίβαξης): Η στοίβαξη περιλαμβάνει εκπαίδευση πολλαπλών μοντέλων (συχνά διαφορετικά) και στη συνέχεια εκπαίδευση ενός μετα-μοντέλου σχετικά με τις προβλέψεις τους. Τα βασικά μοντέλα εκπαιδεύονται στο πλήρες σύνολο δεδομένων εκπαίδευσης και οι προβλέψεις τους (ή ένα υποσύνολο αυτών) χρησιμοποιούνται ως είσοδοι για το μετα-μοντέλο, το οποίο στη συνέχεια κάνει την τελική πρόβλεψη. Αυτή η προσέγγιση επιτρέπει την εκμάθηση των δυνατών και των αδυναμιών κάθε βασικού μοντέλου σε σχέση με

την εργασία πρόβλεψης. Το Ensemble Learning βρίσκει εφαρμογές σε ένα ευρύ φάσμα τομέων, αποδεικνύοντας την ευελιξία και την αποτελεσματικότητά του.

- Προγνωστική Μοντελοποίηση: Σε τομείς όπως τα χρηματοοικονομικά και η υγειονομική περίθαλψη, χρησιμοποιούνται μέθοδοι συνόλου για τη βελτίωση της ακρίβειας των προγνωστικών μοντέλων, είτε για την πρόβλεψη των τιμών των μετοχών είτε για τη διάγνωση ασθενειών.
- Computer Vision: Η εκμάθηση συνόλου βελτιώνει την ανίχνευση αντικειμένων και την ταξινόμηση εικόνων συνδυάζοντας μοντέλα εκπαιδευμένα σε διαφορετικά χαρακτηριστικά ή αρχιτεκτονικές.
- Επεξεργασία φυσικής γλώσσας (NLP): Στο NLP, οι μέθοδοι συνόλου βελτιώνουν τη γλωσσική μετάφραση, την ανάλυση συναισθημάτων και την ταξινόμηση του κειμένου συνδυάζοντας τις προβλέψεις από μοντέλα που έχουν εκπαιδευτεί σε διαφορετικές πτυχές της γλώσσας.

Παρά τα πλεονεκτήματά της, το ensemble learning συνοδεύεται από τις δικές του προκλήσεις:

- Αυξημένη πολυπλοκότητα: Ο συνδυασμός πολλαπλών μοντέλων αυξάνει την υπολογιστική πολυπλοκότητα, τόσο όσον αφορά τον χρόνο εκπαίδευσης όσο και την ερμηνεία του μοντέλου.
- Βέλτιστη επιλογή και συνδυασμός μοντέλων: Ο προσδιορισμός του σωστού μείγματος μοντέλων και της μεθόδου για το συνδυασμό των προβλέψεών τους δεν είναι τετριμμένο και συχνά απαιτεί εκτεταμένο πειραματισμό.
- Μειωμένες αποδόσεις: Υπάρχει ένα σημείο πέρα από το οποίο η προσθήκη περισσότερων μοντέλων στο σύνολο δεν βελτιώνει σημαντικά την απόδοση και μπορεί ακόμη και να οδηγήσει σε μειωμένη απόδοση του μοντέλου.

Το μέλλον του ensemble learning έγκειται στην αντιμετώπιση των τρεχουσών προκλήσεων και στην εξερεύνηση νέων μεθοδολογιών που μπορούν να ενισχύσουν περαιτέρω την εφαρμογή και την αποτελεσματικότητά του:

- Automated Ensemble Learning: Η αυτοματοποίηση της διαδικασίας επιλογής και συνδυασμού μοντέλων μέσω τεχνικών όπως το AutoML μπορεί να βοηθήσει στη βελτιστοποίηση των συνόλων με ελάχιστη ανθρώπινη παρέμβαση.
- Deep Ensemble Learning: Ενσωμάτωση μεθόδων συνόλου με βαθιά εκμάθηση, εξερεύνηση του τρόπου με τον οποίο μπορούν να συνδυαστούν διαφορετικές αρχιτεκτονικές για βελτιωμένη απόδοση.
- Ενεργειακά αποδοτικά σύνολα: Ανάπτυξη μεθόδων για τη μείωση του υπολογιστικού αποτυπώματος των μοντέλων συνόλων, καθιστώντας τα πιο βιώσιμα και προσβάσιμα για εφαρμογές στον πραγματικό κόσμο [15].

## 4. Solution Methodology

Οι ακόλουθες προσεγγίσεις χρησιμοποιήθηκαν στη προσπάθεια της δημιουργίας μιας λύσης για τη πρόβλεψη του διαβήτη:

- Decision tree classifier
- Random forest classifier
- SVM
- Gradient boosting
- neural networks

**Decision Tree Classifier**

Οι decision tree classifiers αποτελούν μία δημοφιλή μέθοδο supervised learning που χρησιμοποιείται στην εξόρυξη δεδομένων και τη μηχανική μάθηση. Τα δέντρα απόφασης είναι μοντέλα προβλέψεων που συνδυάζουν απλότητα και ευελιξία, καθιστώντας τα ιδιαίτερα χρήσιμα για την επίλυση ποικίλων προβλημάτων ταξινόμησης και παλινδρόμησης. Ένα δέντρο απόφασης αναπτύσσεται με τη διαδικασία διαίρεσης του σύνολου δεδομένων σε όλο και πιο μικρά υποσύνολα, ενώ παράλληλα αναπτύσσεται ένα συναφές δέντρο απόφασης. Η τελική μορφή του δέντρου περιλαμβάνει κόμβους αποφάσεων και κόμβους φύλλων. Κάθε κόμβος απόφασης στο δέντρο αντιστοιχεί σε ένα γνώρισμα ή χαρακτηριστικό του σύνολου δεδομένων και κάθε κόμβος φύλλο αντιπροσωπεύει μια τελική απόφαση ή πρόβλεψη. Η διαδικασία δημιουργίας ενός δέντρου απόφασης περιλαμβάνει την επιλογή των καλύτερων γνωρισμάτων για να διαχωριστεί το σύνολο δεδομένων σε υποσύνολα με βάση κριτήρια καθαρότητας, όπως η εντροπία και ο δείκτης Gini [16]. Στόχος είναι η μεγιστοποίηση της ομοιογένειας σε κάθε υποσύνολο. Αυτή η διαδικασία συνεχίζεται αναδρομικά μέχρι να επιτευχθούν ορισμένα κριτήρια διακοπής, όπως το μέγιστο βάθος του δέντρου ή ο ελάχιστος αριθμός δειγμάτων σε κάθε κόμβο φύλλο. Στο πεδίο της ιατρικής, και ειδικότερα στην πρόβλεψη του διαβήτη, οι ταξινομητές δέντρων απόφασης μπορούν να προσφέρουν σημαντικές επιδόσεις λόγω της ικανότητάς τους να χειρίζονται πολύπλοκες μη γραμμικές σχέσεις μεταξύ των γνωρισμάτων και της κλάσης στόχου. Για παράδειγμα, στην περίπτωση της διάγνωσης του διαβήτη, ένα δέντρο απόφασης μπορεί να χρησιμοποιήσει γνωρίσματα όπως η ηλικία, το βάρος, η πίεση, τα επίπεδα γλυκόζης και άλλους δείκτες για να προβλέψει την πιθανότητα ένα άτομο να αναπτύξει διαβήτη.

Η δυνατότητα των δέντρων απόφασης να παρέχουν διαισθητικά και εύκολα κατανοητά μοντέλα καθιστά τη μέθοδο ιδιαίτερα χρήσιμη στην ιατρική κοινότητα, όπου η διαφάνεια και η ερμηνευσιμότητα των προβλεπτικών μοντέλων είναι κρίσιμης σημασίας. Επιπλέον, η ευελιξία τους στην χειρισμό τόσο αριθμητικών όσο και κατηγορικών δεδομένων τα καθιστά ιδανικά για την εφαρμογή σε διαφορετικούς τύπους δεδομένων που συναντώνται στις ιατρικές έρευνες.
Συνοψίζοντας, τα decision tree classifiers αποτελούν ένα ισχυρό εργαλείο στον τομέα της πρόβλεψης και διάγνωσης ιατρικών καταστάσεων όπως ο διαβήτης. Η ικανότητά τους να αναλύουν και να ερμηνεύουν περίπλοκες σχέσεις δεδομένων παρέχει πολύτιμες πληροφορίες για τη λήψη αποφάσεων στον τομέα της υγείας, συμβάλλοντας στην πρόληψη, την έγκαιρη διάγνωση και την αποτελεσματική θεραπεία του διαβήτη.

**Random forest classifier**

Οι ταξινομητές Random Forest αποτελούν μία από τις πιο δυνατές και πολυχρησιμοποιημένες τεχνικές στον τομέα του Machine Learning. Βασίζονται στην ιδέα της συνένωσης πολλαπλών δέντρων απόφασης για τη δημιουργία ενός "δάσους", με σκοπό την αύξηση της ακρίβειας και την ανθεκτικότητα σε overfitting [17]. Η βασική ιδέα πίσω από τους Random Forest classifiers είναι η τεχνική του "bagging" ή αλλιώς Bootstrap Aggregating. Κάθε δέντρο στο "δάσος" εκπαιδεύεται σε ένα τυχαίο υποσύνολο του συνολικού σετ δεδομένων, με αντικατάσταση [18]. Αυτό σημαίνει ότι τα ίδια δεδομένα μπορεί να επιλεγούν περισσότερες από μία φορές για την εκπαίδευση διαφορετικών δέντρων, ενώ κάποια άλλα μπορεί να μην επιλεγούν καθόλου. Επιπλέον, κατά τη διαδικασία διαχωρισμού ενός κόμβου στα δέντρα απόφασης ενός Random Forest, δεν εξετάζονται όλα τα διαθέσιμα χαρακτηριστικά. Αντίθετα, επιλέγεται ένα τυχαίο υποσύνολο χαρακτηριστικών και η καλύτερη διαχωριστική ερώτηση επιλέγεται από αυτό το υποσύνολο. Αυτό συμβάλλει στην περαιτέρω αύξηση της διαφοροποίησης μεταξύ των δέντρων, ενισχύοντας την απόδοση και την ικανότητα γενίκευσης του τελικού μοντέλου. Η διαδικασία πρόβλεψης με τη χρήση Random Forest είναι απλή: κάθε δέντρο στο "δάσος" κάνει τη δική του πρόβλεψη και η τελική απόφαση προκύπτει μέσω ψηφοφορίας για ταξινομήσεις ή μέσω του μέσου όρου για παλινδρόμηση. Η διαδικασία αυτή εξασφαλίζει ότι η επιρροή των ακραίων ή αντιφατικών προβλέψεων μεμονωμένων δέντρων μειώνεται, ενώ ταυτόχρονα αυξάνεται η ακρίβεια και η αξιοπιστία του συνολικού μοντέλου. Στον τομέα της πρόβλεψης και διάγνωσης ασθενειών, οι Random Forest classifiers έχουν επιδείξει ιδιαίτερα ενθαρρυντικά αποτελέσματα. Η ικανότητά τους να χειρίζονται μεγάλο αριθμό χαρακτηριστικών και δεδομένων, καθώς και η ικανότητά τους να αναγνωρίζουν περίπλοκες μη γραμμικές σχέσεις, τα καθιστά ιδανικά για την ανάλυση ιατρικών δεδομένων. Για παράδειγμα, στην περίπτωση της διάγνωσης του διαβήτη, ένας ταξινομητής Random Forest μπορεί να εκπαιδευτεί για να αναγνωρίζει τα πρότυπα και τους συνδυασμούς των διαφόρων βιοδεικτών που συσχετίζονται με την εμφάνιση της νόσου. Η ανθεκτικότητα των Random Forest σε ακραίες τιμές και η ικανότητά τους να παρέχουν ερμηνευτικά αποτελέσματα μέσω της σημαντικότητας των χαρακτηριστικών, επιτρέπουν στους ερευνητές και τους ιατρούς να κατανοήσουν καλύτερα τους παράγοντες κινδύνου και τις δυναμικές που συμβάλλουν στην ανάπτυξη του διαβήτη. Αυτό μπορεί να συμβάλει σημαντικά στην πρόληψη, στην έγκαιρη διάγνωση και στην πιο στοχευμένη θεραπευτική προσέγγιση. Συμπερασματικά, οι Random Forest classifiers προσφέρουν έναν πολύ ισχυρό και ευέλικτο μηχανισμό για την ανάλυση και την ερμηνεία μεγάλων και πολύπλοκων συνόλων δεδομένων. Η εφαρμογή τους στην ιατρική και ειδικά στη διάγνωση ασθενειών όπως ο διαβήτης υπόσχεται να προσφέρει βαθύτερες γνώσεις και να βελτιώσει τις στρατηγικές πρόληψης και θεραπείας, ενισχύοντας την ικανότητα των επαγγελματιών υγείας να προσφέρουν πιο αποτελεσματική φροντίδα.

**SVM**

Τα Support Vector Machines (SVM) αποτελούν μία από τις πιο ισχυρές και ευρέως χρησιμοποιούμενες μεθόδους στον τομέα του Machine Learning [10]. Αυτή η τεχνική είναι ιδιαίτερα δημοφιλής για την επίλυση προβλημάτων ταξινόμησης και παλινδρόμησης, χάρη στην ικανότητά της να χειρίζεται high level data και να βρίσκει πολύπλοκα όρια απόφασης.

Τα SVMs εργάζονται βασιζόμενα στην ιδέα της εύρεσης του "βέλτιστου υπερεπιπέδου" που διαχωρίζει τις κλάσεις δεδομένων στο χώρο των χαρακτηριστικών. Το "βέλτιστο" υπερεπίπεδο είναι εκείνο που έχει τη μέγιστη margin από τα πλησιέστερα δεδομένα προς το υπερεπίπεδο από κάθε κλάση, τα οποία ονομάζονται support vectors. Για να επιτευχθεί αυτό, τα SVM χρησιμοποιούν μια τεχνική γνωστή ως kernel trick, η οποία τους επιτρέπει να χειριστούν γραμμικά μη διαχωρίσιμα δεδομένα μεταφέροντας τα δεδομένα σε έναν υψηλότερο διαστατικό χώρο όπου η γραμμική διαχωρισιμότητα είναι δυνατή. Οι πιο συνήθεις kernels που χρησιμοποιούνται είναι ο γραμμικός, ο πολυωνυμικός, ο radial basis function (RBF) και ο sigmoid. Στον τομέα της ιατρικής, και ειδικότερα στην πρόβλεψη και διάγνωση ασθενειών, τα SVMs έχουν επιδείξει εξαιρετικές επιδόσεις. Ένας από τους τομείς όπου οι SVM έχουν εφαρμοστεί επιτυχώς είναι η πρόβλεψη του διαβήτη. Η δυνατότητά τους να αναλύουν και να εντοπίζουν σύνθετες σχέσεις μεταξύ διαφόρων δεικτών και της παρουσίας του διαβήτη καθιστά τα SVMs ιδιαίτερα κατάλληλα για αυτό το πεδίο.

Για παράδειγμα, στην πρόβλεψη του διαβήτη, μια SVM μπορεί να εκπαιδευτεί χρησιμοποιώντας δεδομένα από ένα σύνολο δεικτών όπως η συγκέντρωση γλυκόζης, η πίεση του αίματος, το δείκτη μάζας σώματος (BMI), η ιστορία οικογένειας, και άλλα σχετικά χαρακτηριστικά [19]. Το μοντέλο μπορεί στη συνέχεια να χρησιμοποιηθεί για να κατατάξει τους ασθενείς σε υψηλό ή χαμηλό κίνδυνο για την ανάπτυξη του διαβήτη, επιτρέποντας την πρόληψη και την έγκαιρη παρέμβαση. Η ακρίβεια και η ανθεκτικότητα των SVMs σε ακραίες τιμές και η ευκολία με την οποία μπορούν να χειριστούν υψηλοδιάστατα δεδομένα τα καθιστούν ιδιαίτερα πολύτιμα εργαλεία στον τομέα της ιατρικής έρευνας. Επιπλέον, η δυνατότητα επιλογής διαφορετικών πυρήνων επιτρέπει στους ερευνητές να πειραματιστούν και να προσαρμόσουν τα μοντέλα στις συγκεκριμένες ανάγκες της κάθε έρευνας, αυξάνοντας την ευελιξία και την εφαρμοσιμότητα των SVM σε μια πληθώρα ιατρικών προκλήσεων. Συνοψίζοντας, τα Support Vector Machines δείχνουν πως μπορούν να χρησιμοποιηθούν ως ένα ισχυρό εργαλείο στον αγώνα κατά του διαβήτη [20], προσφέροντας στους επαγγελματίες της υγείας μια αξιόπιστη μέθοδο για την πρόβλεψη και τη διάγνωση της νόσου. Η ικανότητά τους να αναλύουν με ακρίβεια τα δεδομένα και να παρέχουν σαφείς προβλέψεις μπορεί να συμβάλει σημαντικά στην πρόληψη, την έγκαιρη ανίχνευση και την αποτελεσματική διαχείριση του διαβήτη, βελτιώνοντας την ποιότητα ζωής των ασθενών και συμβάλλοντας στη μείωση των συνολικών επιπτώσεων της νόσου [21].

**Gradient Boosting**

Το Gradient Boosting είναι μια ισχυρή τεχνική Machine Learning που χρησιμοποιείται για προβλήματα ταξινόμησης και παλινδρόμησης. Αυτή η τεχνική ενισχυτικής μάθησης συνδυάζει πολλούς ασθενείς ταξινομητές για να δημιουργήσει έναν ισχυρότερο ταξινομητή. Τα "ασθενή" μοντέλα που συνήθως χρησιμοποιούνται στο Gradient Boosting είναι απλά δέντρα απόφασης. Αυτή η διαδικασία ενίσχυσης λαμβάνει χώρα σε στάδια, με κάθε νέο μοντέλο να προσπαθεί να διορθώσει τα λάθη του προηγούμενου. Η ιδέα πίσω από το Gradient Boosting ξεκίνησε από την παρατήρηση ότι είναι δυνατό να δημιουργηθεί ένας ισχυρός προβλεπτικός ταξινομητής συνδυάζοντας πολλούς ασθενείς. Η τεχνική αυτή έχει τις ρίζες της στην αλγοριθμική θεωρία της πληροφορίας και στη στατιστική μάθηση. Στο Gradient Boosting, κάθε νέο δέντρο απόφασης κατασκευάζεται ώστε να

μειώνει το λάθος που προκύπτει από τα προηγούμενα δέντρα. Αυτό επιτυγχάνεται με τη χρήση ενός αλγορίθμου βελτιστοποίησης που γνωστοποιείται ως gradient descent, ο οποίος εντοπίζει την κατεύθυνση που θα οδηγήσει στη μέγιστη μείωση του λάθους. Κάθε δέντρο επομένως "διορθώνει" τα λάθη του προηγούμενου, καθιστώντας το μοντέλο όλο και πιο ισχυρό με την προσθήκη κάθε νέου δέντρου. Ένα από τα κύρια πλεονεκτήματα του Gradient Boosting είναι η ευελιξία του. Μπορεί να χειριστεί διάφορους τύπους δεδομένων, συμπεριλαμβανομένων των αριθμητικών, κατηγορικών και διατεταγμένων μεταβλητών. Επιπλέον, λόγω της δυνατότητας χειρισμού μη γραμμικών σχέσεων και της ανθεκτικότητάς του στην υπερπροσαρμογή, είναι ιδιαίτερα κατάλληλο για περίπλοκα σετ δεδομένων. Στον τομέα της ιατρικής και της πρόβλεψης ασθενειών, το Gradient Boosting μπορεί να προσφέρει σημαντικά πλεονεκτήματα. Για παράδειγμα, στην πρόβλεψη του διαβήτη, το Gradient Boosting μπορεί να αναλύσει μεγάλο όγκο δεδομένων από διάφορες πηγές, όπως αποτελέσματα εξετάσεων, ιστορικό ασθενών και διατροφικές συνήθειες, για να προβλέψει την πιθανότητα εμφάνισης της νόσου. Μέσω της δυνατότητας αυτής, οι ειδικοί στην υγεία μπορούν να προσδιορίσουν τα άτομα υψηλού κινδύνου πιο αποτελεσματικά και να εφαρμόσουν προληπτικές ή θεραπευτικές στρατηγικές συντομότερα και με μεγαλύτερη ακρίβεια. Ένα ακόμη σημαντικό πλεονέκτημα του Gradient Boosting είναι η δυνατότητά του να παρέχει ερμηνείες για τα αποτελέσματά του, καθιστώντας τις προβλέψεις κατανοητές από επαγγελματίες που μπορεί να μην έχουν βαθιά εξειδίκευση στο Machine Learning. Η δυνατότητα αυτή είναι ιδιαίτερα σημαντική στον τομέα της ιατρικής, όπου η κατανόηση των παραγόντων που συμβάλλουν στην εμφάνιση μιας ασθένειας είναι κρίσιμη για την αποτελεσματική διαχείριση και θεραπεία της. Συνοψίζοντας, το Gradient Boosting αποτελεί μια προηγμένη τεχνική Machine Learning που προσφέρει σημαντικές δυνατότητες στην ανάλυση δεδομένων και την πρόβλεψη αποτελεσμάτων, ιδιαίτερα σε περίπλοκα προβλήματα όπως η πρόβλεψη ασθενειών. Η εφαρμογή του στην πρόβλεψη του διαβήτη υπόσχεται να βελτιώσει την ικανότητα των επαγγελματιών υγείας να ανιχνεύουν προδιαβητικές καταστάσεις, να προσδιορίζουν τους ασθενείς υψηλού κινδύνου και να εφαρμόζουν πιο αποτελεσματικά θεραπευτικά σχέδια, συμβάλλοντας έτσι στη βελτίωση της ποιότητας ζωής και τη μείωση των επιπτώσεων της νόσου [22].

**Νευρωνικά Δίκτυα (neural networks)**

Τα νευρωνικά δίκτυα αποτελούν έναν κρίσιμο τομέα της τεχνητής νοημοσύνης και Machine Learning, προσφέροντας τη δυνατότητα στα συστήματα να μαθαίνουν, να προσαρμόζονται και να αναπτύσσουν δεξιότητες από δεδομένα. Ο όρος "νευρωνικά δίκτυα" αναφέρεται σε μοντέλα εμπνευσμένα από τον ανθρώπινο εγκέφαλο, που αποτελούνται από επίπεδα νευρώνων ή κόμβων, συνδεδεμένα μέσω συνάψεων. Αυτά τα μοντέλα έχουν την ικανότητα να αναλύουν και να ερμηνεύουν περίπλοκα δεδομένα, κάνοντας τα κατάλληλα για μια πληθώρα εφαρμογών. Υπάρχουν διάφοροι τύποι νευρωνικών δικτύων, καθένας από τους οποίους έχει σχεδιαστεί για να εξυπηρετεί συγκεκριμένες ανάγκες και εφαρμογές [23]. Τα πιο κοινά είναι τα προς τα εμπρός νευρωνικά δίκτυα (Feedforward Neural Networks), τα νευρωνικά δίκτυα αναδρομής (Recurrent Neural Networks - RNNs), τα συνελικτικά νευρωνικά δίκτυα (Convolutional Neural Networks - CNNs) και τα νευρωνικά δίκτυα βαθιάς μάθησης (Deep Learning Neural Networks). Τα προς τα εμπρός νευρωνικά δίκτυα είναι η πιο βασική μορφή, όπου οι πληροφορίες κινούνται μόνο προς τα εμπρός, από την είσοδο προς την έξοδο, χωρίς

καμία επαναληπτική διαδικασία. Αυτά τα δίκτυα είναι ιδανικά για βασικές εργασίες πρόβλεψης και ταξινόμησης. Τα RNNs, από την άλλη πλευρά, περιλαμβάνουν επαναληπτικές διαδρομές που επιτρέπουν στην πληροφορία να κυκλοφορεί εντός του δικτύου για έναν ορισμένο αριθμό βημάτων. Αυτό τα καθιστά ιδανικά για την επεξεργασία και πρόβλεψη χρονοσειρών και για εφαρμογές που αφορούν φυσική γλώσσα, όπως η αναγνώριση ομιλίας ή η μετάφραση. Τα CNNs έχουν σχεδιαστεί για την αναγνώριση και την κατηγοριοποίηση εικόνων. Χρησιμοποιούν μια τεχνική που ονομάζεται συνέλιξη για την εξαγωγή χαρακτηριστικών από τα δεδομένα εισόδου, καθιστώντας τα ιδανικά για την επεξεργασία εικόνων, την αναγνώριση προτύπων και την ανάλυση οπτικών δεδομένων. Τα νευρωνικά δίκτυα βαθιάς μάθησης (Deep Learning Neural Networks) εκπροσωπούν μια πιο προχωρημένη κατηγορία, η οποία χρησιμοποιεί πολλά επίπεδα επεξεργασίας για την εκμάθηση από δεδομένα. Αυτό τους επιτρέπει να αναγνωρίζουν περίπλοκα μοτίβα και να εκτελούν δύσκολες εργασίες, όπως η αυτόνομη οδήγηση, η αναγνώριση φωνής, και η αυτόματη μετάφραση γλωσσών. Η ικανότητά τους να μαθαίνουν από τεράστιες ποσότητες δεδομένων και να εξάγουν χρήσιμες πληροφορίες κάνει τα Deep Learning Networks κυρίαρχα στον τομέα της τεχνητής νοημοσύνης.

Ένα άλλο σημαντικό είδος νευρωνικού δικτύου είναι τα Generative Adversarial Networks (GANs). Τα GANs αποτελούνται από δύο δίκτυα, ένα γεννητικό και ένα διακριτικό, τα οποία εκπαιδεύονται παράλληλα με σκοπό τη βελτίωση της απόδοσής τους μέσω του ανταγωνισμού. Τα GANs χρησιμοποιούνται ευρέως για τη δημιουργία ρεαλιστικών εικόνων, βίντεο και άλλων τύπων δεδομένων, ανοίγοντας νέους δρόμους στον σχεδιασμό παιχνιδιών, την τέχνη και την εκπαίδευση. Για την εκπαίδευση των νευρωνικών δικτύων χρησιμοποιούνται διάφορες τεχνικές, με την πιο διαδεδομένη να είναι η αντίστροφη διάδοση (back propagation). Η τεχνική αυτή επιτρέπει την αποτελεσματική εκμάθηση ενός νευρωνικού δικτύου μέσω της προσαρμογής των βαρών των συνάψεων ανάλογα με το σφάλμα της πρόβλεψης. Με αυτόν τον τρόπο, το δίκτυο μπορεί να βελτιώνει σταδιακά την απόδοσή του σε μια δεδομένη εργασία.

Η σημασία των νευρωνικών δικτύων στον σύγχρονο κόσμο είναι αδιαμφισβήτητη, καθώς έχουν ενισχύσει την αυτοματοποίηση, την ανάλυση δεδομένων και την τεχνητή νοημοσύνη σε πολλούς τομείς. Από την ιατρική έρευνα και τη διαχείριση κινδύνων μέχρι την αυτόματη οδήγηση και την προσωπική ψηφιακή βοήθεια, τα νευρωνικά δίκτυα προσφέρουν λύσεις που άλλοτε θεωρούνταν αδύνατες.

Δεδομένου ότι τα νευρωνικά δίκτυα έχουν την ικανότητα να επεξεργάζονται και να αναλύουν μεγάλες ποσότητες δεδομένων για την εξαγωγή πολύπλοκων μοτίβων και σχέσεων, για την πρόβλεψη διαβήτη, θα μπορούσε να χρησιμοποιηθεί ένα νευρωνικό δίκτυο βαθιάς μάθησης (Deep Learning Neural Network). Αυτός ο τύπος δικτύου είναι ικανός να αναλύει μεγάλες ποσότητες δεδομένων υγείας, όπως αποτελέσματα εξετάσεων, ιστορικό υγείας, γενετικές πληροφορίες και άλλους παράγοντες κινδύνου, για να προβλέψει την πιθανότητα εμφάνισης διαβήτη σε ένα άτομο. Η χρήση των Deep Learning Neural Networks [24] αυτοματοποιημένη εξαγωγή χαρακτηριστικών και την αναγνώριση πολύπλοκων μοτίβων μέσα στα δεδομένα που δεν είναι εύκολα αντιληπτά από τους ανθρώπους. Αυτό καθιστά τα Deep Learning Networks ιδιαίτερα κατάλληλα για την πρόβλεψη ασθενειών όπως ο διαβήτης, όπου οι σχετιζόμενοι παράγοντες κινδύνου

και τα συμπτώματα μπορεί να είναι πολυάριθμα και πολύπλοκα. Η δυνατότητα των Deep Learning Neural Networks να μαθαίνουν από τα δεδομένα και να βελτιώνονται με την πάροδο του χρόνου εξασφαλίζει ότι οι προβλέψεις για την εμφάνιση διαβήτη θα γίνονται όλο και πιο ακριβείς.

## 4.1 Γλώσσα προγραμματισμού

Python αποτελεί μια εξαιρετική επιλογή για έρευνα που ενσωματώνει τη μηχανική μάθηση, λόγω της απλότητας, της ευελιξίας και της ισχυρής οικοσυστημικής υποστήριξης που προσφέρει. Η ευκολία προγραμματισμού και η διαισθητική σύνταξή του καθιστούν την Python προσβάσιμη ακόμη και για ερευνητές που δεν έχουν βαθιά εμπειρία στον προγραμματισμό, επιτρέποντας τους να εστιάσουν στην ανάλυση και την ερμηνεία των δεδομένων αντί της περιπλοκότητας του κώδικα. Επιπλέον, η Python υποστηρίζεται από μια εκτενή βιβλιοθήκη εργαλείων και πακέτων όπως NumPy, Pandas, Matplotlib, Scikit-learn και TensorFlow, που παρέχουν προκατασκευασμένες λειτουργίες για διαχείριση δεδομένων, ανάλυση και μηχανική μάθηση, μειώνοντας σημαντικά τον χρόνο ανάπτυξης. Η κοινότητα της Python είναι επίσης ένας ισχυρός παράγοντας, προσφέροντας πλούσια τεκμηρίωση, εκπαιδευτικό υλικό και υποστήριξη, κάτι που είναι καθοριστικό για την επίλυση προβλημάτων και την ανταλλαγή γνώσεων. Συνεπώς, η Python αποτελεί μια ιδανική γλώσσα για ερευνητές που επιδιώκουν να ενσωματώσουν τη μηχανική μάθηση στην ερευνητική τους δουλειά, παρέχοντας ένα ισχυρό, αλλά προσβάσιμο εργαλείο για την εξερεύνηση και την κατανόηση των δεδομένων.

## 4.2 Βιβλιοθήκες

Η έρευνα διεκπεραιώθηκε με τη χρήση της γλώσσα Python και των κατάλληλων βιβλιοθηκών. Από κάτω παρουσιάζονται οι βιβλιοθήκες που χρησιμοποιήθηκαν:

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import itertools
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression, ElasticNet
from sklearn.metrics import mean_squared_error, r2_score, accuracy_score, precision_score, recall_score, f1_score
from sklearn.utils import resample
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import cross_val_score
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import AdaBoostClassifier
```

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.naive_bayes import BernoulliNB
import warnings
import torch
import torch.nn as nn
import torch.optim as optim
from torch.utils.data import DataLoader, TensorDataset
```

### 4.2.1 Pandas

Το Pandas είναι ένα εργαλείο ανάλυσης και χειρισμού δεδομένων ανοιχτού κώδικα που έχει δημιουργηθεί πάνω από τη γλώσσα προγραμματισμού Python. Προσφέρει δομές δεδομένων και λειτουργίες για τον χειρισμό αριθμητικών πινάκων και χρονοσειρών, κάνοντάς το ένα αναντικατάστατο εργαλείο για εργασίες στον τομέα της επιστήμης δεδομένων, της στατιστικής ανάλυσης και της μηχανικής μάθησης. Το όνομα "Pandas" προέρχεται από τον όρο "panel data", έναν όρο της οικονομετρίας για δομημένα σύνολα δεδομένων πολλαπλών διαστάσεων. Κυρίως, το Pandas είναι σχεδιασμένο για να δουλεύει με tabular, ή heterogenous δεδομένα, σε αντίθεση με άλλες βιβλιοθήκες Python όπως το NumPy, το οποίο διαχειρίζεται πιο αποδοτικά ομοιογενή αριθμητικά δεδομένα σε πίνακες. Οι δύο κύριες βιβλιοθήκες του Pandas είναι το DataFrame και το Series. Ένας DataFrame είναι μια δισδιάστατη, με δυνατότητα αλλαγής μεγέθους, πιθανώς ετερογενές tabular με labeled άξονες (σειρές και στήλες). Από την άλλη πλευρά, ένα Series είναι μια μονοδιάστατη labelled σειρά ικανή να κρατήσει οποιοδήποτε τύπο δεδομένων. Αυτές οι δομές δεδομένων είναι χτισμένες πάνω από τα NumPy arrays, παρέχοντας αποδοτική πρόσβαση σε ένα πλήθος μεθόδων για τον χειρισμό και την ανάλυση δεδομένων.
Το Pandas απλοποιεί εργασίες όπως η εισαγωγή δεδομένων από διάφορες μορφές αρχείων (CSV, Excel, SQL databases κ.ά.), η καθαριότητα δεδομένων, η μετασχηματισμός, η συγχώνευση, η διαμόρφωση και η συγκέντρωση δεδομένων. Επίσης, προσφέρει ισχυρές λειτουργίες για τη διαχείριση χρονοσειρών, χειριζόμενο με ευκολία τύπους δεδομένων ημερομηνίας και ώρας και παρέχοντας εργαλεία για εκτελέσεις σύνθετων λειτουργιών όπως η χρονική ευρετηρίαση, το resampling και οι χρονικές μετατοπίσεις. Κάποια από τα προτερήματά του είναι ότι είναι εύκολο, highly efficient, περιεκτικό, και έχει την υποστήριξη του development community. Το Pandas είναι πολυσχιδές, βρίσκοντας εφαρμογές σε διάφορους τομείς όπου η ανάλυση δεδομένων είναι κλειδί. Στα χρηματοοικονομικά, χρησιμοποιείται για την ανάλυση της αγοράς μετοχών, την ανάπτυξη στρατηγικών αλγοριθμικού trading και το χρηματοοικονομικό μοντελοποίηση. Στην ακαδημαϊκή κοινότητα και την έρευνα, υποστηρίζει τον καθαρισμό δεδομένων, τη μετασχηματισμό και την εξερεύνηση περίπλοκων συνόλων δεδομένων. Επίσης, παίζει έναν κρίσιμο ρόλο σε έργα μηχανικής μάθησης, όπου χρησιμοποιείται για προετοιμασία δεδομένων, εξαγωγή χαρακτηριστικών και εξερευνητική ανάλυση δεδομένων. Με βάση τα παραπάνω, είναι κατανοητό γιατί η συγκεκριμένη βιβλιοθήκη επιλέχθηκε για το συγκεκριμένο project [25].

### 4.2.2 matplotlib

Το Matplotlib αποτελεί μία από τις πιο δημοφιλείς βιβλιοθήκες στη γλώσσα προγραμματισμού

Python για τη δημιουργία στατικών, διαδραστικών και κινούμενων οπτικοποιήσεων δεδομένων. Αναπτύχθηκε από τον John D. Hunter το 2003, και προσφέρει μια εξαιρετικά προσαρμόσιμη πλατφόρμα για την κατασκευή διαγραμμάτων, γραφημάτων και άλλων μορφών γραφικών παραστάσεων, καθιστώντας το ένα απαραίτητο εργαλείο για επιστήμονες δεδομένων, αναλυτές και ερευνητές από διάφορους επιστημονικούς κλάδους. Το Matplotlib προσφέρει μια ευρεία γκάμα από λειτουργίες και στυλ, που επιτρέπουν τη δημιουργία σχεδόν κάθε είδους γραφικής παράστασης που μπορεί να χρειαστεί κανείς, από απλά διαγράμματα όπως bar charts και histograms, μέχρι πιο πολύπλοκες οπτικοποιήσεις όπως heatmaps και τρισδιάστατα διαγράμματα. Επιπλέον, η δυνατότητα της βιβλιοθήκης να ενσωματώνεται εύκολα με άλλες βιβλιοθήκες ανάλυσης δεδομένων της Python, όπως το NumPy και το Pandas, προσθέτει στην ευελιξία και την πρακτικότητά της. Ένα από τα βασικά πλεονεκτήματα του Matplotlib είναι η δυνατότητά του να προσαρμόζεται σε διάφορες εξόδους, υποστηρίζοντας πολλαπλές μορφές αρχείων για την αποθήκευση των γραφικών παραστάσεων, όπως PNG, PDF, SVG, και EPS. Αυτό καθιστά το Matplotlib ιδανικό για την παραγωγή υψηλής ποιότητας γραφικών για εκτύπωση ή ηλεκτρονική δημοσίευση. Επιπροσθέτως, η κοινότητα του Matplotlib είναι ενεργή και συνεχώς επεκτείνεται, παρέχοντας τακτικές ενημερώσεις, διορθώσεις και νέες δυνατότητες [26].

### 4.2.3 Numpy

Το Numpy είναι μια βασική βιβλιοθήκη για την επεξεργασία επιστημονικών δεδομένων στην Python, παρέχοντας έναν αποτελεσματικό μηχανισμό για τη δημιουργία και τον χειρισμό πολυδιάστατων πινάκων ή arrays. Αναπτύχθηκε αρχικά από τον Travis Oliphant το 2005, το Numpy αντιπροσωπεύει τον πυρήνα πάνω στον οποίο έχουν χτιστεί πολλές άλλες βιβλιοθήκες επιστημονικού υπολογισμού και ανάλυσης δεδομένων της Python, όπως το SciPy, Matplotlib, Pandas, και Scikit-learn, αποδεικνύοντας την αναντικατάστατη αξία του στον τομέα της επιστήμης δεδομένων. Η βασική δομή δεδομένων που παρέχει το Numpy είναι ο πολυδιάστατος πίνακας, ή ndarray, ο οποίος επιτρέπει την αποθήκευση και τον χειρισμό αριθμητικών δεδομένων με ταχύτητα και αποτελεσματικότητα. Αυτό είναι εφικτό λόγω της εσωτερικής αναπαράστασης των δεδομένων σε συνεχόμενη μνήμη, κάτι που διευκολύνει τις γρήγορες πράξεις αριθμητικής και λογικής σε ολόκληρο τον πίνακα. Επιπροσθέτως, το Numpy υποστηρίζει την εκτέλεση διανυσματικών και μητρικών πράξεων χωρίς την ανάγκη για επαναληπτικές δομές ελέγχου, όπως οι βρόχοι, προσφέροντας μια κομψή και ισχυρή σύνταξη για την επιστημονική προγραμματιστική. Το Numpy αντιμετωπίζει αποτελεσματικά πολλά από τα προβλήματα που σχετίζονται με την επεξεργασία δεδομένων στην Python, κυρίως τις προκλήσεις της αποδοτικότητας και της ευκολίας χρήσης σε σχέση με τα αριθμητικά δεδομένα. Μέσω της χρήσης των πινάκων του Numpy, είναι δυνατή η εκτέλεση σύνθετων αριθμητικών και στατιστικών υπολογισμών με ελάχιστο κώδικα και υψηλή απόδοση, ενισχύοντας την παραγωγικότητα των ερευνητών και των αναλυτών.

Εκτός από την άμεση εφαρμογή του στην επιστήμη δεδομένων και τον επιστημονικό προγραμματισμό, το Numpy είναι επίσης θεμελιώδες για την ανάπτυξη και την εκπαίδευση αλγορίθμων μηχανικής μάθησης, όπου οι πολυδιάστατοι πίνακες χρησιμοποιούνται για την αναπαράσταση δεδομένων, ετικετών και παραμέτρων των μοντέλων. Η δυνατότητα του Numpy να χειρίζεται μεγάλους όγκους δεδομένων με υψηλή ταχύτητα και ακρίβεια το καθιστά αναντικατάστατο σε σύγχρονες εφαρμογές Τεχνητής Νοημοσύνης και Μηχανικής Μάθησης [27].

### 4.2.4 Scikit learn

Το Scikit-learn αποτελεί μία από τις πιο διαδεδομένες βιβλιοθήκες στην Python για την εφαρμογή

μηχανικής μάθησης, προσφέροντας ένα ευρύ φάσμα αλγορίθμων για εποπτευόμενη και μη εποπτευόμενη μάθηση, καθώς και για την προετοιμασία δεδομένων και την αξιολόγηση μοντέλων. Αναπτύχθηκε αρχικά από τον David Cournapeau ως μέρος ενός έργου Google Summer of Code το 2007, το Scikit-learn έχει εξελιχθεί σε ένα ισχυρό εργαλείο για την επιστήμη δεδομένων και την τεχνητή νοημοσύνη, χάρη στην ευελιξία, την ευκολία χρήσης και την πληθώρα δυνατοτήτων που παρέχει. Η βιβλιοθήκη Scikit-learn καθιστά δυνατή την εφαρμογή μιας σειράς αλγορίθμων μηχανικής μάθησης, όπως ταξινόμηση, παλινδρόμηση, συσταδοποίηση και μείωση διαστάσεων, με ελάχιστες γραμμές κώδικα. Οι αλγόριθμοι αυτοί είναι προσβάσιμοι μέσω μιας συνεπούς και καλά τεκμηριωμένης API, κάτι που διευκολύνει την εξερεύνηση και την αξιοποίησή τους από αναλυτές και επιστήμονες δεδομένων, ανεξάρτητα από το επίπεδο εμπειρίας τους στον προγραμματισμό. Ένα από τα κυριότερα πλεονεκτήματα του Scikit-learn είναι η ικανότητά του να διευκολύνει την προετοιμασία δεδομένων, προσφέροντας εργαλεία για την κανονικοποίηση, την κωδικοποίηση κατηγοριών και την επιλογή χαρακτηριστικών, μεταξύ άλλων. Αυτό σημαίνει ότι οι χρήστες μπορούν να εστιάσουν στην ανάλυση και την εκπαίδευση μοντέλων, αντί να ανησυχούν για την πολυπλοκότητα της προετοιμασίας και του καθαρισμού των δεδομένων. Παράλληλα, το Scikit-learn προσφέρει εκτενή υποστήριξη για την αξιολόγηση και τη βελτιστοποίηση των μοντέλων μηχανικής μάθησης. Εργαλεία όπως η διασταυρούμενη επικύρωση (cross-validation), τα μετρικά απόδοσης και οι τεχνικές εύρεσης βέλτιστων παραμέτρων (grid search, random search) επιτρέπουν στους ερευνητές να αναπτύξουν, να δοκιμάσουν και να βελτιστοποιήσουν τα μοντέλα τους με συνέπεια και ακρίβεια. Η συνεχής ανάπτυξη και βελτίωση του Scikit-learn από μια ενεργή κοινότητα συνεισφέρει στην παραμονή της βιβλιοθήκης στην κορυφή των εργαλείων μηχανικής μάθησης. Η πληθώρα προσθηκών και βελτιώσεων που πραγματοποιούνται συνεχώς εξασφαλίζει ότι το Scikit-learn παραμένει συμβατό με τις τελευταίες τάσεις και τεχνολογίες στον τομέα της μηχανικής μάθησης, κάτι που αποτελεί κρίσιμο παράγοντα για την επιτυχία των σύγχρονων ερευνητικών και αναλυτικών εργασιών. Ένας ακόμα τομέας στον οποίο το Scikit-learn διαπρέπει είναι η δυνατότητα εύκολης ενσωμάτωσης με άλλες βιβλιοθήκες της Python για επιστημονικό υπολογισμό, όπως το Numpy και το SciPy, καθώς και με βιβλιοθήκες οπτικοποίησης δεδομένων όπως το Matplotlib. Αυτή η ικανότητα προσφέρει μια ολοκληρωμένη πλατφόρμα για την ανάλυση δεδομένων, από την προετοιμασία και την εξερεύνηση μέχρι την εκπαίδευση, την αξιολόγηση και την οπτικοποίηση των μοντέλων. Σε τελική ανάλυση, το Scikit-learn αναδεικνύεται ως μια από τις πιο προσβάσιμες και ισχυρές βιβλιοθήκες μηχανικής μάθησης στην Python, προσφέροντας μια ολοκληρωμένη λύση για την ανάλυση και επεξεργασία δεδομένων [28].

### 4.2.5 Torch

Το Torch είναι μια open source βιβλιοθήκη για τον προγραμματισμό επιστημονικών υπολογισμών, με ιδιαίτερη έμφαση στις εφαρμογές μηχανικής μάθησης και τεχνητής νοημοσύνης. Αν και αρχικά αναπτύχθηκε και χρησιμοποιήθηκε κυρίως για ερευνητικούς σκοπούς, η ευελιξία και η δυναμική του Torch έχουν κάνει τη βιβλιοθήκη αυτή δημοφιλή και στην βιομηχανία, προσφέροντας μια πλατφόρμα με πλούσιες δυνατότητες για την ανάπτυξη καινοτόμων εφαρμογών στο πεδίο της τεχνητής νοημοσύνης. Ένας από τους βασικούς λόγους που καθιστούν το Torch ένα σημαντικό εργαλείο για την ανάπτυξη εφαρμογών μηχανικής μάθησης είναι η δυνατότητά του να χειρίζεται μεγάλους όγκους δεδομένων με εξαιρετική αποδοτικότητα. Η βιβλιοθήκη παρέχει μια σειρά από εργαλεία και βιβλιοθήκες, όπως το Torch.nn, Torch.optim και Torch.autograd, που διευκολύνουν τον σχεδιασμό, την εκπαίδευση και τη βελτιστοποίηση νευρωνικών δικτύων, κάνοντας την επεξεργασία και την ανάλυση δεδομένων πιο αποδοτική και ευέλικτη. Εκτός από τις δυνατότητες επεξεργασίας δεδομένων, το Torch προσφέρει επίσης ένα εκτεταμένο API για την οπτικοποίηση δεδομένων και αποτελεσμάτων, επιτρέποντας στους ερευνητές και τους προγραμματιστές να παρακολουθούν την πρόοδο της εκπαίδευσης των

μοντέλων και να διενεργούν αναλυτική αξιολόγηση των αλγορίθμων τους. Αυτή η δυνατότητα είναι καθοριστική για την ανάπτυξη αποδοτικών και αξιόπιστων εφαρμογών τεχνητής νοημοσύνης. Η ευελιξία του Torch δεν περιορίζεται μόνο στην επεξεργασία και οπτικοποίηση δεδομένων. Η βιβλιοθήκη υποστηρίζει επίσης την εκτέλεση υπολογισμών σε πολλαπλές συσκευές, όπως CPU και GPU, παρέχοντας σημαντικές βελτιώσεις στην ταχύτητα και την αποδοτικότητα των εφαρμογών. Αυτό είναι ιδιαίτερα σημαντικό σε εφαρμογές που απαιτούν την επεξεργασία μεγάλων συνόλων δεδομένων ή τη χρήση πολύπλοκων νευρωνικών δικτύων, όπου η δυνατότητα για γρήγορη εκπαίδευση και επανάληψη είναι κρίσιμη για την επιτυχία του έργου. Εκτός από τις τεχνικές δυνατότητες, ένας από τους λόγους που το Torch έχει κερδίσει την εμπιστοσύνη και την προτίμηση της ερευνητικής κοινότητας και της βιομηχανίας είναι η ενεργή και ανοιχτή κοινότητα χρηστών και συνεισφερόντων. Η συνεχής ανταλλαγή γνώσης, η διάθεση εκπαιδευτικού υλικού και η τακτική προσθήκη νέων λειτουργιών και βελτιώσεων εξασφαλίζουν ότι το Torch παραμένει στην πρωτοπορία των τεχνολογιών μηχανικής μάθησης και τεχνητής νοημοσύνης [29].

## 5. Solution analysis

Παρακάτω αναλύουμε τη λύση και ταυτόχρονα εξηγούμε τις τεχνικές λεπτομέρειες της δουλειάς μας.

Πρώτα μετατρέπουμε τα δεδομένα μας σε ένα Pandas dataframe:

```
data = pd.read_csv("./data/diabetes_prediction_dataset.csv")
```

Παρατηρούμε ότι έχουμε 100000 instances με 8 features ανά instance γιατί τρέχοντας το

```
data.shape
```

μας δίνει (100000, 9)

Στη συνέχεια, τυπώνουμε τις πρώτες 5 γραμμές χρησιμοποιώντας τη παρακάτω εντολή:

```
data.head()
```

Ακολουθεί το output:

| | gender | age | hypertension | heart_disease | smoking_history | bmi | HbA1c_level | blood_glucose_level | diabetes |
|---|---|---|---|---|---|---|---|---|---|
| 0 | Female | 80.0 | 0 | 1 | never | 25.19 | 6.6 | 140 | 0 |
| 1 | Female | 54.0 | 0 | 0 | No Info | 27.32 | 6.6 | 80 | 0 |
| 2 | Male | 28.0 | 0 | 0 | never | 27.32 | 5.7 | 158 | 0 |
| 3 | Female | 36.0 | 0 | 0 | current | 23.45 | 5.0 | 155 | 0 |
| 4 | Male | 76.0 | 1 | 1 | current | 20.14 | 4.8 | 155 | 0 |

*Figure 1 Pandas Dataframe*

Πρέπει να ελέγξουμε ότι δεν έχουμε null values

```
# check if any null value is present
data.isnull().values.any()
```

Το output από τη παραπάνω εντολή είναι False, άρα δεν έχουμε.

Στη συνέχεια, ελέγχουμε το τύπο των δεδομένων μας σε κάθε column:

```
data.info()
```

Παίρνουμε το παρακάτω output:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100000 entries, 0 to 99999
Data columns (total 9 columns):
 #   Column               Non-Null Count   Dtype  
---  ------               --------------   -----  
 0   gender               100000 non-null  object 
 1   age                  100000 non-null  float64
 2   hypertension         100000 non-null  int64  
 3   heart_disease        100000 non-null  int64  
 4   smoking_history      100000 non-null  object 
 5   bmi                  100000 non-null  float64
 6   HbA1c_level          100000 non-null  float64
 7   blood_glucose_level  100000 non-null  int64  
 8   diabetes             100000 non-null  int64  
dtypes: float64(3), int64(4), object(2)
memory usage: 6.9+ MB
```

*Figure 2 Data type*

## 5.1 Correlation

Η συσχέτιση στη στατιστική είναι ένα μέτρο που δείχνει πόσο στενά συνδέονται δύο μεταβλητές μεταξύ τους. Ο συντελεστής συσχέτισης μπορεί να πάρει τιμές από -1 έως +1. Μια τιμή +1 σημαίνει τέλεια θετική συσχέτιση (όταν η μία αυξάνεται, αυξάνεται και η άλλη), ένας συντελεστής -1 σημαίνει τέλεια αρνητική συσχέτιση (όταν η μία αυξάνεται, η άλλη μειώνεται), ενώ η τιμή 0 δείχνει καμία συσχέτιση.

- Η καταγραφή της συσχέτισης είναι σημαντική για διάφορους λόγους:
- Ανακάλυψη Σχέσεων: Βοηθά στην εύρεση πιθανών σχέσεων μεταξύ διαφορετικών μεταβλητών.
- Προειδοποίηση για Πολυσυσχετισμό: Στη μηχανική μάθηση, η υψηλή συσχέτιση μεταξύ ανεξάρτητων μεταβλητών (πολυσυσχετισμός) μπορεί να δημιουργήσει προβλήματα κατά την εκπαίδευση μοντέλων.
- Ενίσχυση Κατανόησης Δεδομένων: Βοηθά στην κατανόηση της δομής και της φύσης των δεδομένων.

Η διαγραμματική απεικόνιση της συσχέτισης μέσω θερμοχαρτών (heatmap) είναι ένας άμεσος και οπτικός τρόπος για να κατανοηθούν αυτές οι σχέσεις. Οι θερμοχάρτες δείχνουν με χρωματικές αποχρώσεις το επίπεδο της συσχέτισης, κάνοντας την ανάλυση των δεδομένων πιο εύκολη και

διαισθητική [30].

Ακολουθεί ο σχετικός κώδικας:

```
#get correlations of each features in dataset
corrmat = data.corr()
top_corr_features = corrmat.index
plt.figure(figsize=(20,20))
#plot heat map
g=sns.heatmap(data[top_corr_features].corr(),annot=True,cmap="RdYlGn")
```

Kai to heatmap

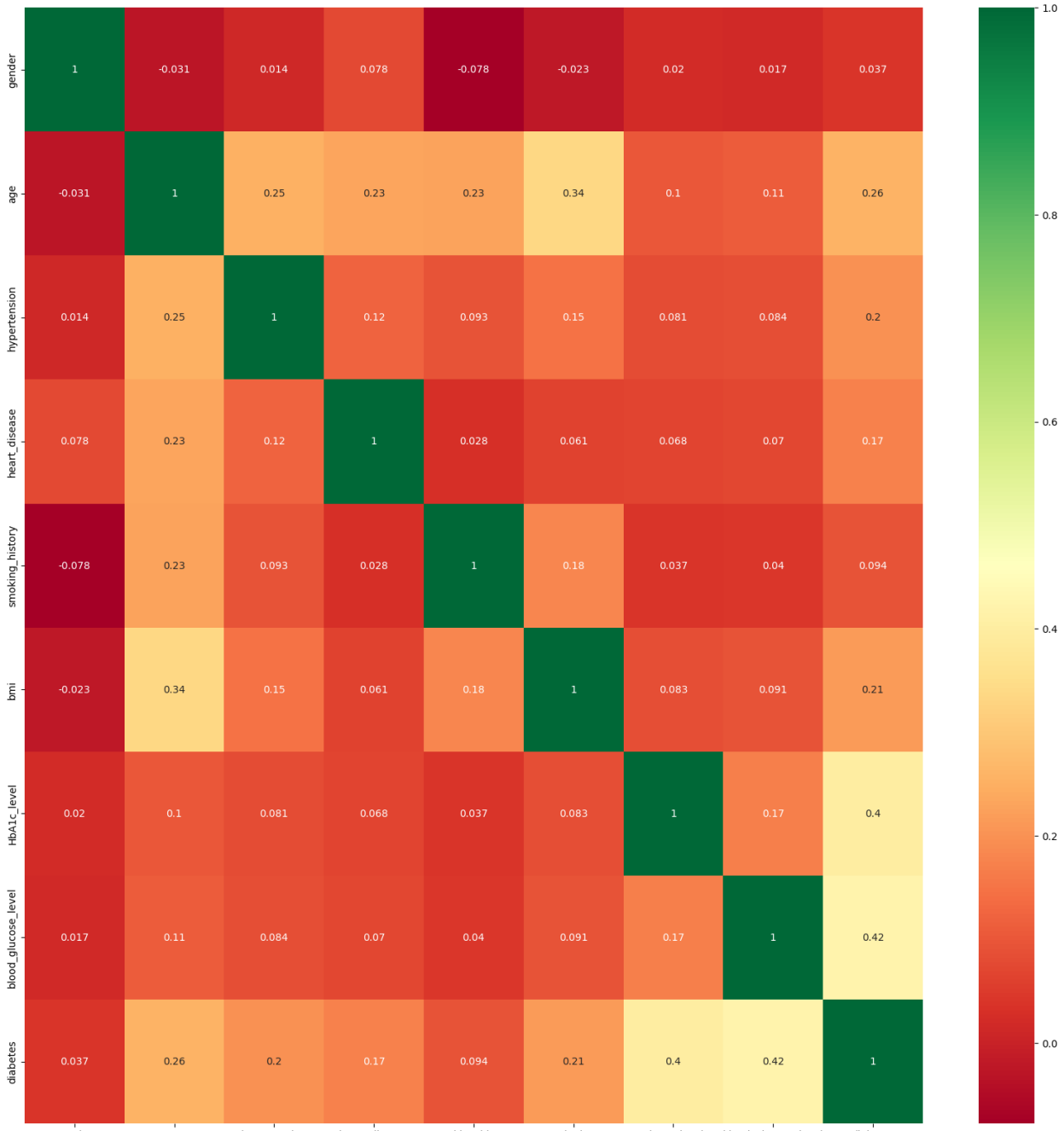


*Figure 3 Heatmap*

Στη συνέχεια με το command

```
data.corr()
```

βλέπουμε το correlation

| | age | hypertension | heart_disease | bmi | HbA1c_level | blood_glucose_level | diabetes |
|---|---|---|---|---|---|---|---|
| age | 1.000000 | 0.251171 | 0.233354 | 0.337396 | 0.101354 | 0.110672 | 0.258008 |
| hypertension | 0.251171 | 1.000000 | 0.121262 | 0.147666 | 0.080939 | 0.084429 | 0.197823 |
| heart_disease | 0.233354 | 0.121262 | 1.000000 | 0.061198 | 0.067589 | 0.070066 | 0.171727 |
| bmi | 0.337396 | 0.147666 | 0.061198 | 1.000000 | 0.082997 | 0.091261 | 0.214357 |
| HbA1c_level | 0.101354 | 0.080939 | 0.067589 | 0.082997 | 1.000000 | 0.166733 | 0.400660 |
| blood_glucose_level | 0.110672 | 0.084429 | 0.070066 | 0.091261 | 0.166733 | 1.000000 | 0.419558 |
| diabetes | 0.258008 | 0.197823 | 0.171727 | 0.214357 | 0.400660 | 0.419558 | 1.000000 |

*Figure 4 Correlation*

## 5.2 Categorical to numerical values

Categorical to Numerical values

Η μετατροπή categorical values σε numerical values είναι μια σημαντική διαδικασία στην επεξεργασία δεδομένων για την ανάλυση και τη μηχανική μάθηση για τους εξής λόγους:

Συμβατότητα με Αλγορίθμους: Οι περισσότεροι αλγόριθμοι μηχανικής μάθησης λειτουργούν μόνο με αριθμητικά δεδομένα. Η μετατροπή κατηγορικών δεδομένων σε αριθμητική μορφή επιτρέπει τη χρήση τους σε αυτούς τους αλγορίθμους.

Επεξεργασία Πληροφορίας: Αριθμητικές μεταβλητές μπορούν να παρέχουν πιο ακριβείς πληροφορίες για σχέσεις και διαφορές μεταξύ των κατηγοριών.

Βελτίωση Απόδοσης: Η μετατροπή αυτή μπορεί να βοηθήσει στη βελτίωση της απόδοσης των μοντέλων, καθώς τα αριθμητικά δεδομένα συχνά επεξεργάζονται πιο αποτελεσματικά.

Οπτικοποίηση Δεδομένων: Η αριθμητική αναπαράσταση κάνει ευκολότερη την οπτικοποίηση των δεδομένων, επιτρέποντας την καλύτερη ανάλυση και κατανόηση των σχέσεων ανάμεσα στις μεταβλητές.

Ακουλουθεί ο σχετικός κώδικας

```
le = LabelEncoder()
data['gender'] = le.fit_transform(data['gender'])
data['smoking_history'] = le.fit_transform(data['smoking_history'])
```

Και με την εντολή

```
data.head()
```

βλέπουμε το εξής output:

| | gender | age | hypertension | heart_disease | smoking_history | bmi | HbA1c_level | blood_glucose_level | diabetes |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 80.0 | 0 | 1 | 4 | 25.19 | 6.6 | 140 | 0 |
| 1 | 0 | 54.0 | 0 | 0 | 0 | 27.32 | 6.6 | 80 | 0 |
| 2 | 1 | 28.0 | 0 | 0 | 4 | 27.32 | 5.7 | 158 | 0 |
| 3 | 0 | 36.0 | 0 | 0 | 1 | 23.45 | 5.0 | 155 | 0 |
| 4 | 1 | 76.0 | 1 | 1 | 1 | 20.14 | 4.8 | 155 | 0 |

*Figure 5 Numerical representation*

### 5.3 Imbalanced dataset

Imbalanced classes: Το σύνολο των δεδομένων μας είναι έντονα imbalanced προς την κατηγορία 'Όχι'. Αυτό το imbalance μπορεί να επηρεάσει την απόδοση των μοντέλων μας στην συνέχεια, καθώς ενδέχεται να είναι "προκατειλημμένα" προς την πρόβλεψη της πλειοψηφικής κλάσης ('Όχι' σε αυτή την περίπτωση).

Δυσκολίες στην εκπαίδευση των μοντέλων: Τα μοντέλα που κάνουμε train σε αυτό το dataset μπορεί να μην αποδίδουν καλά στη σωστή πρόβλεψη της κλάσης με τα λιγότερα instances ('Ναί'). Είναι σημαντικό να εφαρμοστούν τεχνικές όπως η υπερδειγματοληψία της κλάσης μειονότητας, η υποδειγματοληψία της πλειοψηφικής κλάσης ή η χρήση αλγορίθμων που είναι από φύση τους πιο ανθεκτικοί σε ανισορροπημένα δεδομένα.

Μετρικές Αξιολόγησης: Το accuracy δεν είναι τόσο αξιόπιστη μετρική για την αξιολόγηση της απόδοσης ενός μοντέλου που έχει εκπαιδευτεί σε αυτό το σύνολο δεδομένων. Μετρικές που παρέχουν πιο εις βάθος πληροφορίες, όπως το recall, precision, f1 score είναι πιο κατάλληλες για μη ισορροπημένα σύνολα δεδομένων.

```
diabetes_counts = data['diabetes'].value_counts()

# Create a bar plot
sns.barplot(x=diabetes_counts.index, y=diabetes_counts.values)

# Adding labels and title for clarity
plt.xlabel('Diabetes')
plt.ylabel('Number of Instances')
plt.title('Distribution of Diabetes in the Dataset')
#add_watermark()
# Display the plot
plt.show()
```

Τρέχοντας τον από πάνω κώδικα παίρνουμε το εξής plot

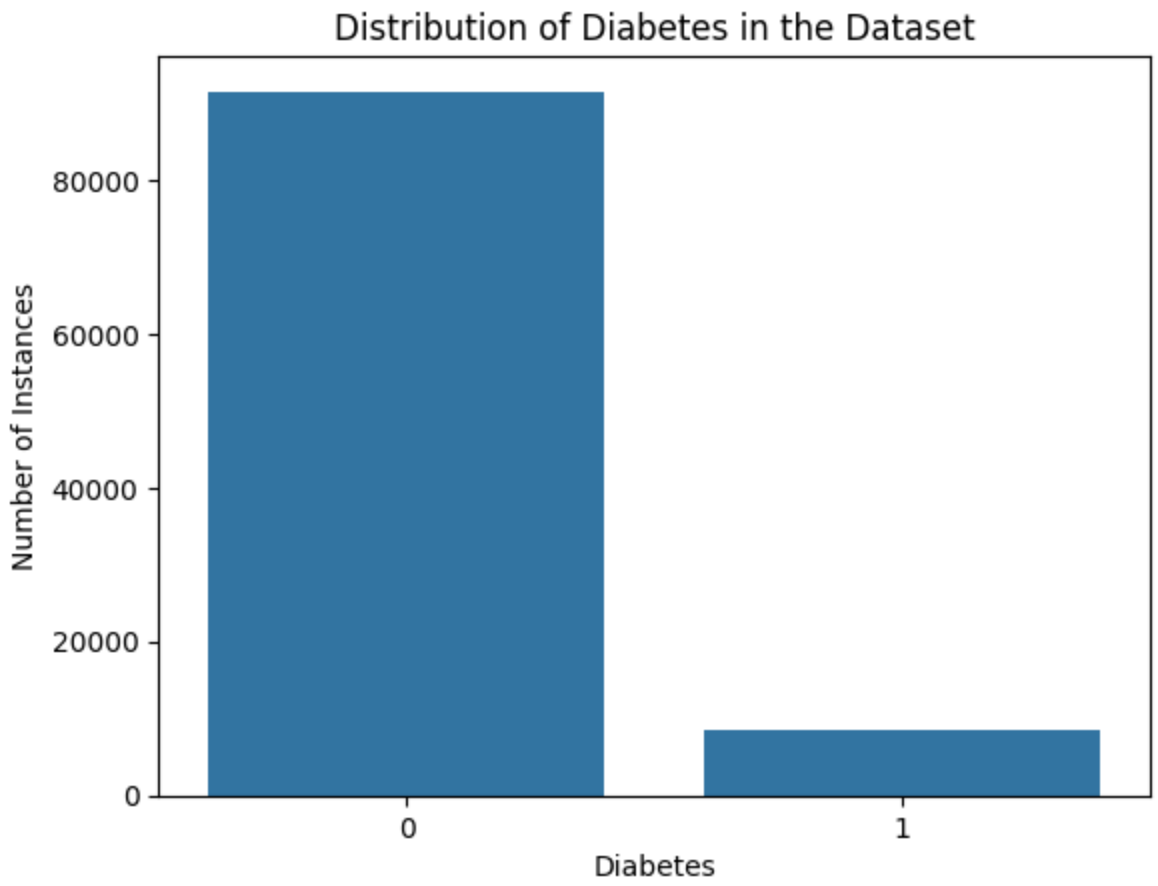


*Figure 6 Distribution of diabetes in the Dataset*

Και με τη παρακάτω εντολή

```
data['diabetes'].value_counts()
```

παίρνουμε:

```
0    91500
1     8500
Name: diabetes, dtype: int64
```

*Figure 7 Value counts*

## .3.1 Histogram

Κάνουμε plot ένα histogram για κάθε column του dataset μας για να δούμε τι τιμές περιέχει κάθε column και αν είναι normally distributed **[31]**

Χρησιμοποιούμε τον παρακάτω κώδικα:

```
data.hist(bins=10,figsize=(10,10))
plt.show()
```

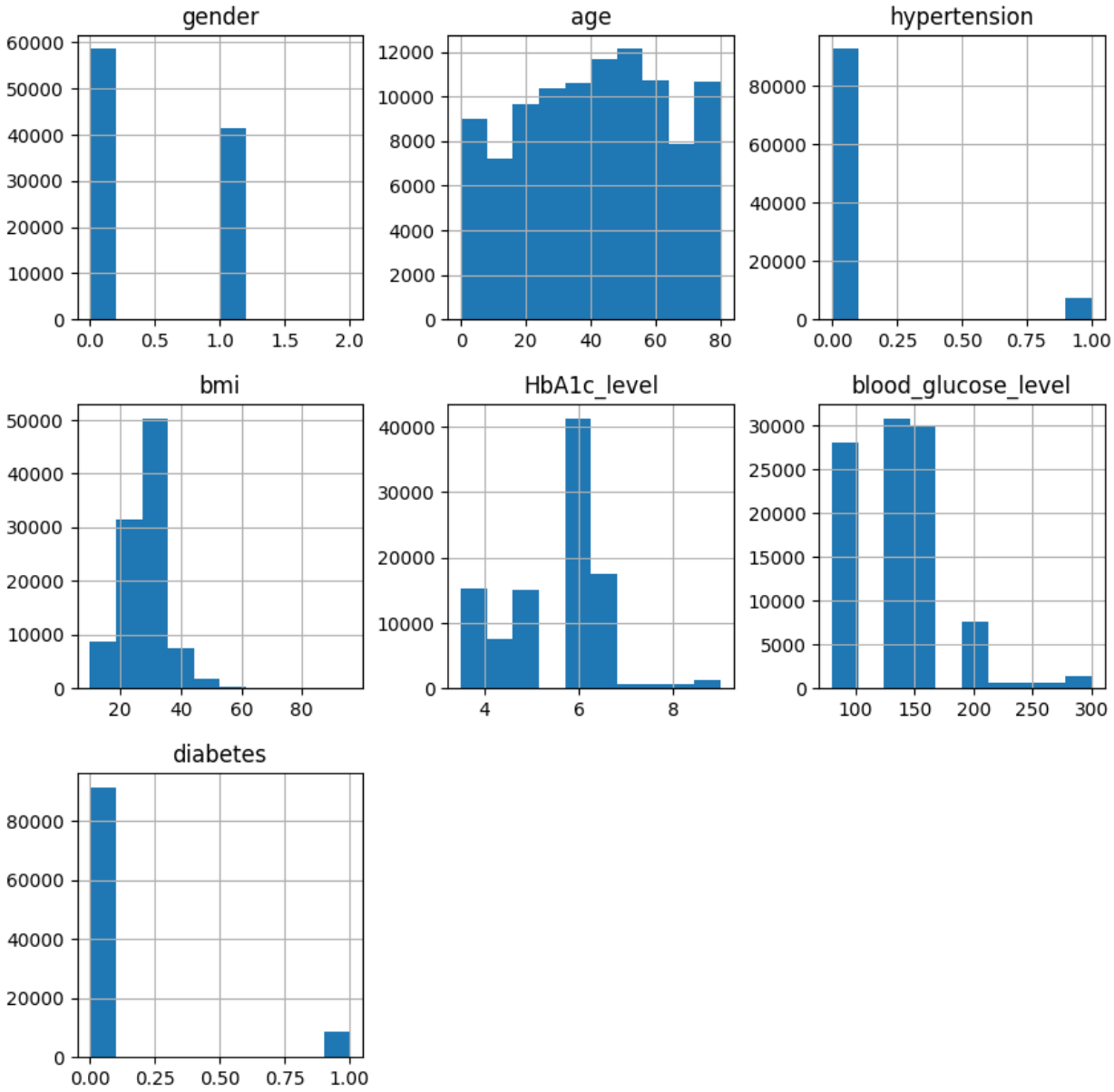


*Figure 8 Histograms for dataset columns*

### 5.3.2 Outliers

Τα outliers είναι ασυνήθιστες τιμές στο σύνολο των δεδομένων σας, και μπορούν να παραποιήσουν τις στατιστικές αναλύσεις και να παραβιάσουν τις υποθέσεις τους. Επομένως, είναι άκρως σημαντικό να αντιμετωπιστούν. Σε αυτή την περίπτωση, η αφαίρεση των ακραίων τιμών μπορεί να προκαλέσει απώλεια δεδομένων, οπότε πρέπει να αντιμετωπιστούν χρησιμοποιώντας διάφορες τεχνικές κλιμάκωσης και μετασχηματισμού.

Χρησιμοποιούμε τον παρακάτω κώδικα για τη σχεδίαση των data μας:

```
plt.figure(figsize=(16,12))
sns.set_style(style='whitegrid')
# First row
plt.subplot(2,3,1)
sns.boxplot(x='gender', data=data)
plt.subplot(2,3,2)
sns.boxplot(x='age', data=data)
plt.subplot(2,3,3)
sns.boxplot(x='hypertension', data=data)

# Second row
plt.subplot(2,3,4)
sns.boxplot(x='bmi', data=data)
plt.subplot(2,3,5)
sns.boxplot(x='HbA1c_level', data=data)
plt.subplot(2,3,6)
sns.boxplot(x='blood_glucose_level', data=data)
#add_watermark()
plt.show()
```

Και παίρνουμε το παρακάτω output:

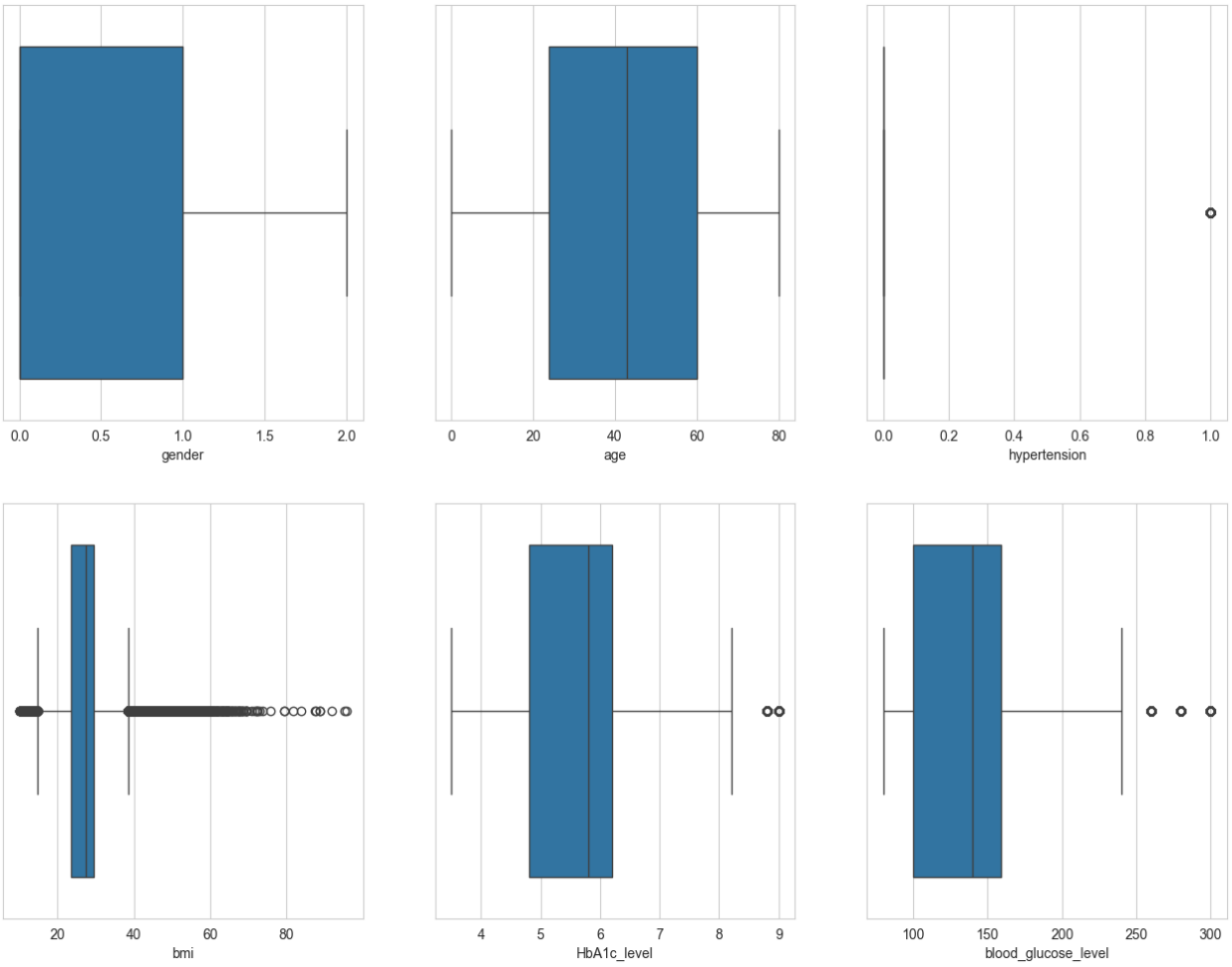


*Figure 9*

### 5.3.3 Feature selection [32]

Από τον παραπάνω correlation heatmap παρατηρούμε πως τα columns blood_glucose_level, HbA1c_level, bmi, age συμβάλλουν περισσότερο στην πρόβλεψη, οπότε θα κρατήσουμε μόνο αυτά τα columns στο dataframe μας.

Το correlation heatmap βοηθά να ανακαλύψουμε τη σχέση μεταξύ δύο ποσοτήτων. Μας δίνει το μέτρο της δύναμης της συσχέτισης μεταξύ δύο μεταβλητών. Αναλόγως πόσο πολύ συσχετίζονται δύο μεταβλητές η τιμή συσχέτισης μπορεί να είναι από από -1 έως 1. 1 σημαίνει ότι είναι αρρήδητα συσχετιζόμενες και 0 σημαίνει καμία συσχέτιση.

Το feature selection είναι σημαντικό στην μηχανική μάθηση, καθώς βοηθά στην αποφυγή του φαινομένου της "κατάρας της διαστατικότητας", όπου ο υπερβολικός αριθμός χαρακτηριστικών μπορεί να οδηγήσει σε overfitting των μοντέλων και να καταστήσει πιο δύσκολη την ερμηνεία των αποτελεσμάτων. Επιπλέον, η επιλογή χαρακτηριστικών μπορεί να βελτιώσει την απόδοση των μοντέλων με τη μείωση του χρόνου εκπαίδευσης και την αποφυγή του θορύβου από τα λιγότερο σημαντικά χαρακτηριστικά.

Χρησιμοποιώντας το παρακάτω κώδικα ζωγραφίζουμε τα αποτελέσματά μας

```
data = data[['gender', 'age', 'hypertension', 'bmi', 'HbA1c_level', 'blood_glucose_level', 'diabetes']]
data.head()
```

| | gender | age | hypertension | bmi | HbA1c_level | blood_glucose_level | diabetes |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 80.0 | 0 | 25.19 | 6.6 | 140 | 0 |
| 1 | 0 | 54.0 | 0 | 27.32 | 6.6 | 80 | 0 |
| 2 | 1 | 28.0 | 0 | 27.32 | 5.7 | 158 | 0 |
| 3 | 0 | 36.0 | 0 | 23.45 | 5.0 | 155 | 0 |
| 4 | 1 | 76.0 | 1 | 20.14 | 4.8 | 155 | 0 |

*Figure 10 Results*

### 5.3.4 Undersample of the majority class

Στην συνέχεια θα πραγματοποιήσουμε undersample του majority class (No) ώστε να μετριάσουμε την "προκατάληψη" του μοντέλου μας ως προς τις προβλέψεις.
Αρχικά χωρίζουμε το dataframe μας

```
df_majority = data[data.diabetes == 0]
df_minority = data[data.diabetes == 1]
```

Στην συνέχεια κρατάμε τόσα instances από το majority class όσα υπάρχουν και στο minority class

```
df_majority_undersampled = resample(df_majority,
                                    replace=False,    # sample without replacement
                                    n_samples=len(df_minority),     # to match minority class
                                    random_state=123) # reproducible results
```

Ακολούθως ενώνουμε ξανά τα 2 dataframes

```
df_undersampled = pd.concat([df_majority_undersampled, df_minority])
```

Τέλος κάνουμε shuffle τα instances ώστε να είναι ανακατεμένα.

```
df_undersampled = df_undersampled.sample(frac=1, random_state=123).reset_index(drop=True)
df_undersampled.head()
```

To output table:

| | gender | age | hypertension | bmi | HbA1c_level | blood_glucose_level | diabetes |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 80.0 | 0 | 25.19 | 6.6 | 140 | 0 |
| 1 | 0 | 54.0 | 0 | 27.32 | 6.6 | 80 | 0 |
| 2 | 1 | 28.0 | 0 | 27.32 | 5.7 | 158 | 0 |
| 3 | 0 | 36.0 | 0 | 23.45 | 5.0 | 155 | 0 |
| 4 | 1 | 76.0 | 1 | 20.14 | 4.8 | 155 | 0 |

*Figure 11 Output table*

## 5.4 Feature scaling

### 5.4.1 Feature Scaling

Αλγόριθμοι μηχανικής μάθησης όπως οι linear regression, logistic regression, neural networks, PCA (principal component analysis) κ.λ.π, οι οποίοι χρησιμοποιούν gradient descent σαν optimization technique αποδίδουν καλύτερα με scaled data. Για παράδειγμα ας παρατηρήσουμε την formula για τον gradient descent algorithm παρακάτω **[33]**:

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \left( h_\theta(x^{(i)}) - y^{(i)} \right) x_j^{(i)}$$

Η ενσωμάτωση της τιμής του feature X στην εξίσωση θα επιδρά στο μέγεθος του βήματος της μεθόδου gradient descent. Οι διαφορές στα εύρη των χαρακτηριστικών θα οδηγήσουν σε διαφορετικά μεγέθη βημάτων για κάθε χαρακτηριστικό. Για να εξασφαλιστεί ότι ο gradient descent κινείται ομαλά προς το σημείο ελαχίστων και ότι η ενημέρωση των βημάτων γίνεται με συνεπή τρόπο για όλα τα χαρακτηριστικά, προβαίνουμε σε κλιμάκωση των δεδομένων πριν την εισαγωγή τους στο μοντέλο.

### 5.4.2 Normalization

Το Normalization είναι ένα data preprocessing technique που χρησιμοποιείται για να κάνουμε adjust τις τιμές των χαρακτηριστικών (features) σε ένα dataset σε ένα common scale. Αυτό γίνετε για να μειώσουμε την επίδραση των διαφορετικών scales που μπορεί να έχουν οι τιμές των features. Στο Normalization οι τιμές των features γίνονται rescaled στο εύρος 0-1. Αυτή η τεχνική είναι επίσης γνωστή ως Min-Max Scaling **[34]**

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

Σε αυτή την περίπτωση, το Xmax και το Xmin αποτελούν τις μέγιστες και ελάχιστες αξίες του εκάστοτε feature αντίστοιχα.

Όταν η αξία του X είναι η ελάχιστη στη στήλη, τότε ο αριθμητής είναι μηδέν, επομένως το X’ είναι επίσης μηδέν.
Αντίθετα, όταν η αξία του X είναι η μέγιστη στη στήλη, ο αριθμητής είναι ίσος με τον παρονομαστή, κατά συνέπεια η αξία του X’ είναι 1.
Εάν η αξία του X βρίσκεται κάπου ανάμεσα στην ελάχιστη και τη μέγιστη αξία, τότε η αξία του X’ κυμαίνεται μεταξύ 0 και 1.

### 5.4.3 Standarization

Το Standarization αποτελεί μία εναλλακτική μέθοδο κλιμάκωσης, όπου οι τιμές ευθυγραμμίζονται γύρω από τον μέσο όρο, διαθέτοντας μοναδιαία τυπική απόκλιση. Αυτό οδηγεί στον μετασχηματισμό του μέσου όρου του χαρακτηριστικού σε μηδέν, ενώ η συνακόλουθη κατανομή έχει τυπική απόκλιση ίση με τη μονάδα.

$$x' = \frac{x-\mu}{\sigma}$$

Σε αυτόν τον τύπο:

x: είναι η αρχική τιμή του χαρακτηριστικού.
μ: είναι ο μέσος όρος του χαρακτηριστικού σε όλα τα δείγματα.
σ: είναι η τυπική απόκλιση του χαρακτηριστικού.

## 5.5 Logistic Regression

Η Λογιστική παλινδρόμηση αναπτύχθηκε από τον στατιστικό David Cox το 1958. Η Λογιστική παλινδρόμηση χρησιμοποιείται σε διάφορους τομείς, συμπεριλαμβανομένης της μηχανικής μάθησης, των περισσότερων ιατρικών πεδίων, των στατιστικών και των κοινωνικών επιστημών. Χρησιμοποιείται επίσης σε εφαρμογές μάρκετινγκ όπως η πρόβλεψη της τάσης του πελάτη να αγοράσει ένα προϊόν ή να σταματήσει μια συνδρομή κ.λπ. Οι ρίζες της εξαπλώθηκαν πολύ πίσω στις αρχές του 19ου αιώνα ,η επιβίωση του όρου logistics και η ευρεία εφαρμογή της εξίσωσης της Λογιστικής Παλινδρόμησης καθορίστηκαν αποφασιστικά από τις προσωπικές ιστορίες και τις ατομικές ενέργειες μερικών μελετητών. Η εξίσωση της Λογιστικής Παλινδρόμησης ανακαλύφθηκε εξ'αρχής το 1920 από τον Pearl και Reed και χρησιμοποιήθηκε σε μια μελέτη σχετικά με την ανάπτυξη του πληθυσμού, στις Ηνωμένες Πολιτείες της Αμερικής. Ωστόσο ,ο όρος Λογιστική δεν είχε ξαναχρησιμοποιηθεί πριν από τον Βέλγο μαθηματικό, τον Verhulst, όπου και την όρισε, λίγο μετά από το άρθρο των Pearl και Reed το 1920.

**Θεωρία Λογιστικής Παλινδρόμησης**
Η μέθοδος της λογιστικής παλινδρόμησης χρησιμεύει στο να αναπτύξουμε σχέση μίας δίτιμης εξαρτημένης τυχαίας μεταβλητής και συνεχών ή διακριτών ανεξάρτητων μεταβλητών. Ουσιαστικά η μέθοδος αυτή γενικεύει τα γραμμικά μοντέλα, έτσι ώστε η εξαρτημένη μεταβλητή να ακολουθεί την εκθετική οικογένεια κατανομών.

Η πιο διαδεδομένη, έκφραση της εξίσωσης της Λογιστικής Παλινδρόμησης είναι [35]:

$$\log\left(\frac{P(y=1|x)}{1-P(y=1|x)}\right) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n$$

όπου β0 είναι είναι το ύψος της κλίσης της γραμμής παλινδρόμησης, β1,……β$k$ είναι συντελεστές παλινδρόμησης(regression coefficients) καθένας των οποίων εκφράζει το μέγεθος συνεισφοράς της αντίστοιχης μεταβλητής. Θετική τιμή του συντελεστή δηλώνει ότι η πεξηγηματική μεταβλητή αυξάνει την πιθανότητα της επιτυχημένης έκβασης (να συμβεί δηλαδή το γεγονός), αρνητική τιμή σημαίνει ότι η μεταβλητή μειώνει την πιθανότητα αυτής της έκβασης. Υψηλή τιμή του συντελεστή σημαίνει ότι η ανεξάρτητη μεταβλητή επηρεάζει πολύ ισχυρά την πιθανότητα να συμβεί το γεγονός ή μη, ενώ χαμηλή τιμή δηλώνει μικρή επίδραση της ανεξάρτητης μεταβλητής στην πιθανότητα εμφάνισης της ανάλογης έκβασης.

Το δεξί μέρος της εξίσωσης δημιουργείται από ένα γραμμικό συνδυασμό των ανεξάρτητων μεταβλητών που συμμετέχουν στο μοντέλο της παλινδρόμησης. Το αριστερό μέρος περιέχει τις τιμές της εξαρτημένης μεταβλητής με την μορφή του λογαρίθμου των odd δηλαδή, του λογαρίθμου της σχέσης: odds = prob/(1-prob) ή με p που είναι η πιθανότητα να εμφανιστεί το γεγονός και 1-p

η πιθανότητα να μη συμβεί, τότε ο λόγος των πιθανοτήτων θα είναι p/(1-p). Το odds εναλλακτικά ονομάζεται logit και ο όρος Prob εκφράζει την πιθανότητα να συμβεί το γεγονός που έχει ορισθεί σαν επιτυχία του πειράματος. Οι συντελεστές των ανεξάρτητων μεταβλητών στην εξίσωση της παλινδρόμησης εκτιμώνται βάση της μεθόδου Μέγιστης Πιθανοφανείας όπου βάση της μεθόδου αυτής η τιμή των συντελεστών των ανεξάρτητων μεταβλητών είναι αυτή που κάνει τις παρατηρήθησες τιμές της εξαρτημένης μεταβλητής πιο πιθανές, βάση του σετ των ανεξάρτητων μεταβλητών. Ο εκτιμώμενος συντελεστής για κάθε ανεξάρτητη μεταβλητή εκφράζει τη μεταβολή του λογαρίθμου του λόγου p/ (1-p) για κάθε μονάδα μεταβολής της αντίστοιχης ανεξάρτητης μεταβλητής, ενώ οι λοιπές παραμένουν σταθερές .

$P(y = 1|x)$
είναι η πιθανότητα η έξοδος y να είναι 1 δεδομένης της εισόδου x.
$\log\left(\frac{P(y=1|x)}{1-P(y=1|x)}\right)$
είναι ο λογάριθμος των raion των πιθανοτήτων, συμβολίζει το log-odds της εξόδου.

$\beta_0, \beta_1, \beta_2, \ldots, \beta_n$
είναι τα coefficients του μοντέλου όπου β0 είναι το intercept, και β1, …, βn είναι τα coefficients των αντίστοιχων features **[36]**.

Αρχικά ας κάνουμε standarization των δεδομένων μας. Επιλέξαμε να κάνουμε standarize τα data μας διότι είναι μια μέθοδος ποιο ανθεκτική στα outliers.

```
columns_to_scale = ['age', 'bmi', 'HbA1c_level', 'blood_glucose_level']
sc = StandardScaler()
df_undersampled[columns_to_scale] = sc.fit_transform(df_undersampled[columns_to_scale])
```

Διαχωρίζουμε τα data μας σε training και validation sets

```
X_train, X_test, y_train, y_test = train_test_split(df_undersampled[['gender', 'age', 'hypertension',
'bmi', 'HbA1c_level', 'blood_glucose_level']],
                                    df_undersampled[['diabetes']], shuffle=False, test_size=0.40)
```

### 5.5.1 Metrics

**Accuracy**: Μετράει το ποσοστό των σωστών προβλέψεων από τις συνολικές προβλέψεις(Number of Correct Predictions) / (Total Number of Predictions)

**F1**:Το F1 score είναι το αρμονικό μέσο του Precision και του Recall
**2 x ((Precision x Recall)/(Precision+Recall))**

**Recall**:Το Recall μετράει την ικανότητα του μοντέλου να βρίσκει τα relevant cases (positive samples - instanses που είναι classified as diabetic στην περίπτωση μας) σε ένα dataset **TP/(TP+FN)**

**Precision**: Το Precision μετράει το ποσοστό των positive cases που είναι στην πραγματικότητα σωστά
**TP/(TP+FP)**

## Hyperparameters για την Λογιστική Παλινδρόμηση

**max_iter:** Το max_iter hyperparameter στο LogisticRegression καθορίζει τον μέγιστο αριθμό επαναλήψεων που θα εκτελέσει ο αλγόριθμος κατά τη διαδικασία της βελτιστοποίησης. Στη μηχανική μάθηση, ιδιαίτερα σε μοντέλα που χρησιμοποιούν τεχνικές βελτιστοποίησης βασισμένες στην κλίση όπως η λογιστική παλινδρόμηση, ο αριθμός των επαναλήψεων αποτελεί έναν κρίσιμο παράγοντα που επηρεάζει και την απόδοση και το υπολογιστικό κόστος του μοντέλου. Αν ο αριθμός των επαναλήψεων είναι πολύ χαμηλός, το μοντέλο μπορεί να μη συγκλίνει σε μια λύση, οδηγώντας σε underfitting. Αντίθετα, πάρα πολλές επαναλήψεις μπορεί να οδηγήσουν σε overfitting, ειδικά αν τα δεδομένα εκπαίδευσης περιέχουν θόρυβο **[37]**

**solver:** Καθορίζει τον αλγόριθμο που θα χρησιμοποιηθεί στο πρόβλημα βελτιστοποίησης <br>

Οι lbfgs, newton-cg και sag είναι καταλληλότεροι για μεγάλα σετ δεδομένων καθώς διαχειρίζονται την πολυομική απώλεια και παράγουν πιο αξιόπιστα αποτελέσματα για multilabel προβλήματα.
Ο liblinear είναι μια καλή επιλογή για μικρά σετ δεδομένων και υποστηρίζει και τις κανονικοποιήσεις l1 και l2.

**penalty:**Καθορίζει τη νόρμα που χρησιμοποιείται στο penalty των βαρών. Η κανονικοποίηση εφαρμόζεται στη συνάρτηση κόστους για να αποφευχθεί το overfitting, αποθαρρύνοντας τα περίπλοκα μοντέλα

Το l1 μπορεί να οδηγήσει σε λιτά μοντέλα με συντελεστές που μπορούν να είναι ακριβώς μηδέν. Είναι χρήσιμο για την επιλογή χαρακτηριστικών καθώς τείνει να μειώνει τα coefficients για λιγότερο σημαντικά χαρακτηριστικά σε μηδέν.
Το l2 τείνει να μειώνει τα coefficients ομοιόμορφα αλλά συνήθως δεν τα θέτει μηδέν. Αυτό είναι συχνά πιο κατάλληλο για προβλήματα με πολλά συσχετιζόμενα χαρακτηριστικά.

Το elasticnet αποτελεί μια συνδυασμένη προσέγγιση που ενσωματώνει το l1 και το l2. Αυτή η μέθοδος ενθαρρύνει τόσο τη λιτότητα του μοντέλου όσο και την ομοιόμορφη μείωση των συντελεστών. Με τον συνδυασμό των πλεονεκτημάτων τόσο του l1 όσο και του l2, το elasticnet είναι ιδιαίτερα χρήσιμο σε σενάρια όπου υπάρχει μεγάλος αριθμός χαρακτηριστικών, και ειδικά όταν αυτά τα χαρακτηριστικά είναι συσχετιζόμενα. Επιτρέπει τη διατήρηση των σημαντικών χαρακτηριστικών στο μοντέλο, ενώ παράλληλα μειώνει την επιρροή των λιγότερο σημαντικών, προσφέροντας ένα καλό συμβιβασμό μεταξύ της επιλογής χαρακτηριστικών και της αποφυγής του overfitting.

**C:**Το C hyperparameter διαδραματίζει κρίσιμο ρόλο στον έλεγχο της ισορροπίας μεταξύ της επίτευξης χαμηλού σφάλματος στα δεδομένα εκπαίδευσης και της διατήρησης ενός μοντέλου που γενικεύει καλά σε καινούργια δεδομένα. Αυτή η υπερπαράμετρος είναι άμεσα συνδεδεμένη με την ισχύ της κανονικοποίησης, με την τιμή του να αντιστοιχεί στο αντίστροφο της ισχύος κανονικοποίησης.

**Χαμηλές τιμές του C:**

Αυξάνουν την ισχύ της κανονικοποίησης, δημιουργώντας απλούστερα μοντέλα που μπορεί να

κάνουν underfitting στα δεδομένα εκπαίδευσης. Αυτό συμβαίνει επειδή η συνάρτηση βελτιστοποίησης θα δώσει προτεραιότητα στην απλότητα (μικρότεροι συντελεστές, ανάλογα με τον κανόνα που χρησιμοποιείται στο penalty) του μοντέλου έναντι της τέλειας προσαρμογής στα δεδομένα εκπαίδευσης.
Χρήσιμες όταν πιστεύουμε ότι τα δεδομένα είναι πολύ θορυβώδη και πρέπει να αποτρέψουμε το μοντέλο από το να μάθει τον θόρυβο.

**Υψηλές τιμές του C**:

Μειώνουν την ισχύ της κανονικοποίησης, επιτρέποντας στα μοντέλα να γίνουν πιο περίπλοκα και να προσαρμόζονται καλύτερα στα δεδομένα εκπαίδευσης. Αυτό μπορεί να οδηγήσει σε overfitting αν το μοντέλο αρχίσει να μαθαίνει τον θόρυβο και τις λεπτομερείς διακυμάνσεις μέσα στα δεδομένα εκπαίδευσης. Χρήσιμες όταν το μοντέλο υποφέρει από υψηλό bias, δηλαδή είναι πολύ απλό και δεν αποτυπώνει καλά τις υποκείμενες τάσεις.

**multi_class:**

**auto:** Η επιλογή auto αυτόματα επιλέγει μεταξύ ovr και multinomial με βάση τα δεδομένα και τις άλλες υπερπαραμέτρους. Εάν το μοντέλο εκπαιδεύεται σε ένα δυαδικό πρόβλημα κατηγοριοποίησης ή εάν επιλεγεί l1 penalty, τότε από προεπιλογή χρησιμοποιείται η ovr. Διαφορετικά, χρησιμοποιείται η multinomial. Αυτή η επιλογή είναι καλή για γενική χρήση, καθώς αφήνει την απόφαση στην εσωτερική λογική του αλγορίθμου.

**ovr (One-vs-Rest):** Η επιλογή ovr σημαίνει ότι για multiclass κατηγοριοποίηση, το μοντέλο θα εκπαιδευτεί να διαχωρίζει κάθε κατηγορία από όλες τις άλλες. Για παράδειγμα, αν υπάρχουν τρεις κατηγορίες, θα εκπαιδευτούν τρία διαφορετικά μοντέλα: ένα για τη διαχωρισμό της πρώτης κατηγορίας από τις άλλες δύο, ένα για τη δεύτερη, και ένα για την τρίτη. Αυτή η μέθοδος είναι απλή και αποτελεσματική, αλλά μπορεί να μην είναι η καλύτερη για περίπλοκα προβλήματα με πολλές κατηγορίες που είναι στενά συνδεδεμένες.

**multinomial**: Η multinomial επιλογή εφαρμόζει ένα πιο πολύπλοκο μοντέλο που εκπαιδεύεται να προβλέπει τις πιθανότητες μεταξύ όλων των κατηγοριών ταυτόχρονα, αντί να τις χειρίζεται ανεξάρτητα. Αυτό είναι πιο κατάλληλο για προβλήματα όπου οι κατηγορίες είναι αλληλένδετες ή όταν τα δεδομένα δεν είναι balanced. Η multinomial προσέγγιση τείνει να δίνει καλύτερα αποτελέσματα σε περίπλοκα multiclass προβλήματα κατηγοριοποίησης.

Το function για plotting

```
def autolabel(rects, ax):
    for rect in rects:
        height = rect.get_height()
        ax.annotate('{}'.format(round(height, 2)),
                    xy=(rect.get_x() + rect.get_width() / 2, height),
                    xytext=(0, 3),  # 3 points vertical offset
                    textcoords="offset points",
                    ha='center', va='bottom')
```

```
solvers = ['lbfgs', 'liblinear', 'liblinear', 'newton-cg', 'newton-cholesky', 'sag']
```

```
penalty = ['l1', 'l2', 'elasticnet']
C = [0.0001, 0.001, 0.01, 0.1, 1, 10, 100]
max_iter = list(range(1000, 10000, 1000))
multi_class = ['auto', 'ovr', 'multinomial']
combinations = list(itertools.product(solvers, penalty, C, max_iter, multi_class))
```

```
def tune_logistic_regression(X_train, y_train, X_test, y_test, combinations):
    y_train = y_train.values.ravel()
    y_test = y_test.values.ravel()

    # DataFrame to store results
    results_df = pd.DataFrame(columns=["Solver", "Penalty", "C", "Max_Iter", "Multi_Class",
                                       "Train_Accuracy", "Test_Accuracy",
                                       "Train_Precision", "Test_Precision",
                                       "Train_Recall", "Test_Recall",
                                       "Train_F1", "Test_F1"])

    # Iterate over combinations
    for comb in combinations:
        solver, penalty, C, max_iter, multi_class = comb

        if solver in ['lbfgs', 'newton-cg', 'newton-cholesky', 'sag'] and penalty not in ['l2', None]:
            continue
        if solver == 'liblinear' and penalty == 'elasticnet':
            continue
        if solver == 'liblinear' and multi_class == 'multinomial':
            continue
        try:
            # Create and train the Logistic Regression model
                model = LogisticRegression(solver=solver, penalty=penalty, C=C, max_iter=max_iter,
multi_class=multi_class)
            model.fit(X_train, y_train)

            # Make predictions
            y_pred_train = model.predict(X_train)
            y_pred_test = model.predict(X_test)

            # Calculate metrics
            train_accuracy = accuracy_score(y_train, y_pred_train)
            test_accuracy = accuracy_score(y_test, y_pred_test)
            train_precision = precision_score(y_train, y_pred_train, average='binary', zero_division=0)
            test_precision = precision_score(y_test, y_pred_test, average='binary', zero_division=0)
            train_recall = recall_score(y_train, y_pred_train, average='binary', zero_division=0)
            test_recall = recall_score(y_test, y_pred_test, average='binary', zero_division=0)
            train_f1 = f1_score(y_train, y_pred_train, average='binary', zero_division=0)
            test_f1 = f1_score(y_test, y_pred_test, average='binary', zero_division=0)

            new_row = pd.DataFrame([{
                "Solver": solver, "Penalty": penalty, "C": C, "Max_Iter": max_iter, "Multi_Class": multi_class,
                "Train_Accuracy": train_accuracy, "Test_Accuracy": test_accuracy,
                "Train_Precision": train_precision, "Test_Precision": test_precision,
                "Train_Recall": train_recall, "Test_Recall": test_recall,
```

```
                "Train_F1": train_f1, "Test_F1": test_f1
            }])
            results_df = pd.concat([results_df, new_row], ignore_index=True)

        except Exception as e:
            #print(f"An error occurred with combination {comb}: {e}")
            pass
        continue

    # Find the best combination based on test F1-score
    best_combination = results_df.loc[results_df['Test_F1'].idxmax()]

    metrics = ['Accuracy', 'F1 Score', 'Recall', 'Precision']
            train_scores = [results_df['Train_Accuracy'].mean(), results_df['Train_F1'].mean(),
results_df['Train_Recall'].mean(), results_df['Train_Precision'].mean()]
            valid_scores = [results_df['Test_Accuracy'].mean(), results_df['Test_F1'].mean(),
results_df['Test_Recall'].mean(), results_df['Test_Precision'].mean()]

    x = np.arange(len(metrics))  # the label locations
    width = 0.35  # the width of the bars

    fig, ax = plt.subplots()
    rects1 = ax.bar(x - width/2, train_scores, width, label='Train')
    rects2 = ax.bar(x + width/2, valid_scores, width, label='Validation')

    # Add some text for labels, title and custom x-axis tick labels, etc.
    ax.set_ylabel('Scores')
    ax.set_title('Mean Training / Validation Performance')
    ax.set_xticks(x)
    ax.set_xticklabels(metrics)

    # Positioning the legend at the bottom right inside the plot
    ax.legend(loc='lower right')

    autolabel(rects1, ax)
    autolabel(rects2, ax)
    plt.show()

    return best_combination, results_df
```

```
best_comb_log, results = tune_logistic_regression(X_train, y_train, X_test, y_test, combinations)
```

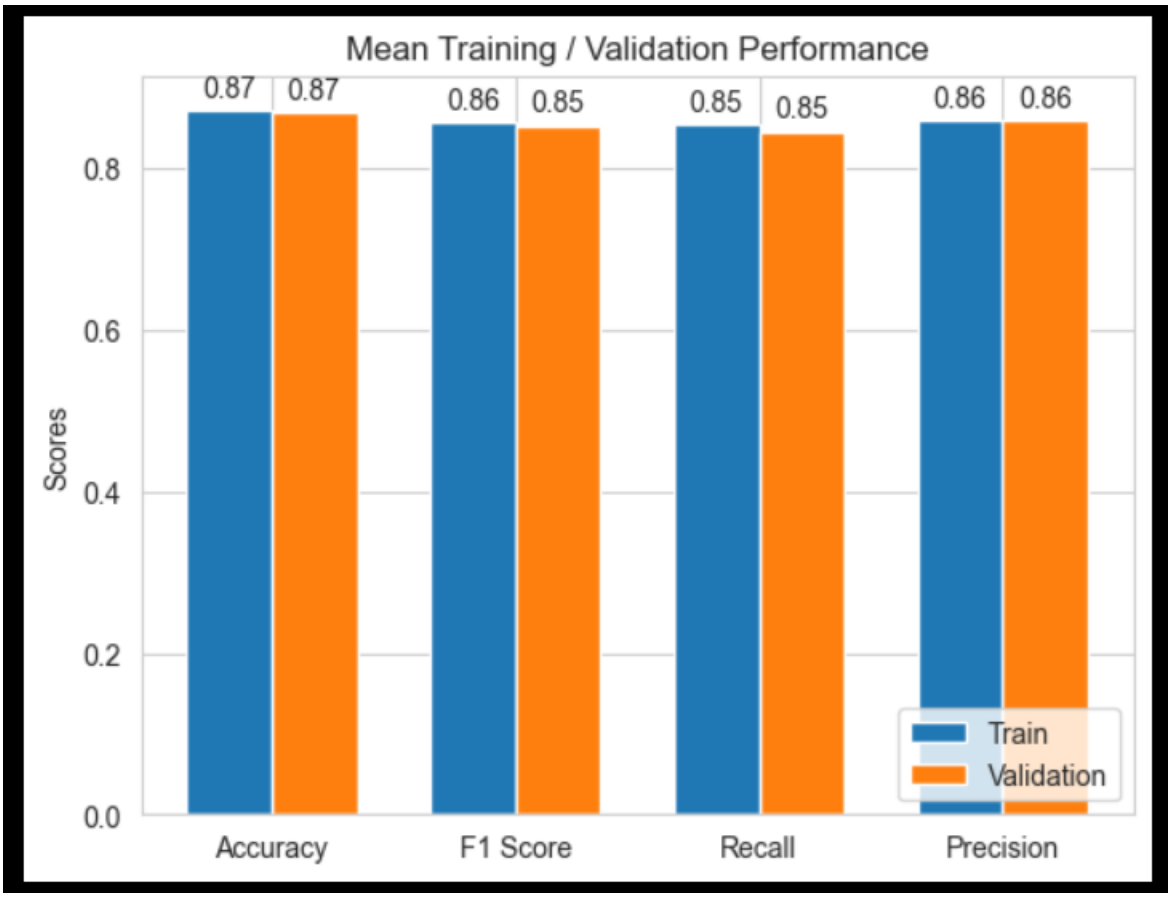


*Figure 12 Mean training / Validation performance*

```
best_comb_log
```

```
Solver              liblinear
Penalty                    l2
C                       0.001
Max_Iter                 1000
Multi_Class              auto
Train_Accuracy       0.885392
Test_Accuracy        0.886029
Train_Precision      0.884233
Test_Precision       0.887111
Train_Recall          0.88752
Test_Recall          0.883707
Train_F1             0.885873
Test_F1              0.885406
Name: 333, dtype: object
```

*Figure 13 Results*

### 5.5.2 KNN

Nearest neighbor algorithm **[38]**

```
n_neighbors = range(1, 31)
weights = ['uniform', 'distance']
metrics = ['euclidean', 'manhattan', 'minkowski']
combinations = list(itertools.product(n_neighbors, weights, metrics))



def tune_knn(X_train, y_train, X_test, y_test, combinations):
    # Flatten y arrays
    y_train = y_train.values.ravel()
    y_test = y_test.values.ravel()

    # DataFrame to store results
    results_df = pd.DataFrame(columns=["N_Neighbors", "Weights", "Metric",
                                       "Train_Accuracy", "Test_Accuracy",
                                       "Train_Precision", "Test_Precision",
                                       "Train_Recall", "Test_Recall",
                                       "Train_F1", "Test_F1"])

    # Iterate over combinations
    for comb in combinations:
        n_neighbor, weight, metric = comb

        try:
            # Create and train the KNN model
            model = KNeighborsClassifier(n_neighbors=n_neighbor, weights=weight, metric=metric)
            model.fit(X_train, y_train)

            # Make predictions
            y_pred_train = model.predict(X_train)
```

```
            y_pred_test = model.predict(X_test)

            # Calculate metrics
            train_accuracy = accuracy_score(y_train, y_pred_train)
            test_accuracy = accuracy_score(y_test, y_pred_test)
            train_precision = precision_score(y_train, y_pred_train, average='binary', zero_division=0)
            test_precision = precision_score(y_test, y_pred_test, average='binary', zero_division=0)
            train_recall = recall_score(y_train, y_pred_train, average='binary', zero_division=0)
            test_recall = recall_score(y_test, y_pred_test, average='binary', zero_division=0)
            train_f1 = f1_score(y_train, y_pred_train, average='binary', zero_division=0)
            test_f1 = f1_score(y_test, y_pred_test, average='binary', zero_division=0)

            new_row = pd.DataFrame([{
                "N_Neighbors": n_neighbor, "Weights": weight, "Metric": metric,
                "Train_Accuracy": train_accuracy, "Test_Accuracy": test_accuracy,
                "Train_Precision": train_precision, "Test_Precision": test_precision,
                "Train_Recall": train_recall, "Test_Recall": test_recall,
                "Train_F1": train_f1, "Test_F1": test_f1
            }])
            results_df = pd.concat([results_df, new_row], ignore_index=True)

        except Exception as e:
            #print(f"An error occurred with combination {comb}: {e}")
            pass

    # Find the best combination based on test F1-score
    best_combination = results_df.loc[results_df['Test_F1'].idxmax()]

    metrics = ['Accuracy', 'F1 Score', 'Recall', 'Precision']
            train_scores      =      [results_df['Train_Accuracy'].mean(),      results_df['Train_F1'].mean(),
results_df['Train_Recall'].mean(), results_df['Train_Precision'].mean()]
             valid_scores      =      [results_df['Test_Accuracy'].mean(),      results_df['Test_F1'].mean(),
results_df['Test_Recall'].mean(), results_df['Test_Precision'].mean()]

    x = np.arange(len(metrics))  # the label locations
    width = 0.35  # the width of the bars

    fig, ax = plt.subplots()
    rects1 = ax.bar(x - width/2, train_scores, width, label='Train')
    rects2 = ax.bar(x + width/2, valid_scores, width, label='Validation')

    # Add some text for labels, title and custom x-axis tick labels, etc.
    ax.set_ylabel('Scores')
    ax.set_title('Mean Training / Validation Performance')
    ax.set_xticks(x)
    ax.set_xticklabels(metrics)

    # Positioning the legend at the bottom right inside the plot
    ax.legend(loc='lower right')

    autolabel(rects1, ax)
    autolabel(rects2, ax)
    #add_watermark()
```

```
    plt.show()

    return best_combination, results_df

best_comb_knn, results_knn = tune_knn(X_train, y_train, X_test, y_test, combinations)
```

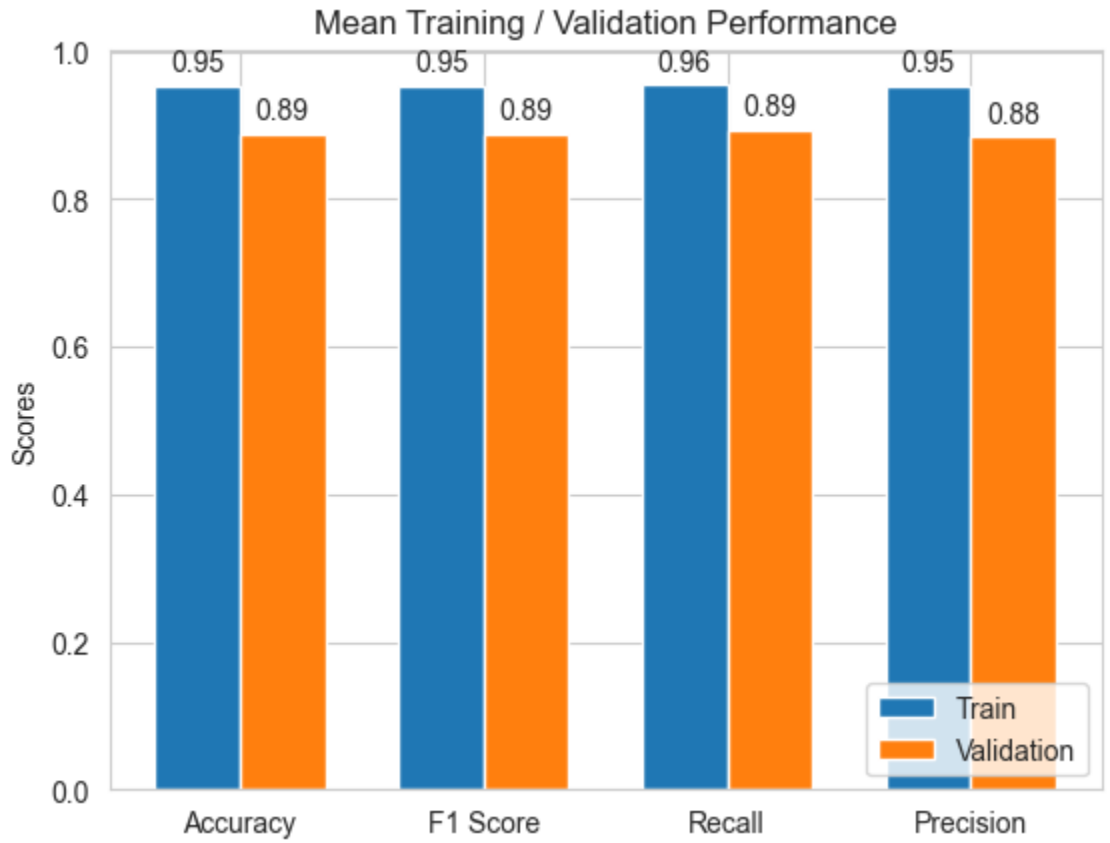


*Figure 14 KNN Mean Training / Validation Performance*

**AdaBoost Classifier**

Ο Ada-boost classifier είναι ένας ensemble boosting classification algorithm ο οποίος προτάθηκε από τον Yoav Freund και τον Robert Schapire το 1996. Συνδυάζει πολλαπλούς ταξινομητές για να μεγιστοποιείσει την ακρίβεια ταξινόμησης. Ο AdaBoost είναι μια επαναληπτική ensemble μέθοδος. Συγκεκριμένα με αυτή την μέθοδο μπορούμε να κατασκευάσουμε έναν "δυνατό" ταξινομητής συνδιάζοντας πολλούς μη αποδοτικούς classifiers. Η βασική ιδέα πίσω από τον Adaboost είναι να θέσουμε τα βάρη των ταξινομητών και να εκπαιδεύσουμε τους ταξινομητές έτσι ώστε να μπορούν να προβλέψουν ασηνύθιστα instances. Οποιοσδήποτε machine learning algorithm μπορεί να χρησιμοποιηθεί για base classifier αν μπορεί να κάνει utilize βάρη στο training set **[39]**.

Για τον AdaBoost πρέπει να ισχύουν τα εξής conditions:

Ο ταξινομητής πρέπει να εκπαιδευθεί σε συνδιασμό με τα weighed training examples
Σε κάθε επανάληψη, πρέπει να προσπαθεί να κάνει fit τα δεδομένα ακριβώς ελαχιστοποιώντας το training error

Τρόπος λειτουργίας

Αρχικά, ο AdaBoost επιλέγει τυχαία ένα υποσύνολο εκπαίδευσης.
Εκπαιδεύει επαναληπτικά το μοντέλο μηχανικής μάθησης AdaBoost επιλέγοντας το σετ εκπαίδευσης με βάση την ακριβή πρόβλεψη της τελευταίας εκπαίδευσης.
Αναθέτει μεγαλύτερο βάρος στις παρατηρήσεις που ταξινομήθηκαν λανθασμένα, ώστε στην επόμενη επανάληψη αυτές οι παρατηρήσεις να λάβουν υψηλότερη πιθανότητα για ταξινόμηση.
Αναθέτει βάρος στον εκπαιδευμένο ταξινομητή σε κάθε επανάληψη ανάλογα με την ακρίβεια του ταξινομητή. Ο πιο ακριβής ταξινομητής θα λάβει υψηλότερο βάρος.
Αυτή η διαδικασία επαναλαμβάνεται μέχρι να ταιριάξει πλήρως το σύνολο δεδομένων εκπαίδευσης χωρίς κανένα λάθος ή μέχρι να φτάσει στον καθορισμένο μέγιστο αριθμό εκτιμητών.
Τέλος για να γίνει ταξινόμηση μιας παρατήρησης, διεξάγεται μια "ψηφοφορία" μεταξύ όλων των αλγορίθμων μάθησης που έχουν δημιουργηθεί.

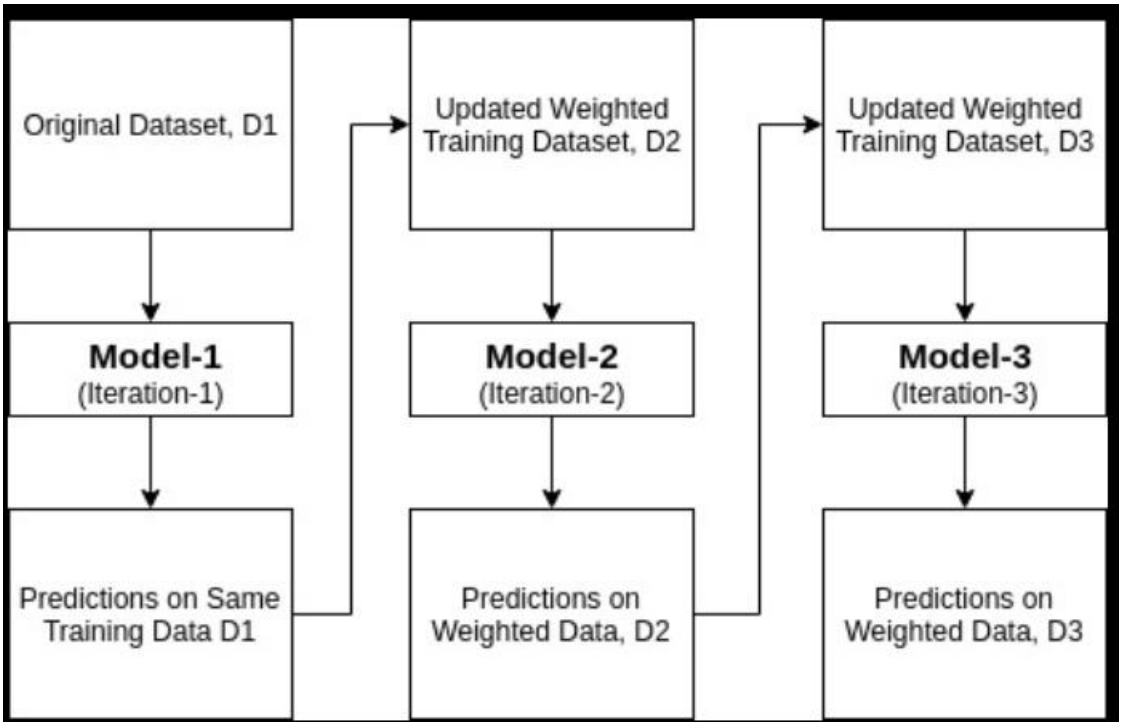


*Figure 15 Adaboost classifier*

A python implementation **[40]**

```
n_estimators_range = range(10, 101, 10)
learning_rate_range = [0.01, 0.1, 1, 10]
base_estimators = [DecisionTreeClassifier(max_depth=1), DecisionTreeClassifier(max_depth=2)]
combinations = list(itertools.product(n_estimators_range, learning_rate_range, base_estimators))
```

```
def tune_adaboost(X_train, y_train, X_test, y_test, combinations):


    # DataFrame to store results
    results_df = pd.DataFrame(columns=["N_Estimators", "Learning_Rate", "Base_Estimator",
                                       "Train_Accuracy", "Test_Accuracy",
                                       "Train_Precision", "Test_Precision",
                                       "Train_Recall", "Test_Recall",
                                       "Train_F1", "Test_F1"])

    # Flatten y arrays
    y_train = y_train.values.ravel()
    y_test = y_test.values.ravel()

    # Iterate over combinations
    for comb in combinations:
        n_estimators, learning_rate, base_estimator = comb
        try:
            # Create and train the AdaBoost model
```

```
            model = AdaBoostClassifier(estimator=base_estimator,
                                       n_estimators=n_estimators,
                                       learning_rate=learning_rate,
                                       random_state=42)
            model.fit(X_train, y_train)

            # Make predictions
            y_pred_train = model.predict(X_train)
            y_pred_test = model.predict(X_test)

            # Calculate metrics
            train_accuracy = accuracy_score(y_train, y_pred_train)
            test_accuracy = accuracy_score(y_test, y_pred_test)
            train_precision = precision_score(y_train, y_pred_train, average='binary', zero_division=0)
            test_precision = precision_score(y_test, y_pred_test, average='binary', zero_division=0)
            train_recall = recall_score(y_train, y_pred_train, average='binary', zero_division=0)
            test_recall = recall_score(y_test, y_pred_test, average='binary', zero_division=0)
            train_f1 = f1_score(y_train, y_pred_train, average='binary', zero_division=0)
            test_f1 = f1_score(y_test, y_pred_test, average='binary', zero_division=0)

            # Store results
            new_row = pd.DataFrame([{
                        "N_Estimators": n_estimators, "Learning_Rate": learning_rate, "Base_Estimator":
type(base_estimator).__name__,
                "Train_Accuracy": train_accuracy, "Test_Accuracy": test_accuracy,
                "Train_Precision": train_precision, "Test_Precision": test_precision,
                "Train_Recall": train_recall, "Test_Recall": test_recall,
                "Train_F1": train_f1, "Test_F1": test_f1
            }])
            results_df = pd.concat([results_df, new_row], ignore_index=True)

        except Exception as e:
            print(f"An error occurred with combination {n_estimators, learning_rate, base_estimator}: {e}")
            pass

    # Find the best combination based on test F1-score
    best_combination = results_df.loc[results_df['Test_F1'].idxmax()]

    metrics = ['Accuracy', 'F1 Score', 'Recall', 'Precision']
            train_scores = [results_df['Train_Accuracy'].mean(), results_df['Train_F1'].mean(),
results_df['Train_Recall'].mean(), results_df['Train_Precision'].mean()]
            valid_scores = [results_df['Test_Accuracy'].mean(), results_df['Test_F1'].mean(),
results_df['Test_Recall'].mean(), results_df['Test_Precision'].mean()]

    x = np.arange(len(metrics))  # the label locations
    width = 0.35  # the width of the bars

    fig, ax = plt.subplots()
    rects1 = ax.bar(x - width/2, train_scores, width, label='Train')
    rects2 = ax.bar(x + width/2, valid_scores, width, label='Validation')

    # Add some text for labels, title and custom x-axis tick labels, etc.
    ax.set_ylabel('Scores')
```

```
    ax.set_title('Mean Training / Validation Performance')
    ax.set_xticks(x)
    ax.set_xticklabels(metrics)

    # Positioning the legend at the bottom right inside the plot
    ax.legend(loc='lower right')

    autolabel(rects1, ax)
    autolabel(rects2, ax)
    plt.show()

    return best_combination, results_df
```

```
best_comb_ada, results_ada = tune_adaboost(X_train, y_train, X_test, y_test, combinations)
```

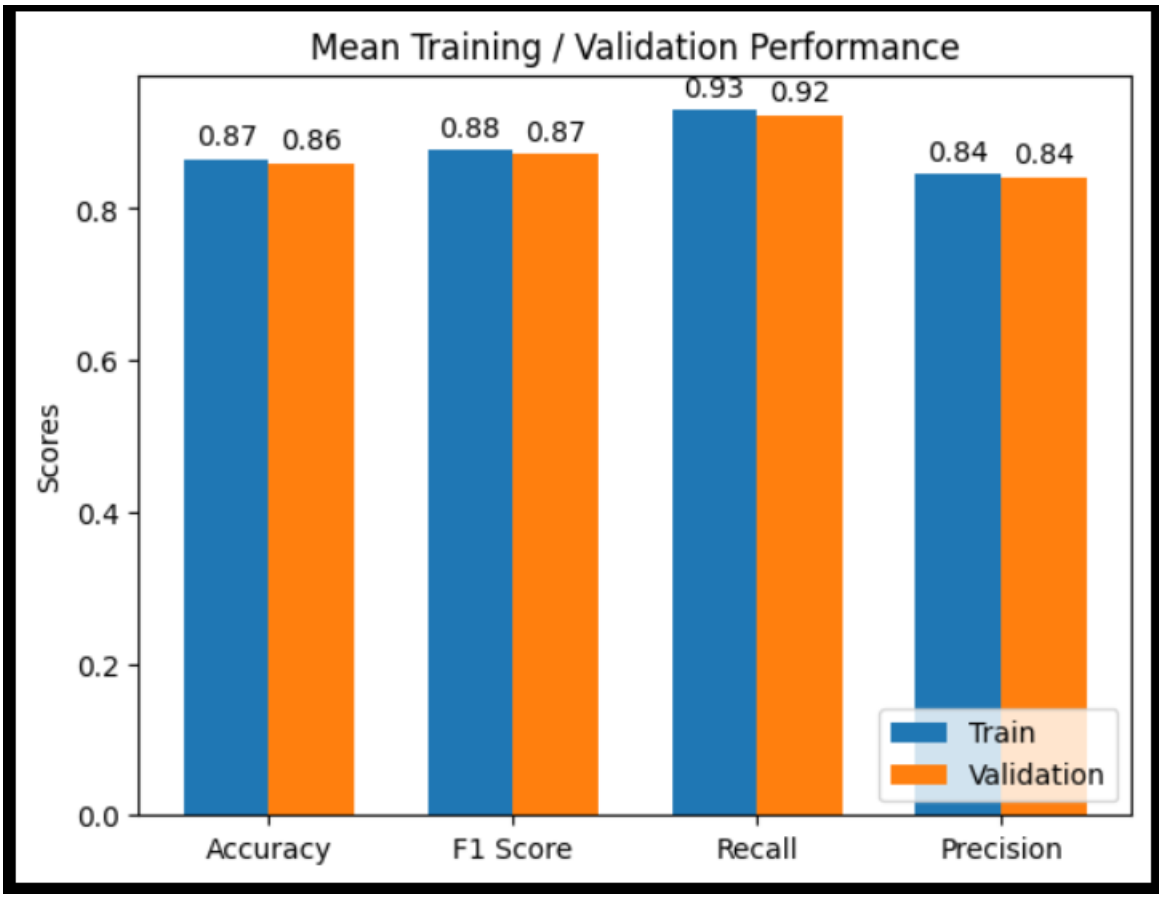


*Figure 16 Adaboost classifier Training / Validation performance*

### 5.5.3 BernoulliNB

Ο Bernoulli Naive Bayes είναι ένας απλός αλγόριθμος ταξινόμησης βασισμένος στο θεώρημα Bayes, ιδανικός για δεδομένα με δυαδικά (binary) χαρακτηριστικά. Χρησιμοποιείται συχνά σε κατηγορίες δεδομένων όπου τα χαρακτηριστικά παίρνουν μόνο δύο τιμές, όπως Ναι/Όχι ή 0/1 **[41]**.

**Τρόπος Λειτουργίας:**
Υπολογίζει τις πιθανότητες των διαφόρων χαρακτηριστικών για κάθε κλάση. Χρησιμοποιεί αυτές τις πιθανότητες για να αξιολογήσει την πιθανότητα μιας νέας παρατήρησης να ανήκει σε κάθε δυνατή κλάση. Αποφασίζει την ταξινόμηση της παρατήρησης βάσει της κλάσης με την υψηλότερη πιθανότητα.

**Κύριες Παράμετροι:**
alpha (Αλφα): Παράγοντας εξομάλυνσης για την αποφυγή του προβλήματος της μηδενικής πιθανότητας σε απουσία συγκεκριμένου χαρακτηριστικού.
fit_prior (Εφαρμογή προτεραιότητας): Εάν είναι True, ο αλγόριθμος μαθαίνει τις πιθανότητες των κλάσεων από τα δεδομένα. Αν είναι False, χρησιμοποιεί ίσες πιθανότητες για κάθε κλάση.
binarize (Δυαδικοποίηση): Ορίζει το κατώφλι για τη δυαδικοποίηση των χαρακτηριστικών. Χαρακτηριστικά με τιμή πάνω από αυτό το κατώφλι μετατρέπονται σε 1, αλλιώς σε 0.

```
alpha_range = np.linspace(0.0, 1.0, num=10)  # Alpha values (smoothing parameter)
fit_prior_options = [True, False]  # Whether to learn class prior probabilities or not
binarize_range = np.linspace(0.0, 1.0, num=10)  # Threshold for binarizing (mapping to boolean) of sample features
combinations = list(itertools.product(alpha_range, fit_prior_options, binarize_range))
```

```
def tune_bernoulli_nb(X_train, y_train, X_test, y_test, combinations):


    # DataFrame to store results
        results_df = pd.DataFrame(columns=["Alpha", "Fit_Prior", "Binarize", "Train_Accuracy", "Test_Accuracy",
                                   "Train_Precision", "Test_Precision",
                                   "Train_Recall", "Test_Recall",
                                   "Train_F1", "Test_F1"])

    # Flatten y arrays
    y_train = y_train.values.ravel()
    y_test = y_test.values.ravel()

    # Iterate over hyperparameter combinations
    for alpha, fit_prior, binarize in combinations:
        try:
            # Create and train the Bernoulli Naive Bayes model
            model = BernoulliNB(alpha=alpha, fit_prior=fit_prior, binarize=binarize)
            model.fit(X_train, y_train)

            # Make predictions
```

```
            y_pred_train = model.predict(X_train)
            y_pred_test = model.predict(X_test)

            # Calculate metrics
            train_accuracy = accuracy_score(y_train, y_pred_train)
            test_accuracy = accuracy_score(y_test, y_pred_test)
            train_precision = precision_score(y_train, y_pred_train, average='binary', zero_division=0)
            test_precision = precision_score(y_test, y_pred_test, average='binary', zero_division=0)
            train_recall = recall_score(y_train, y_pred_train, average='binary', zero_division=0)
            test_recall = recall_score(y_test, y_pred_test, average='binary', zero_division=0)
            train_f1 = f1_score(y_train, y_pred_train, average='binary', zero_division=0)
            test_f1 = f1_score(y_test, y_pred_test, average='binary', zero_division=0)

            new_row = pd.DataFrame([{
                "Alpha": alpha, "Fit_Prior": fit_prior, "Binarize": binarize,
                "Train_Accuracy": train_accuracy, "Test_Accuracy": test_accuracy,
                "Train_Precision": train_precision, "Test_Precision": test_precision,
                "Train_Recall": train_recall, "Test_Recall": test_recall,
                "Train_F1": train_f1, "Test_F1": test_f1
            }])
            results_df = pd.concat([results_df, new_row], ignore_index=True)

        except Exception as e:
            print(f"An error occurred with parameters {alpha, fit_prior, binarize}: {e}")

    # Find the best combination based on test F1-score
    best_combination = results_df.loc[results_df['Test_F1'].idxmax()]

    metrics = ['Accuracy', 'F1 Score', 'Recall', 'Precision']
            train_scores      =      [results_df['Train_Accuracy'].mean(),      results_df['Train_F1'].mean(),
results_df['Train_Recall'].mean(), results_df['Train_Precision'].mean()]
             valid_scores      =      [results_df['Test_Accuracy'].mean(),      results_df['Test_F1'].mean(),
results_df['Test_Recall'].mean(), results_df['Test_Precision'].mean()]

    x = np.arange(len(metrics))  # the label locations
    width = 0.35  # the width of the bars

    fig, ax = plt.subplots()
    rects1 = ax.bar(x - width/2, train_scores, width, label='Train')
    rects2 = ax.bar(x + width/2, valid_scores, width, label='Validation')

    # Add some text for labels, title and custom x-axis tick labels, etc.
    ax.set_ylabel('Scores')
    ax.set_title('Mean Training / Validation Performance')
    ax.set_xticks(x)
    ax.set_xticklabels(metrics)

    # Positioning the legend at the bottom right inside the plot
    ax.legend(loc='lower right')

    autolabel(rects1, ax)
    autolabel(rects2, ax)
    plt.show()
```

```
    return best_combination, results_df

warnings.filterwarnings('ignore') # I supress the warnings about scikit learn
best_comb_mnb, results_mnb = tune_bernoulli_nb(X_train, y_train, X_test, y_test, combinations)
```

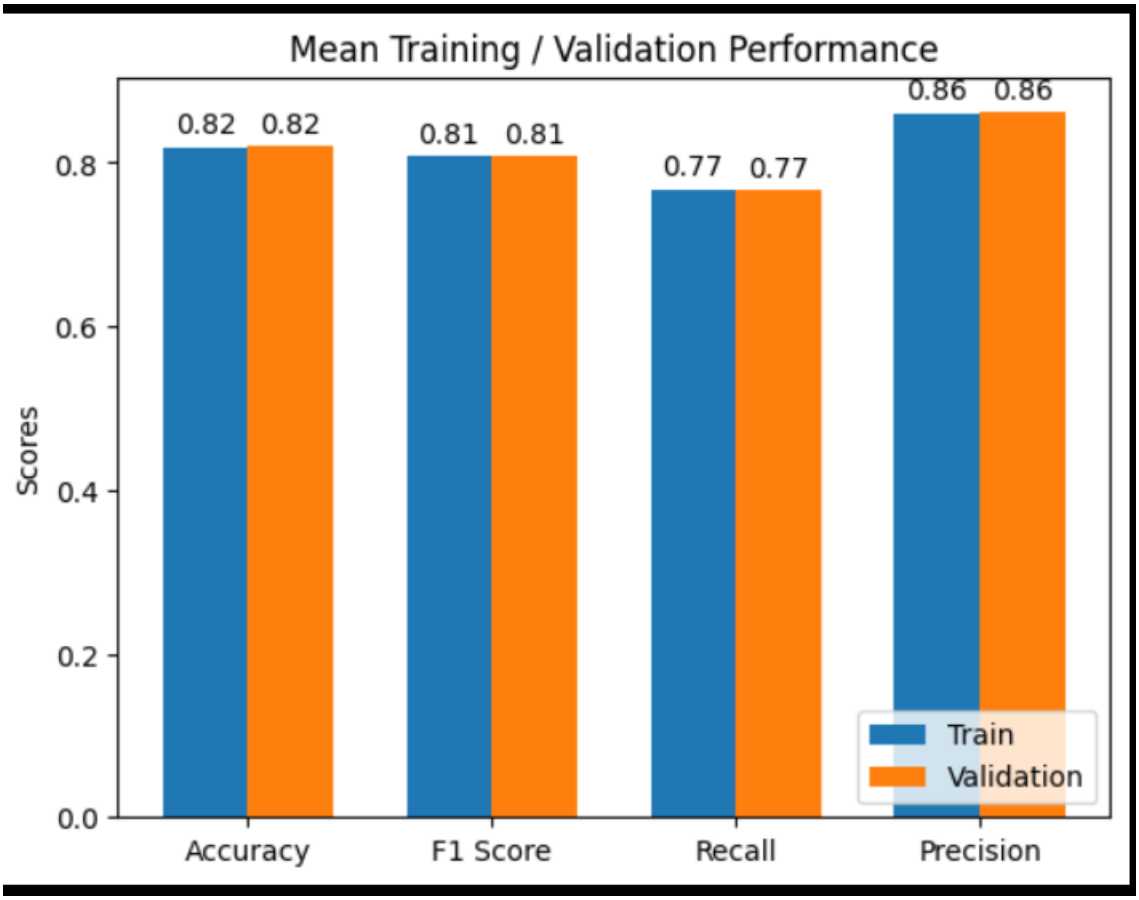


*Figure 17 Bernouli Training / Validation performance*

### 5.5.4 ElasticNet

Τρόπος λειτουργίας

Η μέθοδος ElasticNet συνδιάζει τις μεθόδους του Ridge Regression και του Lasso Regression **[42]**

Ridge Regression: Είναι χρήσιμο για την μείωση των coefficients σε λιγότερο σημαντικά features
Lasso Regression: Με αυτή την τεχνική μειώνουμε σε μηδέν τα coefficient των λιγότερο σημαντικών features

Σε datasets με πολλά features και ειδικότερα σε datasets με features που είναι correlated πολλές φορές είναι δύσκολο να βρεθούν τα σημαντικά features

Ο αλγόριθμος ElasticNet προσφέρει λύση στο παραπάνω πρόβλημα με τις τεχνικές της επιλογής (Lasso) και της μείωσης (Ridge) των coefficients.
Το τελικό αποτέλεσμα είναι ένα μοντέλο που δεν είναι ούτε πολύ πολύπλοκο, οπότε αποφεύγουμε το overfitting ούτε πολύ από κατα συνέπεια αποφεύγουμε το underfitting

Κύριες παράμετροι

alpha: Είναι το hyperparameter που ελέγχει το ποσοστό του penaltie. Υψυλότερο α σημαίνει περισσότερο regularization (ποιο απλό μοντέλο)
l1_ratio: Η συγκεκριμένη υπερπαράμετρος καθορίζει τον συνδιασμό Lasso και Ridge. Στην περίπτωση που το θέσουμε 1 σημαίνει ότι θα χρησιμοποιήσουμε αποκλειστικά Lasso, Αντιθέτως στην περίπτωση που το θέσουμε 0 θα χρησιμοποιήσουμε αποκλειστικά Ridge.

R2:Το R2 μετράει το ποσοστό του variance της εξαρτημένης μεταβλητής που είναι προβλέψιμο από τις ανεξάρτητες μεταβλητές. Δηλαδή δείχνει πόσο καλά το παρατηρούμενο αποτέλεσμα αναπαριστάται από το μοντέλο βασιζόμενο στο total variation των εξόδων. Υψηλότερο R2 δείχνει καλύτερο fit μεταξύ προβλέψεων και δεδομένων

MSE:Το MSE μετράει το average των τετραγώνων των errors, δηλαδή τον μέσο όρο της τετραγωνισμένης διαφοράς μεταξύ της τιμής που πρόβλεφθηκε και της πραγματικής τιμής.

```
def tune_elastic_net(X_train, y_train, X_test, y_test):
    # Define the range of hyperparameters
    alphas = np.logspace(-4, 4, 20)  # Regularization strength
    l1_ratios = np.linspace(0, 1, 10)  # Mix of L1 and L2

    # DataFrame to store results
    results_df = pd.DataFrame(columns=['Alpha', 'L1_Ratio', 'Train_R2', 'Test_R2', 'Train_MSE',
'Test_MSE'])

    # Iterate over hyperparameter combinations
    for alpha, l1_ratio in itertools.product(alphas, l1_ratios):
        try:
            # Create and train the Elastic Net model
            model = ElasticNet(alpha=alpha, l1_ratio=l1_ratio, max_iter=100000)
            model.fit(X_train, y_train)

            # Make predictions
            y_pred_train = model.predict(X_train)
            y_pred_test = model.predict(X_test)

            # Calculate metrics
            train_r2 = r2_score(y_train, y_pred_train)
            test_r2 = r2_score(y_test, y_pred_test)
            train_mse = mean_squared_error(y_train, y_pred_train)
            test_mse = mean_squared_error(y_test, y_pred_test)

            # Append results to the DataFrame
            new_row = {
                'Alpha': alpha,
                'L1_Ratio': l1_ratio,
```

```
                'Train_R2': train_r2,
                'Test_R2': test_r2,
                'Train_MSE': train_mse,
                'Test_MSE': test_mse
            }
            new_row = pd.DataFrame([new_row])
            results_df = pd.concat([results_df, new_row], ignore_index=True)

        except Exception as e:
            print(f"An error occurred with parameters {alpha, l1_ratio}: {e}")

    # Find the best combination based on test R2
    best_combination = results_df.loc[results_df['Test_R2'].idxmax()]

    # Plotting
    metrics = ['Train R2', 'Test R2', 'Train MSE', 'Test MSE']
    train_scores = [best_combination['Train_R2'], best_combination['Test_R2']]
    test_scores = [best_combination['Train_MSE'], best_combination['Test_MSE']]

    x = np.arange(len(metrics) // 2)  # the label locations
    width = 0.35  # the width of the bars

    fig, ax = plt.subplots()
    rects1 = ax.bar(x - width/2, train_scores, width, label='Train')
    rects2 = ax.bar(x + width/2, test_scores, width, label='Test')

    # Add some text for labels, title and custom x-axis tick labels, etc.
    ax.set_ylabel('Scores')
    ax.set_title('Elastic Net Performance Metrics')
    ax.set_xticks(x)
    ax.set_xticklabels(metrics[::2])

    # Positioning the legend at the bottom right inside the plot
    ax.legend(loc='lower right')

    autolabel(rects1, ax)
    autolabel(rects2, ax)
    plt.show()

    return best_combination, results_df


warnings.filterwarnings('ignore')
best_comb, results = tune_elastic_net(X_train, y_train, X_test, y_test)
```

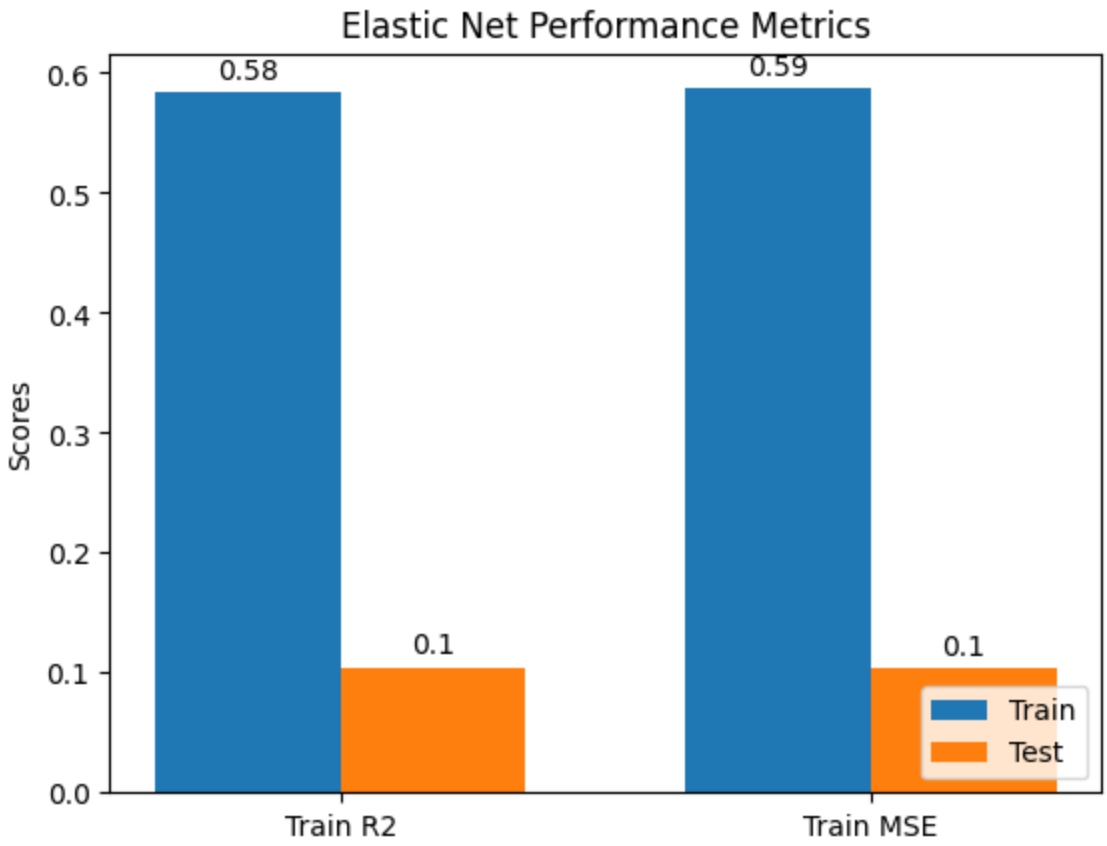


*Figure 18 Elastic Net metrics*

### 5.6 Neural Network

```
# convert our dataframe to numpy array
# X_train, X_test, y_train, y_test
X_train_np = X_train.to_numpy()
y_train_np = y_train.to_numpy()
X_test_np = X_test.to_numpy()
y_test_np = y_test.to_numpy()


X_train_tensor = torch.Tensor(X_train_np)  # transform to torch tensor
y_train_tensor = torch.Tensor(y_train_np)
X_test_tensor = torch.Tensor(X_test_np)
y_test_tensor = torch.Tensor(y_test_np)

train_dataset = TensorDataset(X_train_tensor, y_train_tensor)  # create our dataset
train_loader = DataLoader(train_dataset, batch_size=16, shuffle=True)  # create our dataloader

test_dataset = TensorDataset(X_test_tensor, y_test_tensor)  # create our dataset
test_loader = DataLoader(test_dataset, batch_size=16, shuffle=True)  # create our dataloader
```

Neural network with 1 hidden layer
References: **[43]**, **[44]**

```
class Net_1(nn.Module):
    def __init__(self, D_in, H1, D_out):
        super(Net_1, self).__init__()
        self.fc1 = nn.Linear(D_in, H1)
        self.fc2 = nn.Linear(H1, D_out)

    def forward(self, x):
        x = torch.relu(self.fc1(x))
        x = torch.sigmoid(self.fc2(x))  # Sigmoid for binary classification
        return x
```

```
def train(model, loss_func, optimizer, num_epochs, train_loader, valid_loader):
    # Lists to store performance metrics
    train_acc, valid_acc = [], []
    train_f1, valid_f1 = [], []
    train_precision, valid_precision = [], []
    train_recall, valid_recall = [], []

    for epoch in range(num_epochs):
        # Training phase
        model.train()  # Set the model to training mode
        train_preds, train_labels = [], []

        for inputs, labels in train_loader:
            optimizer.zero_grad()  # Zero the gradients
            outputs = model(inputs)
            predicted = (outputs.data > 0.5).float()
            loss = loss_func(outputs, labels.float())  # Compute loss
            loss.backward()  # Backward pass
            optimizer.step()  # Update weights

            #print('PREDICTED: ', predicted, '\n')
            train_preds.extend(predicted.cpu().numpy())
            train_labels.extend(labels.cpu().numpy())

        # Calculate and store training metrics
        train_acc.append(accuracy_score(train_labels, train_preds))
        train_f1.append(f1_score(train_labels, train_preds, average='weighted', zero_division=0))
                train_precision.append(precision_score(train_labels,    train_preds,    average='weighted',
zero_division=0))
        train_recall.append(recall_score(train_labels, train_preds, average='weighted', zero_division=0))

        # Validation phase
        model.eval()  # Set the model to evaluation mode
        valid_preds, valid_labels = [], []
        with torch.no_grad():  # Disable gradient calculation for efficiency
            for inputs, labels in valid_loader:
                outputs = model(inputs)
```

```
            predicted = (outputs.data > 0.5).float()
            valid_preds.extend(predicted.cpu().numpy())
            valid_labels.extend(labels.cpu().numpy())

        # Calculate and store validation metrics
        valid_acc.append(accuracy_score(valid_labels, valid_preds))
        valid_f1.append(f1_score(valid_labels, valid_preds, average='weighted', zero_division=0))
            valid_precision.append(precision_score(valid_labels,  valid_preds,  average='weighted',
zero_division=0))
                    valid_recall.append(recall_score(valid_labels,  valid_preds,  average='weighted',
zero_division=0))

        # Print epoch results
        print(f'Epoch {epoch+1}/{num_epochs}, '
            f'Train Acc: {train_acc[-1]:.2f}, Valid Acc: {valid_acc[-1]:.2f}, '
            f'Train F1: {train_f1[-1]:.2f}, Valid F1: {valid_f1[-1]:.2f}, '
            f'Train Precision: {train_precision[-1]:.2f}, Valid Precision: {valid_precision[-1]:.2f}, '
            f'Train Recall: {train_recall[-1]:.2f}, Valid Recall: {valid_recall[-1]:.2f}')

    print(f'MEAN TRAIN ACCURACY: {np.mean(train_acc):.2f}\n'
        f'MEAN VALIDATION ACCURACY: {np.mean(valid_acc):.2f}\n'
        f'MEAN TRAIN F1 SCORE: {np.mean(train_f1):.2f}\n'
        f'MEAN VALIDATION F1 SCORE: {np.mean(valid_f1):.2f}\n'
        f'MEAN TRAIN PRECISION: {np.mean(train_precision):.2f}\n'
        f'MEAN VALIDATION PRECISION: {np.mean(valid_precision):.2f}\n'
        f'MEAN TRAIN RECALL: {np.mean(train_recall):.2f}\n'
        f'MEAN VALIDATION RECALL: {np.mean(valid_recall):.2f}\n')

    metrics = ['Accuracy', 'F1 Score', 'Recall', 'Precision']
            train_scores  =  [np.mean(train_acc),  np.mean(train_f1),  np.mean(train_recall),
np.mean(train_precision)]
            valid_scores  =  [np.mean(valid_acc),  np.mean(valid_f1),  np.mean(valid_recall),
np.mean(valid_precision)]

    x = np.arange(len(metrics))  # the label locations
    width = 0.35  # the width of the bars

    fig, ax = plt.subplots()
    rects1 = ax.bar(x - width/2, train_scores, width, label='Train')
    rects2 = ax.bar(x + width/2, valid_scores, width, label='Validation')

    # Add some text for labels, title and custom x-axis tick labels, etc.
    ax.set_ylabel('Scores')
    ax.set_title('Mean Training / Validation Performance')
    ax.set_xticks(x)
    ax.set_xticklabels(metrics)

    # Positioning the legend at the bottom right inside the plot
    ax.legend(loc='lower right')

    autolabel(rects1, ax)
    autolabel(rects2, ax)
    #add_watermark()
```

```
    plt.show()
        return train_acc, valid_acc, train_f1, valid_f1, train_recall, valid_recall, train_precision,
valid_precision
```

```
# Μοντέλο 1
D_in, H1, D_out = 6, 128, 1
model = Net_1(D_in, H1, D_out)
num_epochs = 20
loss_func = nn.BCELoss() # Binary Cross Entropy Loss for binary classification
optimizer = optim.Adam(model.parameters(), lr=0.00001)
train_acc, valid_acc, train_f1, valid_f1, train_recall, valid_recall, train_precision, valid_precision =
train(model, loss_func, optimizer, num_epochs, train_loader, test_loader)
```

```
Epoch 1/20, Train Acc: 0.82, Valid Acc: 0.84, Train F1: 0.82, Valid F1: 0.83, Train Precision: 0.83,
Valid Precision: 0.84, Train Recall: 0.82, Valid Recall: 0.84
Epoch 2/20, Train Acc: 0.85, Valid Acc: 0.86, Train F1: 0.85, Valid F1: 0.86, Train Precision: 0.86,
Valid Precision: 0.86, Train Recall: 0.85, Valid Recall: 0.86
Epoch 3/20, Train Acc: 0.87, Valid Acc: 0.87, Train F1: 0.87, Valid F1: 0.87, Train Precision: 0.87,
Valid Precision: 0.87, Train Recall: 0.87, Valid Recall: 0.87
Epoch 4/20, Train Acc: 0.88, Valid Acc: 0.87, Train F1: 0.88, Valid F1: 0.87, Train Precision: 0.88,
Valid Precision: 0.87, Train Recall: 0.88, Valid Recall: 0.87
Epoch 5/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88,
Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 6/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88,
Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 7/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88,
Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 8/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88,
Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 9/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88,
Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 10/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88,
Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 11/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88,
Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 12/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88,
Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 13/20, Train Acc: 0.89, Valid Acc: 0.88, Train F1: 0.89, Valid F1: 0.88, Train Precision: 0.89,
Valid Precision: 0.88, Train Recall: 0.89, Valid Recall: 0.88
Epoch 14/20, Train Acc: 0.89, Valid Acc: 0.88, Train F1: 0.89, Valid F1: 0.88, Train Precision: 0.89,
Valid Precision: 0.88, Train Recall: 0.89, Valid Recall: 0.88
Epoch 15/20, Train Acc: 0.89, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.89,
Valid Precision: 0.88, Train Recall: 0.89, Valid Recall: 0.88
Epoch 16/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88,
Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 17/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88,
Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 18/20, Train Acc: 0.89, Valid Acc: 0.88, Train F1: 0.89, Valid F1: 0.88, Train Precision: 0.89,
Valid Precision: 0.88, Train Recall: 0.89, Valid Recall: 0.88
Epoch 19/20, Train Acc: 0.89, Valid Acc: 0.88, Train F1: 0.89, Valid F1: 0.88, Train Precision: 0.89,
Valid Precision: 0.88, Train Recall: 0.89, Valid Recall: 0.88
```

Epoch 20/20, Train Acc: 0.89, Valid Acc: 0.88, Train F1: 0.89, Valid F1: 0.88, Train Precision: 0.89, Valid Precision: 0.88, Train Recall: 0.89, Valid Recall: 0.88
MEAN TRAIN ACCURACY: 0.88
MEAN VALIDATION ACCURACY: 0.88
MEAN TRAIN F1 SCORE: 0.88
MEAN VALIDATION F1 SCORE: 0.88
MEAN TRAIN PRECISION: 0.88
MEAN VALIDATION PRECISION: 0.88
MEAN TRAIN RECALL: 0.88
MEAN VALIDATION RECALL: 0.88

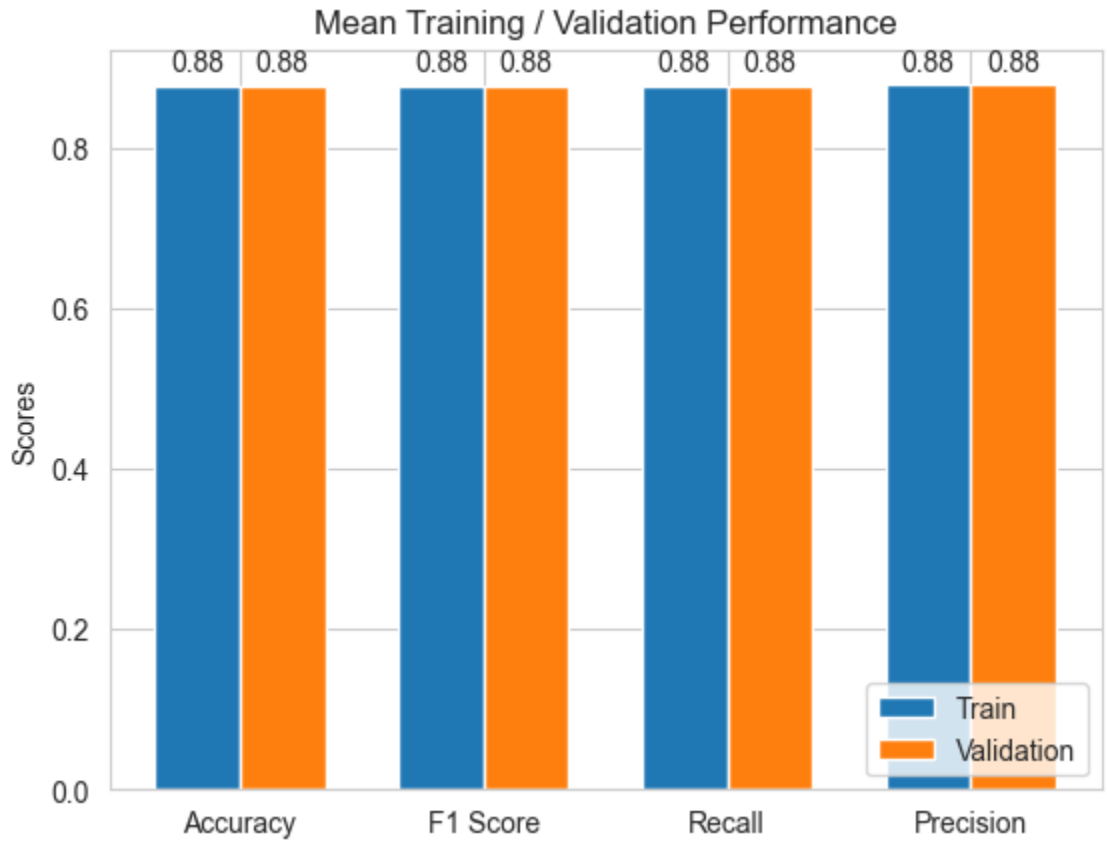


*Figure 19 Swallow NN, Training / Validation performance*

Neural Network with 2 hidden layers

These are called deep neural networks **[45]**.

```
class Net_2(nn.Module):
    def __init__(self, D_in, H1, H2, D_out):
        super(Net_2, self).__init__()
        self.fc1 = nn.Linear(D_in, H1)
        self.fc2 = nn.Linear(H1, H2)
        self.fc3 = nn.Linear(H2, D_out)

    def forward(self, x):
        x = torch.relu(self.fc1(x))
        x = torch.relu(self.fc2(x))
        x = torch.sigmoid(self.fc3(x))  # Sigmoid for binary classification
        return x


# Μοντέλο 2
D_in, H1, H2, D_out = 6, 128, 64, 1
model = Net_2(D_in, H1, H2, D_out)
num_epochs = 20
loss_func = nn.BCELoss() # Binary Cross Entropy Loss for binary classification
optimizer = optim.Adam(model.parameters(), lr=0.00001)
train_acc, valid_acc, train_f1, valid_f1, train_recall, valid_recall, train_precision, valid_precision =
train(model, loss_func, optimizer, num_epochs, train_loader, test_loader)
```

Epoch 1/20, Train Acc: 0.53, Valid Acc: 0.68, Train F1: 0.39, Valid F1: 0.65, Train Precision: 0.75, Valid Precision: 0.80, Train Recall: 0.53, Valid Recall: 0.68
Epoch 2/20, Train Acc: 0.81, Valid Acc: 0.86, Train F1: 0.80, Valid F1: 0.85, Train Precision: 0.84, Valid Precision: 0.87, Train Recall: 0.81, Valid Recall: 0.86
Epoch 3/20, Train Acc: 0.87, Valid Acc: 0.87, Train F1: 0.87, Valid F1: 0.87, Train Precision: 0.87, Valid Precision: 0.88, Train Recall: 0.87, Valid Recall: 0.87

Epoch 4/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 5/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 6/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 7/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 8/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 9/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 10/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 11/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 12/20, Train Acc: 0.89, Valid Acc: 0.88, Train F1: 0.89, Valid F1: 0.88, Train Precision: 0.89, Valid Precision: 0.88, Train Recall: 0.89, Valid Recall: 0.88
Epoch 13/20, Train Acc: 0.89, Valid Acc: 0.88, Train F1: 0.89, Valid F1: 0.88, Train Precision: 0.89, Valid Precision: 0.88, Train Recall: 0.89, Valid Recall: 0.88
Epoch 14/20, Train Acc: 0.89, Valid Acc: 0.88, Train F1: 0.89, Valid F1: 0.88, Train Precision: 0.89, Valid Precision: 0.88, Train Recall: 0.89, Valid Recall: 0.88
Epoch 15/20, Train Acc: 0.89, Valid Acc: 0.88, Train F1: 0.89, Valid F1: 0.88, Train Precision: 0.89, Valid Precision: 0.88, Train Recall: 0.89, Valid Recall: 0.88
Epoch 16/20, Train Acc: 0.89, Valid Acc: 0.88, Train F1: 0.89, Valid F1: 0.88, Train Precision: 0.89, Valid Precision: 0.88, Train Recall: 0.89, Valid Recall: 0.88
Epoch 17/20, Train Acc: 0.89, Valid Acc: 0.88, Train F1: 0.89, Valid F1: 0.88, Train Precision: 0.89, Valid Precision: 0.88, Train Recall: 0.89, Valid Recall: 0.88
Epoch 18/20, Train Acc: 0.89, Valid Acc: 0.88, Train F1: 0.89, Valid F1: 0.88, Train Precision: 0.89, Valid Precision: 0.88, Train Recall: 0.89, Valid Recall: 0.88
Epoch 19/20, Train Acc: 0.89, Valid Acc: 0.88, Train F1: 0.89, Valid F1: 0.88, Train Precision: 0.89, Valid Precision: 0.88, Train Recall: 0.89, Valid Recall: 0.88
Epoch 20/20, Train Acc: 0.89, Valid Acc: 0.88, Train F1: 0.89, Valid F1: 0.88, Train Precision: 0.89, Valid Precision: 0.88, Train Recall: 0.89, Valid Recall: 0.88
MEAN TRAIN ACCURACY: 0.86
MEAN VALIDATION ACCURACY: 0.87
MEAN TRAIN F1 SCORE: 0.85
MEAN VALIDATION F1 SCORE: 0.87
MEAN TRAIN PRECISION: 0.88
MEAN VALIDATION PRECISION: 0.88
MEAN TRAIN RECALL: 0.86
MEAN VALIDATION RECALL: 0.87

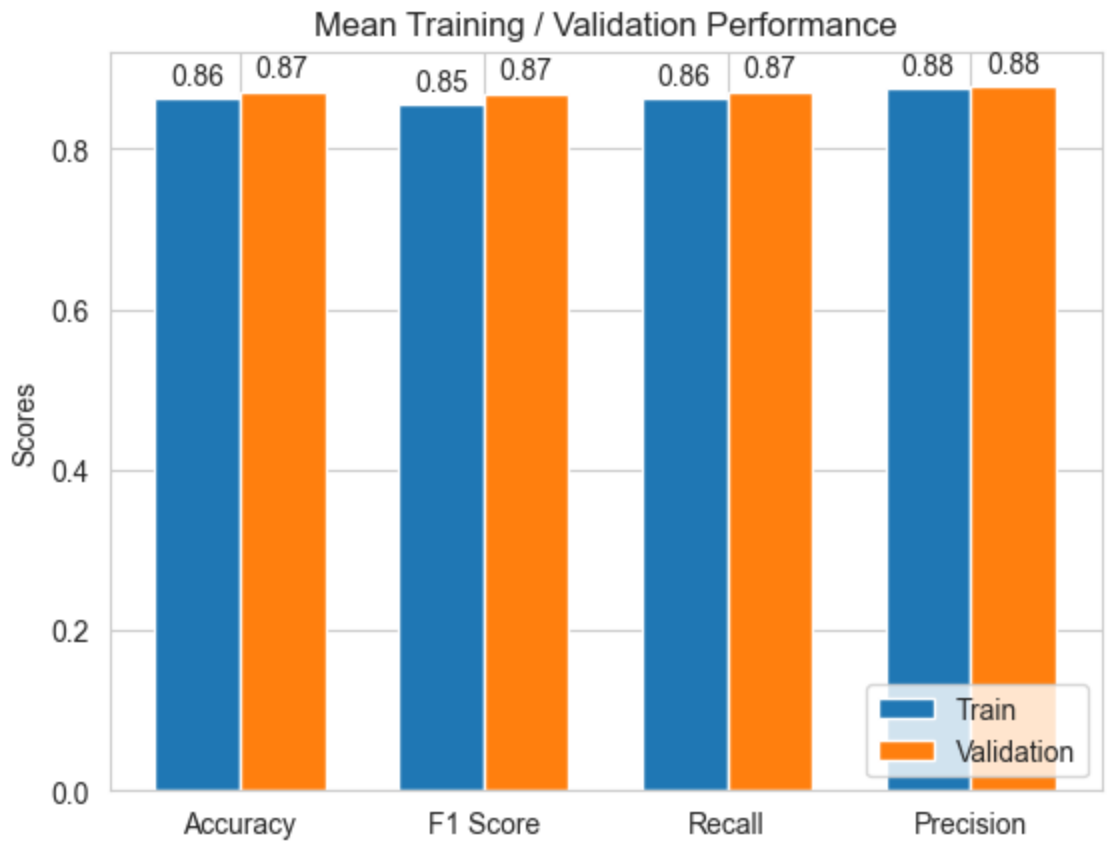


*Figure 20 Training / Validation (deep Neural Network with 2 layers)*

### 5.6.1 Neural network with 3 hidden layers

```
class Net_3(nn.Module):
    def __init__(self, D_in, H1, H2, H3, D_out):
        super(Net_3, self).__init__()
        self.fc1 = nn.Linear(D_in, H1)
        self.fc2 = nn.Linear(H1, H2)
        self.fc3 = nn.Linear(H2, H3)
        self.fc4 = nn.Linear(H3, D_out)

    def forward(self, x):
        x = torch.relu(self.fc1(x))
        x = torch.relu(self.fc2(x))
        x = torch.relu(self.fc3(x))
        x = torch.sigmoid(self.fc4(x))  # Sigmoid for binary classification
        return x
```

```
D_in, H1, H2, H3, D_out = 6, 128, 64, 32, 1
model = Net_3(D_in, H1, H2, H3, D_out)
num_epochs = 20
loss_func = nn.BCELoss() # Binary Cross Entropy Loss for binary classification
optimizer = optim.Adam(model.parameters(), lr=0.00001)
```

```
train_acc, valid_acc, train_f1, valid_f1, train_recall, valid_recall, train_precision, valid_precision = train(model, loss_func, optimizer, num_epochs, train_loader, test_loader)
```

Epoch 1/20, Train Acc: 0.60, Valid Acc: 0.79, Train F1: 0.53, Valid F1: 0.78, Train Precision: 0.77, Valid Precision: 0.84, Train Recall: 0.60, Valid Recall: 0.79
Epoch 2/20, Train Acc: 0.86, Valid Acc: 0.87, Train F1: 0.86, Valid F1: 0.87, Train Precision: 0.86, Valid Precision: 0.87, Train Recall: 0.86, Valid Recall: 0.87
Epoch 3/20, Train Acc: 0.87, Valid Acc: 0.86, Train F1: 0.87, Valid F1: 0.86, Train Precision: 0.87, Valid Precision: 0.87, Train Recall: 0.87, Valid Recall: 0.86
Epoch 4/20, Train Acc: 0.87, Valid Acc: 0.87, Train F1: 0.87, Valid F1: 0.87, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.87, Valid Recall: 0.87
Epoch 5/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 6/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 7/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 8/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 9/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 10/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 11/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 12/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 13/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 14/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 15/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 16/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 17/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 18/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 19/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 20/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
MEAN TRAIN ACCURACY: 0.87
MEAN VALIDATION ACCURACY: 0.88
MEAN TRAIN F1 SCORE: 0.86
MEAN VALIDATION F1 SCORE: 0.87
MEAN TRAIN PRECISION: 0.88
MEAN VALIDATION PRECISION: 0.88
MEAN TRAIN RECALL: 0.87
MEAN VALIDATION RECALL: 0.88

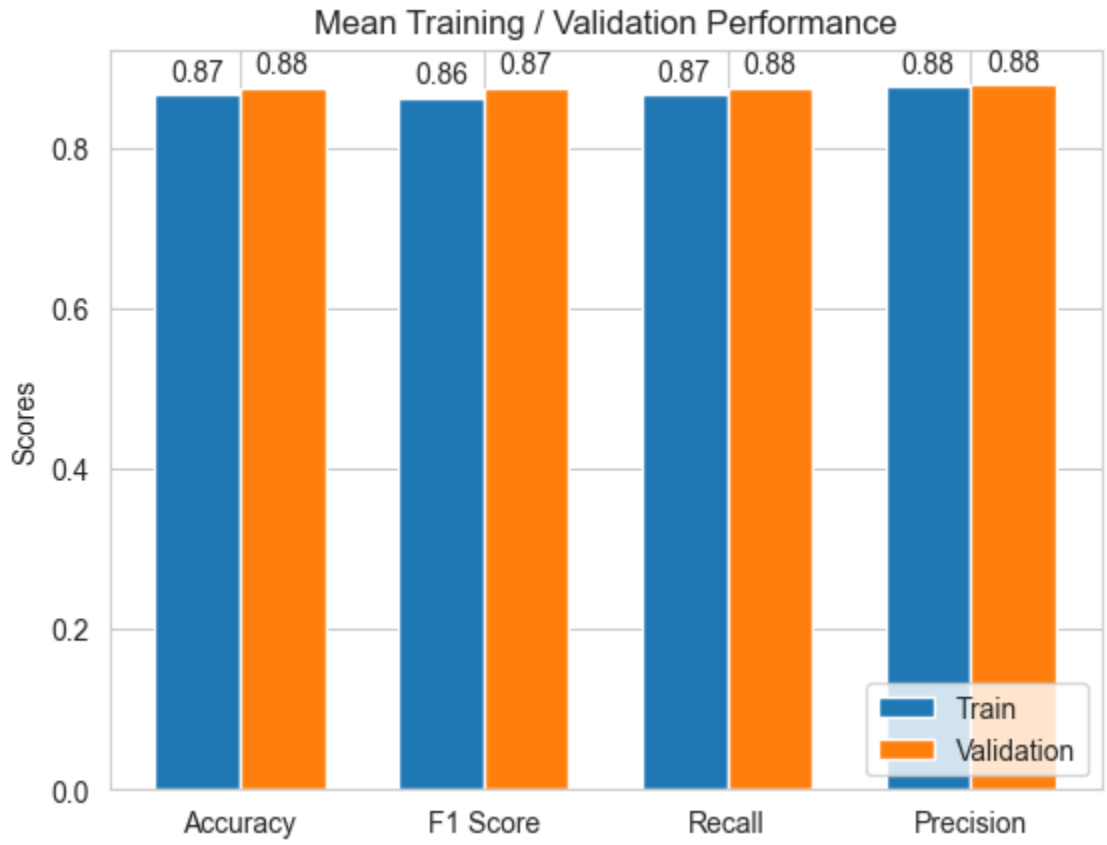


*Figure 21 Training / Validation NN 3 layers*

Παρατηρούμε πως πετυχαίνουμε παρόμοιο mean score στις μετρικές μας και ακόμα παρατηρούμε πως το μοντέλο μας έχει υψηλή προβλεπτική ικανότητα ανεξαρτήτως αν προσθέσουμε περισσότερα layers. Θα κρατήσουμε το NN με τα 3 hidden layers και δεν θα προσθέσουμε περισσότερα για να αποφύγουμε το overfitting **[46]**

### Πειράματα με το Learning Rate

```
class Net_lr(nn.Module):
    def __init__(self, D_in, H1, H2, H3, D_out):
        super(Net_lr, self).__init__()
        self.fc1 = nn.Linear(D_in, H1)
        self.fc2 = nn.Linear(H1, H2)
        self.fc3 = nn.Linear(H2, H3)
        self.fc4 = nn.Linear(H3, D_out)

    def forward(self, x):
        x = torch.relu(self.fc1(x))
        x = torch.relu(self.fc2(x))
        x = torch.relu(self.fc3(x))
        x = torch.sigmoid(self.fc4(x))  # Sigmoid for binary classification
        return x
```

```
D_in, H1, H2, H3, D_out = 6, 128, 64, 32, 1
num_epochs = 20
lr_ls = [0.1, 0.01, 0.001, 0.0001, 0.00001, 0.000001]
loss_func = nn.BCELoss() # Binary Cross Entropy Loss for binary classification
best_lr = {}
for idx, lr in enumerate(lr_ls):
    model = Net_lr(D_in, H1, H2, H3, D_out)
    optimizer = optim.Adam(model.parameters(), lr=lr)
    print(f'\n\nLEARNING RATE = {lr}\n\n')
    train_acc, valid_acc, train_f1, valid_f1, train_recall, valid_recall, train_precision, valid_precision =
train(model, loss_func, optimizer, num_epochs, train_loader, test_loader)

    best_lr[idx+1] = valid_acc
max_key = max(best_lr, key=best_lr.get)
print(f'The Learning Rate that gives best results is {lr_ls[max_key-1]}')
```

LEARNING RATE = 0.1

Epoch 1/20, Train Acc: 0.77, Valid Acc: 0.63, Train F1: 0.77, Valid F1: 0.57, Train Precision: 0.78, Valid Precision: 0.79, Train Recall: 0.77, Valid Recall: 0.63
Epoch 2/20, Train Acc: 0.77, Valid Acc: 0.79, Train F1: 0.77, Valid F1: 0.78, Train Precision: 0.77, Valid Precision: 0.85, Train Recall: 0.77, Valid Recall: 0.79
Epoch 3/20, Train Acc: 0.82, Valid Acc: 0.83, Train F1: 0.82, Valid F1: 0.83, Train Precision: 0.85, Valid Precision: 0.86, Train Recall: 0.82, Valid Recall: 0.83
Epoch 4/20, Train Acc: 0.83, Valid Acc: 0.84, Train F1: 0.83, Valid F1: 0.83, Train Precision: 0.86, Valid Precision: 0.86, Train Recall: 0.83, Valid Recall: 0.84
Epoch 5/20, Train Acc: 0.86, Valid Acc: 0.84, Train F1: 0.86, Valid F1: 0.83, Train Precision: 0.87, Valid Precision: 0.86, Train Recall: 0.86, Valid Recall: 0.84
Epoch 6/20, Train Acc: 0.80, Valid Acc: 0.79, Train F1: 0.79, Valid F1: 0.78, Train Precision: 0.85, Valid Precision: 0.85, Train Recall: 0.80, Valid Recall: 0.79
Epoch 7/20, Train Acc: 0.79, Valid Acc: 0.80, Train F1: 0.78, Valid F1: 0.79, Train Precision: 0.85, Valid Precision: 0.85, Train Recall: 0.79, Valid Recall: 0.80
Epoch 8/20, Train Acc: 0.80, Valid Acc: 0.80, Train F1: 0.80, Valid F1: 0.79, Train Precision: 0.85, Valid Precision: 0.85, Train Recall: 0.80, Valid Recall: 0.80
Epoch 9/20, Train Acc: 0.80, Valid Acc: 0.80, Train F1: 0.80, Valid F1: 0.79, Train Precision: 0.85, Valid Precision: 0.85, Train Recall: 0.80, Valid Recall: 0.80
Epoch 10/20, Train Acc: 0.80, Valid Acc: 0.80, Train F1: 0.80, Valid F1: 0.79, Train Precision: 0.85, Valid Precision: 0.85, Train Recall: 0.80, Valid Recall: 0.80
Epoch 11/20, Train Acc: 0.80, Valid Acc: 0.80, Train F1: 0.80, Valid F1: 0.79, Train Precision: 0.85, Valid Precision: 0.85, Train Recall: 0.80, Valid Recall: 0.80
Epoch 12/20, Train Acc: 0.80, Valid Acc: 0.80, Train F1: 0.80, Valid F1: 0.79, Train Precision: 0.85, Valid Precision: 0.85, Train Recall: 0.80, Valid Recall: 0.80
Epoch 13/20, Train Acc: 0.80, Valid Acc: 0.80, Train F1: 0.80, Valid F1: 0.79, Train Precision: 0.85, Valid Precision: 0.85, Train Recall: 0.80, Valid Recall: 0.80
Epoch 14/20, Train Acc: 0.80, Valid Acc: 0.80, Train F1: 0.80, Valid F1: 0.79, Train Precision: 0.85, Valid Precision: 0.85, Train Recall: 0.80, Valid Recall: 0.80
Epoch 15/20, Train Acc: 0.80, Valid Acc: 0.80, Train F1: 0.80, Valid F1: 0.79, Train Precision: 0.85, Valid Precision: 0.85, Train Recall: 0.80, Valid Recall: 0.80

Epoch 16/20, Train Acc: 0.82, Valid Acc: 0.88, Train F1: 0.81, Valid F1: 0.88, Train Precision: 0.85, Valid Precision: 0.88, Train Recall: 0.82, Valid Recall: 0.88
Epoch 17/20, Train Acc: 0.82, Valid Acc: 0.88, Train F1: 0.82, Valid F1: 0.88, Train Precision: 0.83, Valid Precision: 0.88, Train Recall: 0.82, Valid Recall: 0.88
Epoch 18/20, Train Acc: 0.87, Valid Acc: 0.88, Train F1: 0.87, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.87, Valid Recall: 0.88
Epoch 19/20, Train Acc: 0.87, Valid Acc: 0.88, Train F1: 0.87, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.87, Valid Recall: 0.88
Epoch 20/20, Train Acc: 0.87, Valid Acc: 0.88, Train F1: 0.87, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.87, Valid Recall: 0.88
...
MEAN VALIDATION PRECISION: 0.86
MEAN TRAIN RECALL: 0.82
MEAN VALIDATION RECALL: 0.82

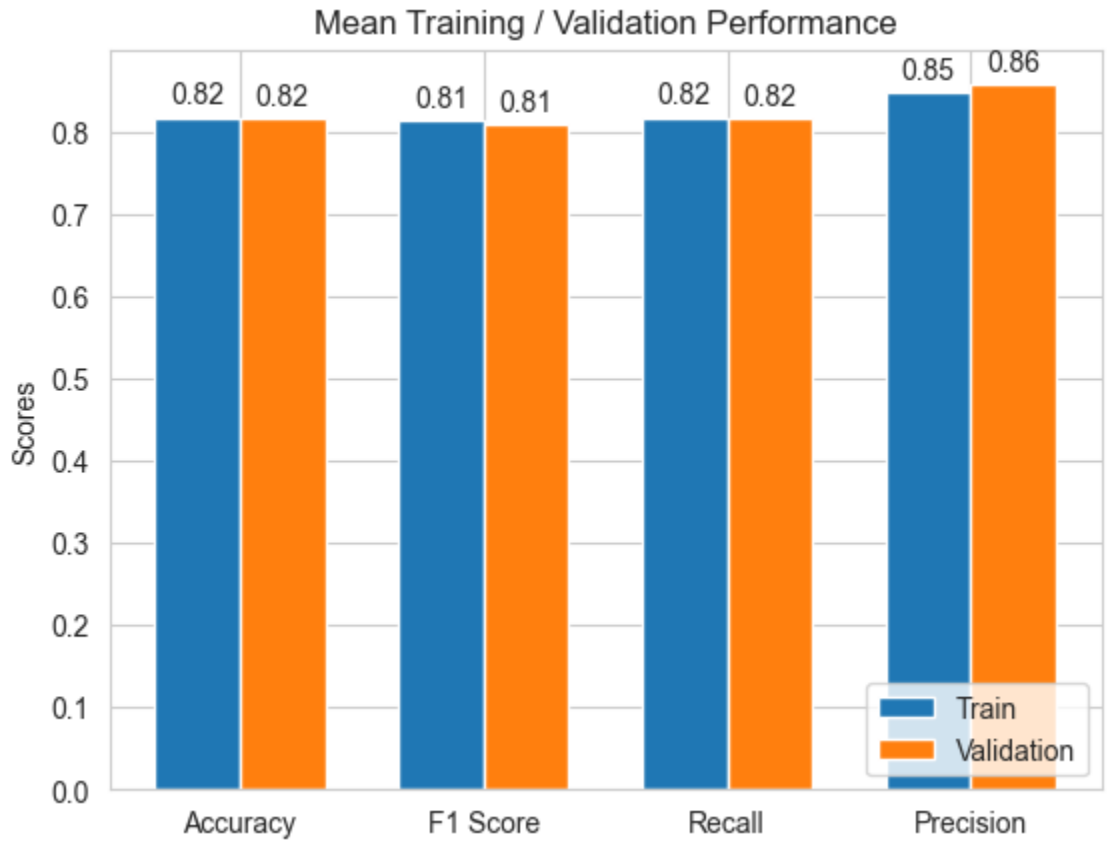


*Figure 22 Learning rate 0.1, Training / Validation*

LEARNING RATE = 0.01

Epoch 1/20, Train Acc: 0.89, Valid Acc: 0.90, Train F1: 0.89, Valid F1: 0.90, Train Precision: 0.89, Valid Precision: 0.90, Train Recall: 0.89, Valid Recall: 0.90
Epoch 2/20, Train Acc: 0.90, Valid Acc: 0.91, Train F1: 0.90, Valid F1: 0.90, Train Precision: 0.90, Valid Precision: 0.91, Train Recall: 0.90, Valid Recall: 0.91

Epoch 3/20, Train Acc: 0.90, Valid Acc: 0.91, Train F1: 0.90, Valid F1: 0.91, Train Precision: 0.90, Valid Precision: 0.91, Train Recall: 0.90, Valid Recall: 0.91
Epoch 4/20, Train Acc: 0.90, Valid Acc: 0.91, Train F1: 0.90, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.90, Valid Recall: 0.91
Epoch 5/20, Train Acc: 0.90, Valid Acc: 0.91, Train F1: 0.90, Valid F1: 0.91, Train Precision: 0.90, Valid Precision: 0.91, Train Recall: 0.90, Valid Recall: 0.91
Epoch 6/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.90, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 7/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 8/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 9/20, Train Acc: 0.91, Valid Acc: 0.90, Train F1: 0.91, Valid F1: 0.90, Train Precision: 0.91, Valid Precision: 0.90, Train Recall: 0.91, Valid Recall: 0.90
Epoch 10/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 11/20, Train Acc: 0.90, Valid Acc: 0.90, Train F1: 0.90, Valid F1: 0.90, Train Precision: 0.90, Valid Precision: 0.90, Train Recall: 0.90, Valid Recall: 0.90
Epoch 12/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 13/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.90, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 14/20, Train Acc: 0.91, Valid Acc: 0.90, Train F1: 0.91, Valid F1: 0.90, Train Precision: 0.91, Valid Precision: 0.90, Train Recall: 0.91, Valid Recall: 0.90
Epoch 15/20, Train Acc: 0.91, Valid Acc: 0.90, Train F1: 0.91, Valid F1: 0.90, Train Precision: 0.91, Valid Precision: 0.90, Train Recall: 0.91, Valid Recall: 0.90
Epoch 16/20, Train Acc: 0.91, Valid Acc: 0.90, Train F1: 0.91, Valid F1: 0.90, Train Precision: 0.91, Valid Precision: 0.90, Train Recall: 0.91, Valid Recall: 0.90
Epoch 17/20, Train Acc: 0.90, Valid Acc: 0.90, Train F1: 0.90, Valid F1: 0.90, Train Precision: 0.90, Valid Precision: 0.91, Train Recall: 0.90, Valid Recall: 0.90
Epoch 18/20, Train Acc: 0.91, Valid Acc: 0.90, Train F1: 0.91, Valid F1: 0.90, Train Precision: 0.91, Valid Precision: 0.90, Train Recall: 0.91, Valid Recall: 0.90
Epoch 19/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 20/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
...
MEAN VALIDATION PRECISION: 0.91
MEAN TRAIN RECALL: 0.90
MEAN VALIDATION RECALL: 0.90

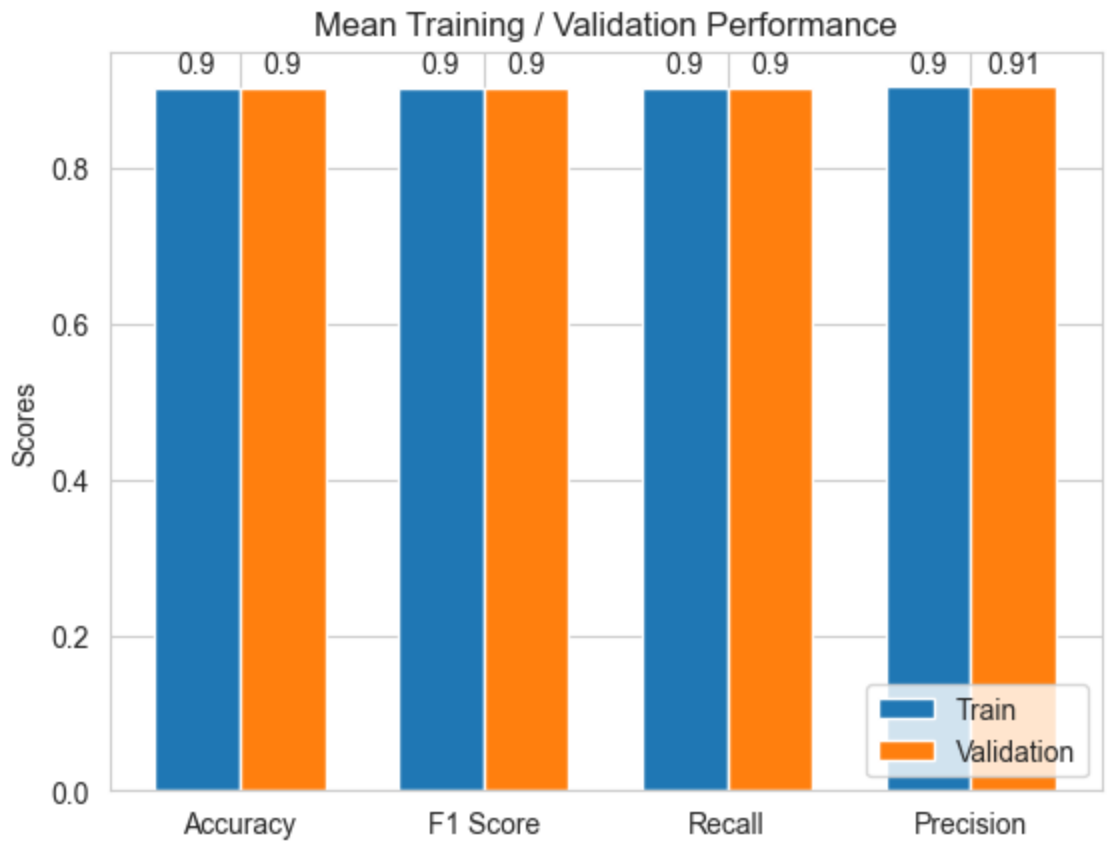


*Figure 23 Learning rate 0.01, Training / Validation*

LEARNING RATE = 0.001

Epoch 1/20, Train Acc: 0.88, Valid Acc: 0.89, Train F1: 0.88, Valid F1: 0.89, Train Precision: 0.88, Valid Precision: 0.89, Train Recall: 0.88, Valid Recall: 0.89
Epoch 2/20, Train Acc: 0.89, Valid Acc: 0.90, Train F1: 0.89, Valid F1: 0.90, Train Precision: 0.89, Valid Precision: 0.90, Train Recall: 0.89, Valid Recall: 0.90
Epoch 3/20, Train Acc: 0.90, Valid Acc: 0.91, Train F1: 0.90, Valid F1: 0.91, Train Precision: 0.90, Valid Precision: 0.91, Train Recall: 0.90, Valid Recall: 0.91
Epoch 4/20, Train Acc: 0.90, Valid Acc: 0.91, Train F1: 0.90, Valid F1: 0.91, Train Precision: 0.90, Valid Precision: 0.91, Train Recall: 0.90, Valid Recall: 0.91
Epoch 5/20, Train Acc: 0.90, Valid Acc: 0.91, Train F1: 0.90, Valid F1: 0.91, Train Precision: 0.90, Valid Precision: 0.91, Train Recall: 0.90, Valid Recall: 0.91
Epoch 6/20, Train Acc: 0.90, Valid Acc: 0.91, Train F1: 0.90, Valid F1: 0.91, Train Precision: 0.90, Valid Precision: 0.91, Train Recall: 0.90, Valid Recall: 0.91
Epoch 7/20, Train Acc: 0.90, Valid Acc: 0.91, Train F1: 0.90, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.90, Valid Recall: 0.91
Epoch 8/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 9/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 10/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91

Epoch 11/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 12/20, Train Acc: 0.90, Valid Acc: 0.91, Train F1: 0.90, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.90, Valid Recall: 0.91
Epoch 13/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 14/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 15/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 16/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 17/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 18/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 19/20, Train Acc: 0.91, Valid Acc: 0.90, Train F1: 0.91, Valid F1: 0.90, Train Precision: 0.91, Valid Precision: 0.90, Train Recall: 0.91, Valid Recall: 0.90
Epoch 20/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
...
MEAN VALIDATION PRECISION: 0.91
MEAN TRAIN RECALL: 0.90
MEAN VALIDATION RECALL: 0.91

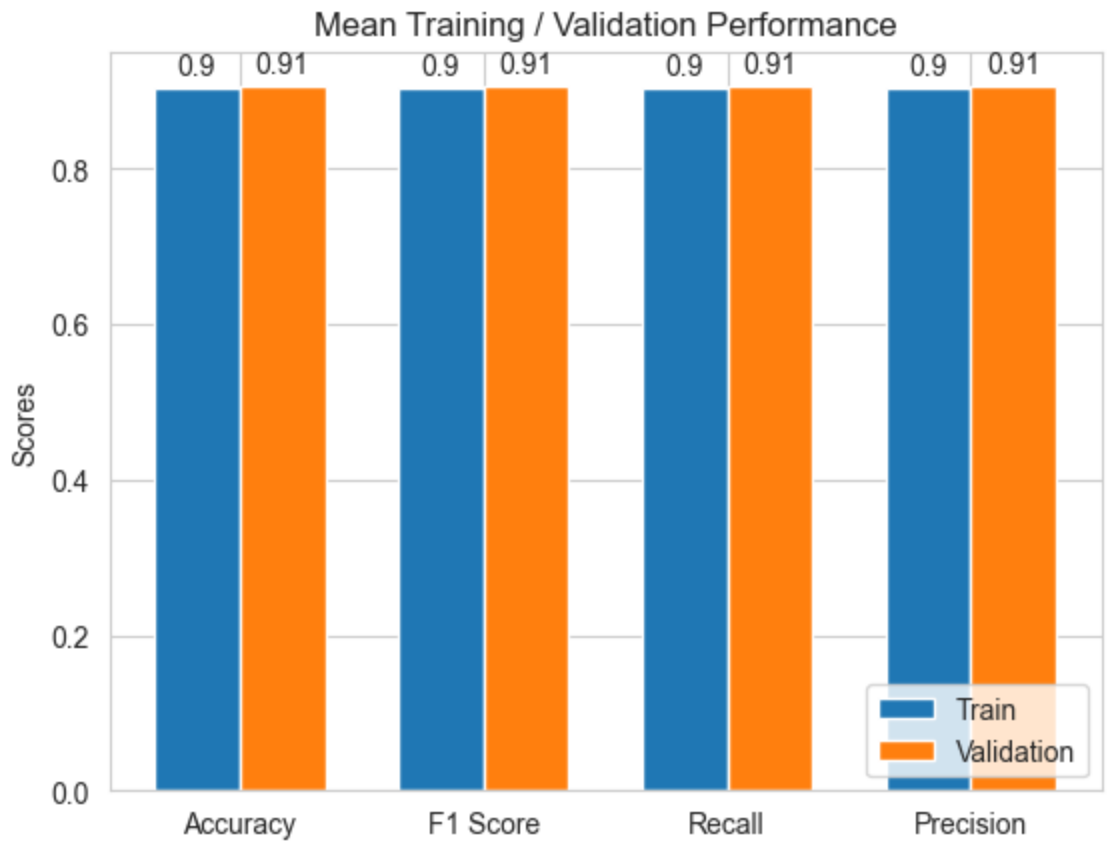


*Figure 24 Learning rate 0.001, Training / Validation*

Epoch 1/20, Train Acc: 0.77, Valid Acc: 0.63, Train F1: 0.77, Valid F1: 0.57, Train Precision: 0.78, Valid Precision: 0.79, Train Recall: 0.77, Valid Recall: 0.63
Epoch 2/20, Train Acc: 0.77, Valid Acc: 0.79, Train F1: 0.77, Valid F1: 0.78, Train Precision: 0.77, Valid Precision: 0.85, Train Recall: 0.77, Valid Recall: 0.79
Epoch 3/20, Train Acc: 0.82, Valid Acc: 0.83, Train F1: 0.82, Valid F1: 0.83, Train Precision: 0.85, Valid Precision: 0.86, Train Recall: 0.82, Valid Recall: 0.83
Epoch 4/20, Train Acc: 0.83, Valid Acc: 0.84, Train F1: 0.83, Valid F1: 0.83, Train Precision: 0.86, Valid Precision: 0.86, Train Recall: 0.83, Valid Recall: 0.84
Epoch 5/20, Train Acc: 0.86, Valid Acc: 0.84, Train F1: 0.86, Valid F1: 0.83, Train Precision: 0.87, Valid Precision: 0.86, Train Recall: 0.86, Valid Recall: 0.84
Epoch 6/20, Train Acc: 0.80, Valid Acc: 0.79, Train F1: 0.79, Valid F1: 0.78, Train Precision: 0.85, Valid Precision: 0.85, Train Recall: 0.80, Valid Recall: 0.79
Epoch 7/20, Train Acc: 0.79, Valid Acc: 0.80, Train F1: 0.78, Valid F1: 0.79, Train Precision: 0.85, Valid Precision: 0.85, Train Recall: 0.79, Valid Recall: 0.80
Epoch 8/20, Train Acc: 0.80, Valid Acc: 0.80, Train F1: 0.80, Valid F1: 0.79, Train Precision: 0.85, Valid Precision: 0.85, Train Recall: 0.80, Valid Recall: 0.80
Epoch 9/20, Train Acc: 0.80, Valid Acc: 0.80, Train F1: 0.80, Valid F1: 0.79, Train Precision: 0.85, Valid Precision: 0.85, Train Recall: 0.80, Valid Recall: 0.80
Epoch 10/20, Train Acc: 0.80, Valid Acc: 0.80, Train F1: 0.80, Valid F1: 0.79, Train Precision: 0.85, Valid Precision: 0.85, Train Recall: 0.80, Valid Recall: 0.80

Epoch 11/20, Train Acc: 0.80, Valid Acc: 0.80, Train F1: 0.80, Valid F1: 0.79, Train Precision: 0.85, Valid Precision: 0.85, Train Recall: 0.80, Valid Recall: 0.80
Epoch 12/20, Train Acc: 0.80, Valid Acc: 0.80, Train F1: 0.80, Valid F1: 0.79, Train Precision: 0.85, Valid Precision: 0.85, Train Recall: 0.80, Valid Recall: 0.80
Epoch 13/20, Train Acc: 0.80, Valid Acc: 0.80, Train F1: 0.80, Valid F1: 0.79, Train Precision: 0.85, Valid Precision: 0.85, Train Recall: 0.80, Valid Recall: 0.80
Epoch 14/20, Train Acc: 0.80, Valid Acc: 0.80, Train F1: 0.80, Valid F1: 0.79, Train Precision: 0.85, Valid Precision: 0.85, Train Recall: 0.80, Valid Recall: 0.80
Epoch 15/20, Train Acc: 0.80, Valid Acc: 0.80, Train F1: 0.80, Valid F1: 0.79, Train Precision: 0.85, Valid Precision: 0.85, Train Recall: 0.80, Valid Recall: 0.80
Epoch 16/20, Train Acc: 0.82, Valid Acc: 0.88, Train F1: 0.81, Valid F1: 0.88, Train Precision: 0.85, Valid Precision: 0.88, Train Recall: 0.82, Valid Recall: 0.88
Epoch 17/20, Train Acc: 0.82, Valid Acc: 0.88, Train F1: 0.82, Valid F1: 0.88, Train Precision: 0.83, Valid Precision: 0.88, Train Recall: 0.82, Valid Recall: 0.88
Epoch 18/20, Train Acc: 0.87, Valid Acc: 0.88, Train F1: 0.87, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.87, Valid Recall: 0.88
Epoch 19/20, Train Acc: 0.87, Valid Acc: 0.88, Train F1: 0.87, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.87, Valid Recall: 0.88
Epoch 20/20, Train Acc: 0.87, Valid Acc: 0.88, Train F1: 0.87, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.87, Valid Recall: 0.88
...
MEAN VALIDATION PRECISION: 0.86
MEAN TRAIN RECALL: 0.82
MEAN VALIDATION RECALL: 0.8

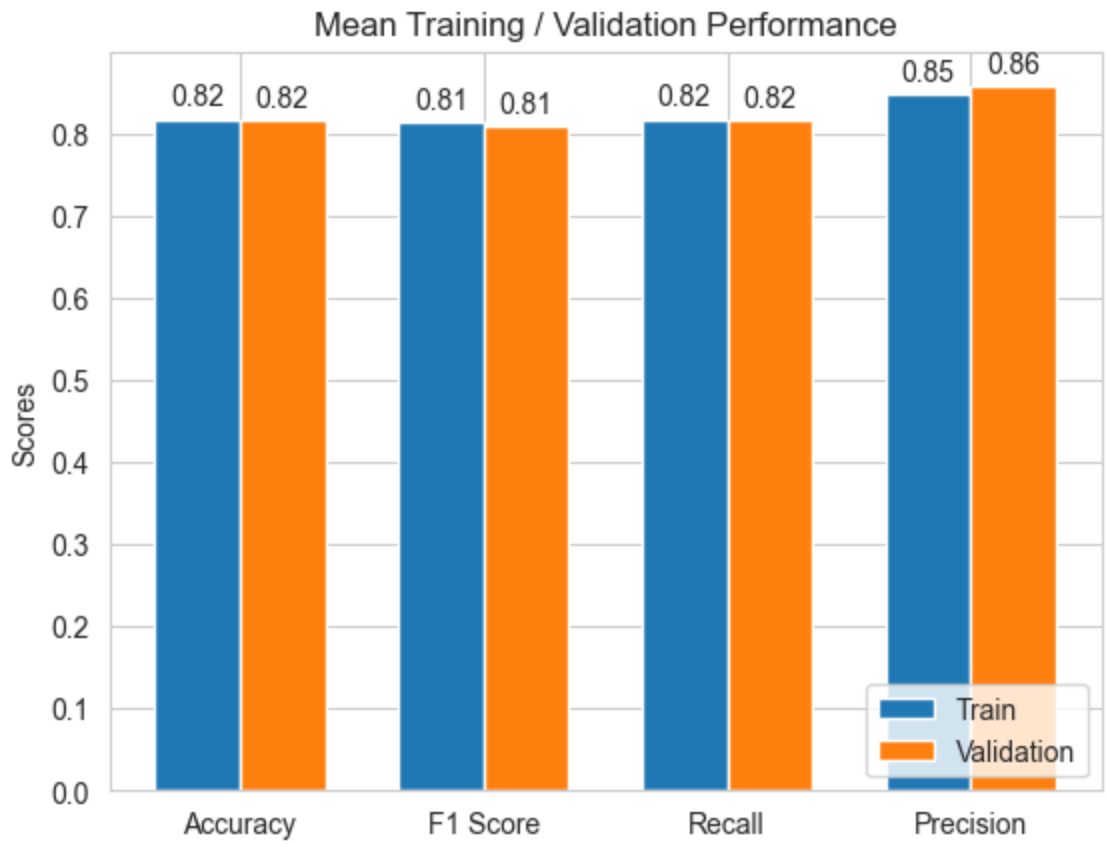


*Figure 25 Training / Validation*

LEARNING RATE = 0.0001

Epoch 1/20, Train Acc: 0.83, Valid Acc: 0.89, Train F1: 0.83, Valid F1: 0.89, Train Precision: 0.84, Valid Precision: 0.89, Train Recall: 0.83, Valid Recall: 0.89
Epoch 2/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 3/20, Train Acc: 0.88, Valid Acc: 0.88, Train F1: 0.88, Valid F1: 0.88, Train Precision: 0.88, Valid Precision: 0.88, Train Recall: 0.88, Valid Recall: 0.88
Epoch 4/20, Train Acc: 0.89, Valid Acc: 0.89, Train F1: 0.89, Valid F1: 0.89, Train Precision: 0.89, Valid Precision: 0.89, Train Recall: 0.89, Valid Recall: 0.89
Epoch 5/20, Train Acc: 0.89, Valid Acc: 0.89, Train F1: 0.89, Valid F1: 0.89, Train Precision: 0.89, Valid Precision: 0.89, Train Recall: 0.89, Valid Recall: 0.89
Epoch 6/20, Train Acc: 0.89, Valid Acc: 0.89, Train F1: 0.89, Valid F1: 0.89, Train Precision: 0.89, Valid Precision: 0.89, Train Recall: 0.89, Valid Recall: 0.89
Epoch 7/20, Train Acc: 0.89, Valid Acc: 0.89, Train F1: 0.89, Valid F1: 0.89, Train Precision: 0.89, Valid Precision: 0.89, Train Recall: 0.89, Valid Recall: 0.89
Epoch 8/20, Train Acc: 0.89, Valid Acc: 0.89, Train F1: 0.89, Valid F1: 0.89, Train Precision: 0.89, Valid Precision: 0.89, Train Recall: 0.89, Valid Recall: 0.89
Epoch 9/20, Train Acc: 0.89, Valid Acc: 0.89, Train F1: 0.89, Valid F1: 0.89, Train Precision: 0.89, Valid Precision: 0.89, Train Recall: 0.89, Valid Recall: 0.89

Epoch 10/20, Train Acc: 0.89, Valid Acc: 0.89, Train F1: 0.89, Valid F1: 0.89, Train Precision: 0.89, Valid Precision: 0.89, Train Recall: 0.89, Valid Recall: 0.89
Epoch 11/20, Train Acc: 0.89, Valid Acc: 0.90, Train F1: 0.89, Valid F1: 0.89, Train Precision: 0.89, Valid Precision: 0.90, Train Recall: 0.89, Valid Recall: 0.90
Epoch 12/20, Train Acc: 0.89, Valid Acc: 0.89, Train F1: 0.89, Valid F1: 0.89, Train Precision: 0.89, Valid Precision: 0.89, Train Recall: 0.89, Valid Recall: 0.89
Epoch 13/20, Train Acc: 0.89, Valid Acc: 0.90, Train F1: 0.89, Valid F1: 0.90, Train Precision: 0.89, Valid Precision: 0.90, Train Recall: 0.89, Valid Recall: 0.90
Epoch 14/20, Train Acc: 0.90, Valid Acc: 0.90, Train F1: 0.90, Valid F1: 0.90, Train Precision: 0.90, Valid Precision: 0.90, Train Recall: 0.90, Valid Recall: 0.90
Epoch 15/20, Train Acc: 0.90, Valid Acc: 0.90, Train F1: 0.90, Valid F1: 0.90, Train Precision: 0.90, Valid Precision: 0.90, Train Recall: 0.90, Valid Recall: 0.90
Epoch 16/20, Train Acc: 0.90, Valid Acc: 0.90, Train F1: 0.90, Valid F1: 0.90, Train Precision: 0.90, Valid Precision: 0.90, Train Recall: 0.90, Valid Recall: 0.90
Epoch 17/20, Train Acc: 0.90, Valid Acc: 0.90, Train F1: 0.90, Valid F1: 0.90, Train Precision: 0.90, Valid Precision: 0.90, Train Recall: 0.90, Valid Recall: 0.90
Epoch 18/20, Train Acc: 0.90, Valid Acc: 0.90, Train F1: 0.90, Valid F1: 0.90, Train Precision: 0.90, Valid Precision: 0.90, Train Recall: 0.90, Valid Recall: 0.90
Epoch 19/20, Train Acc: 0.90, Valid Acc: 0.90, Train F1: 0.90, Valid F1: 0.90, Train Precision: 0.90, Valid Precision: 0.90, Train Recall: 0.90, Valid Recall: 0.90
Epoch 20/20, Train Acc: 0.90, Valid Acc: 0.90, Train F1: 0.90, Valid F1: 0.90, Train Precision: 0.90, Valid Precision: 0.90, Train Recall: 0.90, Valid Recall: 0.90
MEAN TRAIN ACCURACY: 0.89
MEAN VALIDATION ACCURACY: 0.89
MEAN TRAIN F1 SCORE: 0.89
MEAN VALIDATION F1 SCORE: 0.89
MEAN TRAIN PRECISION: 0.89
MEAN VALIDATION PRECISION: 0.89
MEAN TRAIN RECALL: 0.89
MEAN VALIDATION RECALL: 0.89

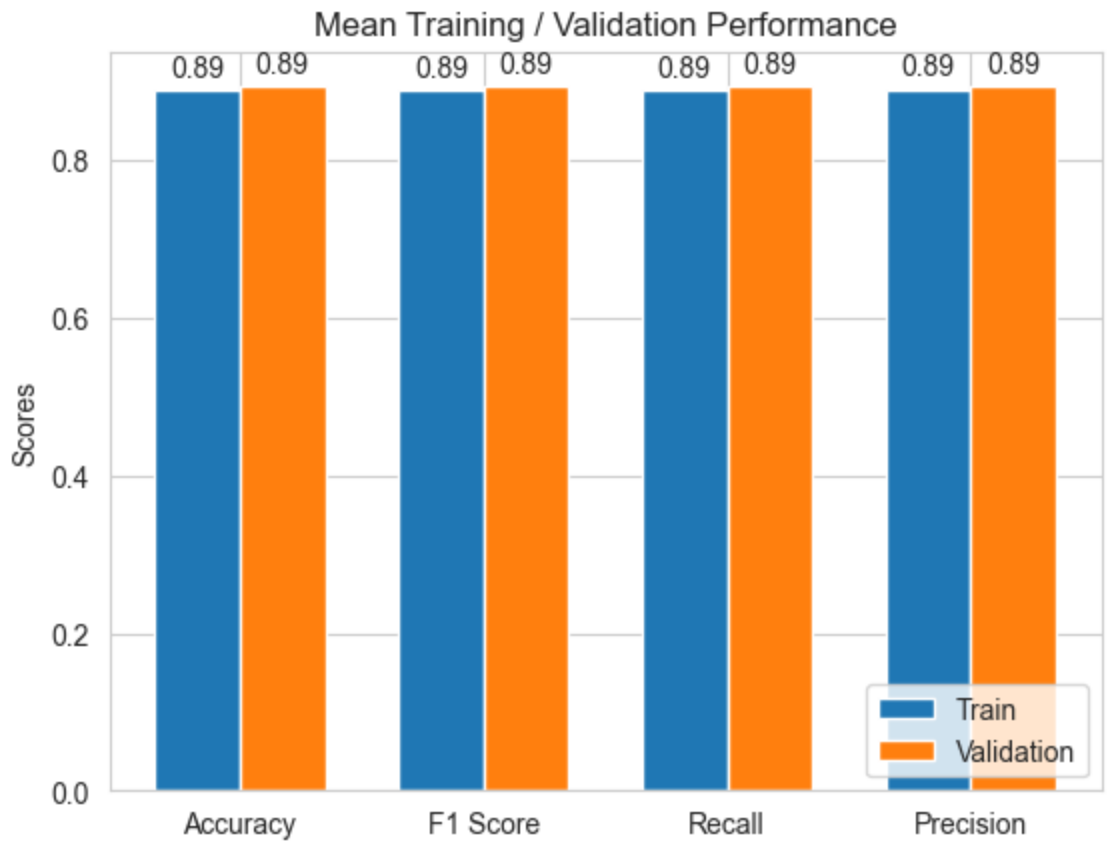


*Figure 28, Learning rate 0.0001, Training / Validation*

LEARNING RATE = 1e-06

Epoch 1/20, Train Acc: 0.50, Valid Acc: 0.50, Train F1: 0.33, Valid F1: 0.33, Train Precision: 0.25, Valid Precision: 0.25, Train Recall: 0.50, Valid Recall: 0.50
Epoch 2/20, Train Acc: 0.50, Valid Acc: 0.50, Train F1: 0.33, Valid F1: 0.33, Train Precision: 0.25, Valid Precision: 0.25, Train Recall: 0.50, Valid Recall: 0.50
Epoch 3/20, Train Acc: 0.50, Valid Acc: 0.50, Train F1: 0.33, Valid F1: 0.33, Train Precision: 0.25, Valid Precision: 0.25, Train Recall: 0.50, Valid Recall: 0.50
Epoch 4/20, Train Acc: 0.50, Valid Acc: 0.50, Train F1: 0.33, Valid F1: 0.33, Train Precision: 0.25, Valid Precision: 0.25, Train Recall: 0.50, Valid Recall: 0.50
Epoch 5/20, Train Acc: 0.50, Valid Acc: 0.50, Train F1: 0.34, Valid F1: 0.34, Train Precision: 0.75, Valid Precision: 0.75, Train Recall: 0.50, Valid Recall: 0.50
Epoch 6/20, Train Acc: 0.52, Valid Acc: 0.53, Train F1: 0.37, Valid F1: 0.40, Train Precision: 0.75, Valid Precision: 0.76, Train Recall: 0.52, Valid Recall: 0.53
Epoch 7/20, Train Acc: 0.56, Valid Acc: 0.58, Train F1: 0.46, Valid F1: 0.50, Train Precision: 0.76, Valid Precision: 0.77, Train Recall: 0.56, Valid Recall: 0.58
Epoch 8/20, Train Acc: 0.62, Valid Acc: 0.64, Train F1: 0.55, Valid F1: 0.59, Train Precision: 0.78, Valid Precision: 0.79, Train Recall: 0.62, Valid Recall: 0.64

Epoch 9/20, Train Acc: 0.67, Valid Acc: 0.68, Train F1: 0.63, Valid F1: 0.65, Train Precision: 0.80, Valid Precision: 0.80, Train Recall: 0.67, Valid Recall: 0.68
Epoch 10/20, Train Acc: 0.71, Valid Acc: 0.72, Train F1: 0.68, Valid F1: 0.70, Train Precision: 0.81, Valid Precision: 0.81, Train Recall: 0.71, Valid Recall: 0.72
Epoch 11/20, Train Acc: 0.74, Valid Acc: 0.75, Train F1: 0.72, Valid F1: 0.74, Train Precision: 0.82, Valid Precision: 0.83, Train Recall: 0.74, Valid Recall: 0.75
Epoch 12/20, Train Acc: 0.77, Valid Acc: 0.78, Train F1: 0.76, Valid F1: 0.77, Train Precision: 0.83, Valid Precision: 0.84, Train Recall: 0.77, Valid Recall: 0.78
Epoch 13/20, Train Acc: 0.79, Valid Acc: 0.80, Train F1: 0.78, Valid F1: 0.79, Train Precision: 0.84, Valid Precision: 0.85, Train Recall: 0.79, Valid Recall: 0.80
Epoch 14/20, Train Acc: 0.80, Valid Acc: 0.81, Train F1: 0.80, Valid F1: 0.81, Train Precision: 0.85, Valid Precision: 0.85, Train Recall: 0.80, Valid Recall: 0.81
Epoch 15/20, Train Acc: 0.82, Valid Acc: 0.83, Train F1: 0.81, Valid F1: 0.82, Train Precision: 0.85, Valid Precision: 0.86, Train Recall: 0.82, Valid Recall: 0.83
Epoch 16/20, Train Acc: 0.83, Valid Acc: 0.84, Train F1: 0.83, Valid F1: 0.83, Train Precision: 0.86, Valid Precision: 0.86, Train Recall: 0.83, Valid Recall: 0.84
Epoch 17/20, Train Acc: 0.84, Valid Acc: 0.85, Train F1: 0.84, Valid F1: 0.85, Train Precision: 0.86, Valid Precision: 0.87, Train Recall: 0.84, Valid Recall: 0.85
Epoch 18/20, Train Acc: 0.85, Valid Acc: 0.85, Train F1: 0.85, Valid F1: 0.85, Train Precision: 0.87, Valid Precision: 0.87, Train Recall: 0.85, Valid Recall: 0.85
Epoch 19/20, Train Acc: 0.86, Valid Acc: 0.86, Train F1: 0.86, Valid F1: 0.86, Train Precision: 0.87, Valid Precision: 0.88, Train Recall: 0.86, Valid Recall: 0.86
Epoch 20/20, Train Acc: 0.86, Valid Acc: 0.86, Train F1: 0.86, Valid F1: 0.86, Train Precision: 0.87, Valid Precision: 0.87, Train Recall: 0.86, Valid Recall: 0.86
...
MEAN VALIDATION PRECISION: 0.71
MEAN TRAIN RECALL: 0.69
MEAN VALIDATION RECALL: 0.69

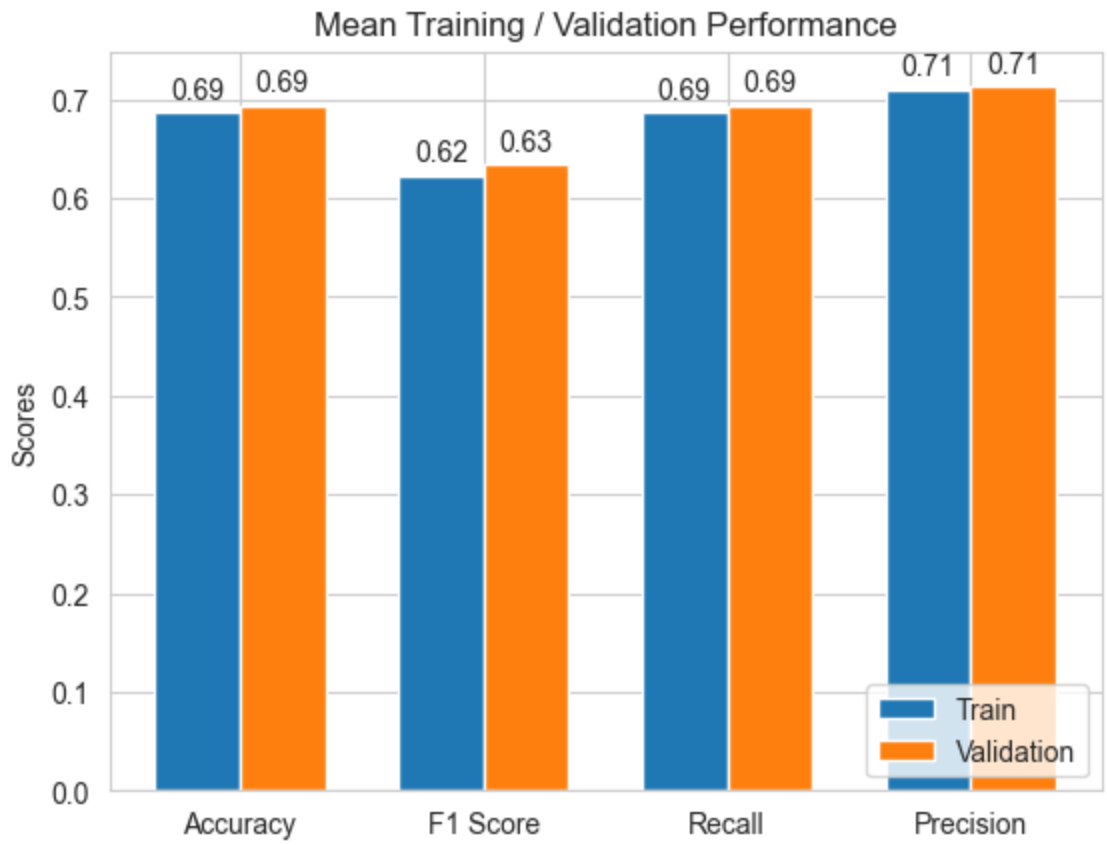


*Figure 29, Learning rate 1e-06, Training / Validation*

**The Learning Rate that gives best results is 0.01**

### 5.6.2 Πειράματα με Optimizers

```
class Net_opt(nn.Module):
    def __init__(self, D_in, H1, H2, H3, D_out):
        super(Net_opt, self).__init__()
        self.fc1 = nn.Linear(D_in, H1)
        self.fc2 = nn.Linear(H1, H2)
        self.fc3 = nn.Linear(H2, H3)
        self.fc4 = nn.Linear(H3, D_out)

    def forward(self, x):
        x = torch.relu(self.fc1(x))
        x = torch.relu(self.fc2(x))
        x = torch.relu(self.fc3(x))
        x = torch.sigmoid(self.fc4(x))  # Sigmoid for binary classification
        return x
```

```
D_in, H1, H2, H3, D_out = 6, 128, 64, 32, 1
num_epochs = 20
lr = 0.01
loss_func = nn.BCELoss()

optimizers = {
    'Adam': optim.Adam,
    'SGD': optim.SGD,
    'RMSprop': optim.RMSprop,
    'Adagrad': optim.Adagrad
}

best_optimizer_results = {}

for optimizer_name, optimizer_func in optimizers.items():
    model = Net_opt(D_in, H1, H2, H3, D_out)
    optimizer = optimizer_func(model.parameters(), lr=lr)
    print(f'\n\nOPTIMIZER = {optimizer_name}\n\n')
    train_acc, valid_acc, train_f1, valid_f1, train_recall, valid_recall, train_precision, valid_precision =
train(model, loss_func, optimizer, num_epochs, train_loader, test_loader)

    best_optimizer_results[optimizer_name] = {
        'Validation Accuracy': max(valid_acc),
        'Validation F1': max(valid_f1)
    }

best_optimizer = max(best_optimizer_results, key=lambda x: best_optimizer_results[x]['Validation
F1'])
print(f'The Optimizer that gives best results is {best_optimizer} with Validation Accuracy:
{best_optimizer_results[best_optimizer]["Validation Accuracy"]}')
```

- **OPTIMIZER = Adam**

Reference: **[47]**

Epoch 1/20, Train Acc: 0.88, Valid Acc: 0.90, Train F1: 0.88, Valid F1: 0.90, Train Precision: 0.89, Valid Precision: 0.90, Train Recall: 0.88, Valid Recall: 0.90
Epoch 2/20, Train Acc: 0.90, Valid Acc: 0.90, Train F1: 0.90, Valid F1: 0.90, Train Precision: 0.90, Valid Precision: 0.91, Train Recall: 0.90, Valid Recall: 0.90
Epoch 3/20, Train Acc: 0.90, Valid Acc: 0.90, Train F1: 0.90, Valid F1: 0.90, Train Precision: 0.90, Valid Precision: 0.90, Train Recall: 0.90, Valid Recall: 0.90
Epoch 4/20, Train Acc: 0.90, Valid Acc: 0.90, Train F1: 0.90, Valid F1: 0.90, Train Precision: 0.90, Valid Precision: 0.90, Train Recall: 0.90, Valid Recall: 0.90
Epoch 5/20, Train Acc: 0.90, Valid Acc: 0.91, Train F1: 0.90, Valid F1: 0.91, Train Precision: 0.90, Valid Precision: 0.91, Train Recall: 0.90, Valid Recall: 0.91
Epoch 6/20, Train Acc: 0.90, Valid Acc: 0.91, Train F1: 0.90, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.90, Valid Recall: 0.91
Epoch 7/20, Train Acc: 0.90, Valid Acc: 0.91, Train F1: 0.90, Valid F1: 0.91, Train Precision: 0.90, Valid Precision: 0.91, Train Recall: 0.90, Valid Recall: 0.91
Epoch 8/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 9/20, Train Acc: 0.90, Valid Acc: 0.90, Train F1: 0.90, Valid F1: 0.90, Train Precision: 0.91, Valid Precision: 0.90, Train Recall: 0.90, Valid Recall: 0.90
Epoch 10/20, Train Acc: 0.91, Valid Acc: 0.90, Train F1: 0.91, Valid F1: 0.90, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.90
Epoch 11/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 12/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 13/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 14/20, Train Acc: 0.91, Valid Acc: 0.90, Train F1: 0.91, Valid F1: 0.90, Train Precision: 0.91, Valid Precision: 0.90, Train Recall: 0.91, Valid Recall: 0.90
Epoch 15/20, Train Acc: 0.90, Valid Acc: 0.90, Train F1: 0.90, Valid F1: 0.90, Train Precision: 0.90, Valid Precision: 0.91, Train Recall: 0.90, Valid Recall: 0.90
Epoch 16/20, Train Acc: 0.91, Valid Acc: 0.90, Train F1: 0.91, Valid F1: 0.90, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.90
Epoch 17/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 18/20, Train Acc: 0.91, Valid Acc: 0.90, Train F1: 0.91, Valid F1: 0.90, Train Precision: 0.91, Valid Precision: 0.90, Train Recall: 0.91, Valid Recall: 0.90
Epoch 19/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 20/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
...
MEAN VALIDATION PRECISION: 0.91
MEAN TRAIN RECALL: 0.90
MEAN VALIDATION RECALL: 0.90

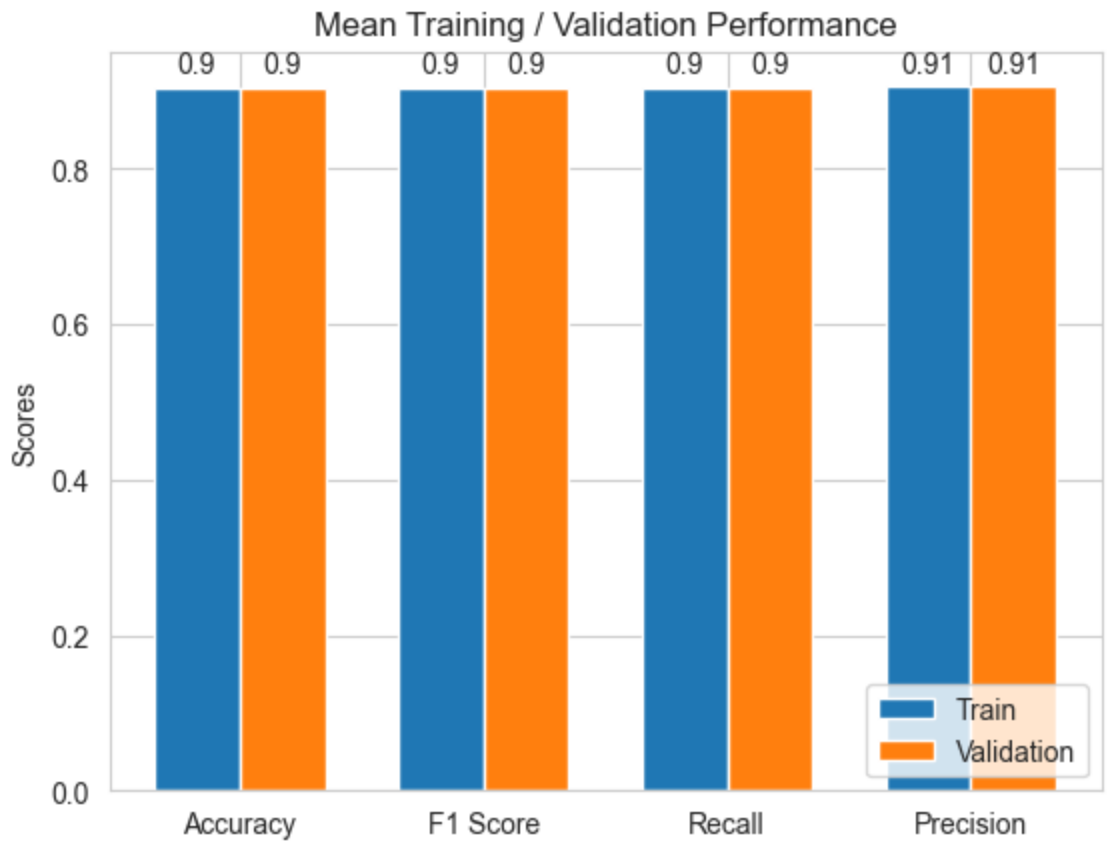


*Figure 30 Adam optimizer, Training / Validation*

- **OPTIMIZER = SGD**

Reference: **[48]**

Epoch 1/20, Train Acc: 0.79, Valid Acc: 0.88, Train F1: 0.78, Valid F1: 0.88, Train Precision: 0.81, Valid Precision: 0.88, Train Recall: 0.79, Valid Recall: 0.88
Epoch 2/20, Train Acc: 0.88, Valid Acc: 0.89, Train F1: 0.88, Valid F1: 0.89, Train Precision: 0.88, Valid Precision: 0.89, Train Recall: 0.88, Valid Recall: 0.89
Epoch 3/20, Train Acc: 0.89, Valid Acc: 0.88, Train F1: 0.89, Valid F1: 0.88, Train Precision: 0.89, Valid Precision: 0.88, Train Recall: 0.89, Valid Recall: 0.88
Epoch 4/20, Train Acc: 0.88, Valid Acc: 0.89, Train F1: 0.88, Valid F1: 0.89, Train Precision: 0.88, Valid Precision: 0.89, Train Recall: 0.88, Valid Recall: 0.89
Epoch 5/20, Train Acc: 0.88, Valid Acc: 0.89, Train F1: 0.88, Valid F1: 0.89, Train Precision: 0.88, Valid Precision: 0.89, Train Recall: 0.88, Valid Recall: 0.89
Epoch 6/20, Train Acc: 0.89, Valid Acc: 0.88, Train F1: 0.89, Valid F1: 0.88, Train Precision: 0.89, Valid Precision: 0.88, Train Recall: 0.89, Valid Recall: 0.88
Epoch 7/20, Train Acc: 0.89, Valid Acc: 0.88, Train F1: 0.89, Valid F1: 0.88, Train Precision: 0.89, Valid Precision: 0.88, Train Recall: 0.89, Valid Recall: 0.88
Epoch 8/20, Train Acc: 0.89, Valid Acc: 0.89, Train F1: 0.89, Valid F1: 0.89, Train Precision: 0.89, Valid Precision: 0.89, Train Recall: 0.89, Valid Recall: 0.89

Epoch 9/20, Train Acc: 0.89, Valid Acc: 0.89, Train F1: 0.89, Valid F1: 0.89, Train Precision: 0.89, Valid Precision: 0.89, Train Recall: 0.89, Valid Recall: 0.89
Epoch 10/20, Train Acc: 0.89, Valid Acc: 0.88, Train F1: 0.89, Valid F1: 0.88, Train Precision: 0.89, Valid Precision: 0.88, Train Recall: 0.89, Valid Recall: 0.88
Epoch 11/20, Train Acc: 0.88, Valid Acc: 0.89, Train F1: 0.88, Valid F1: 0.89, Train Precision: 0.88, Valid Precision: 0.89, Train Recall: 0.88, Valid Recall: 0.89
Epoch 12/20, Train Acc: 0.89, Valid Acc: 0.89, Train F1: 0.89, Valid F1: 0.89, Train Precision: 0.89, Valid Precision: 0.89, Train Recall: 0.89, Valid Recall: 0.89
Epoch 13/20, Train Acc: 0.89, Valid Acc: 0.89, Train F1: 0.88, Valid F1: 0.89, Train Precision: 0.89, Valid Precision: 0.89, Train Recall: 0.89, Valid Recall: 0.89
Epoch 14/20, Train Acc: 0.89, Valid Acc: 0.89, Train F1: 0.89, Valid F1: 0.89, Train Precision: 0.89, Valid Precision: 0.89, Train Recall: 0.89, Valid Recall: 0.89
Epoch 15/20, Train Acc: 0.89, Valid Acc: 0.89, Train F1: 0.89, Valid F1: 0.89, Train Precision: 0.89, Valid Precision: 0.89, Train Recall: 0.89, Valid Recall: 0.89
Epoch 16/20, Train Acc: 0.89, Valid Acc: 0.89, Train F1: 0.89, Valid F1: 0.89, Train Precision: 0.89, Valid Precision: 0.89, Train Recall: 0.89, Valid Recall: 0.89
Epoch 17/20, Train Acc: 0.89, Valid Acc: 0.88, Train F1: 0.89, Valid F1: 0.88, Train Precision: 0.89, Valid Precision: 0.89, Train Recall: 0.89, Valid Recall: 0.88
Epoch 18/20, Train Acc: 0.89, Valid Acc: 0.89, Train F1: 0.89, Valid F1: 0.89, Train Precision: 0.89, Valid Precision: 0.89, Train Recall: 0.89, Valid Recall: 0.89
Epoch 19/20, Train Acc: 0.89, Valid Acc: 0.88, Train F1: 0.89, Valid F1: 0.88, Train Precision: 0.89, Valid Precision: 0.89, Train Recall: 0.89, Valid Recall: 0.88
Epoch 20/20, Train Acc: 0.89, Valid Acc: 0.89, Train F1: 0.89, Valid F1: 0.89, Train Precision: 0.89, Valid Precision: 0.89, Train Recall: 0.89, Valid Recall: 0.89
...
MEAN VALIDATION PRECISION: 0.89
MEAN TRAIN RECALL: 0.88
MEAN VALIDATION RECALL: 0.89

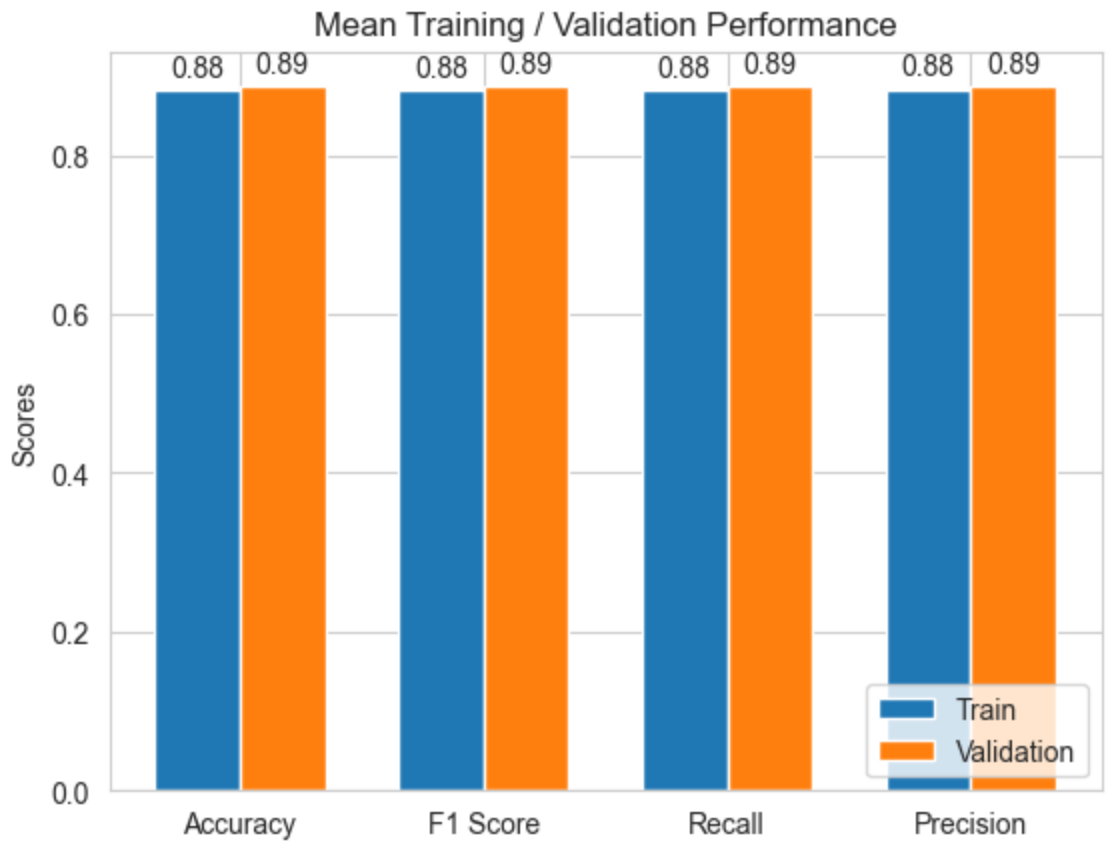


*Figure 31, SGD optimizer, Training / Validation*

- **OPTIMIZER = RMSprop**

Reference: **[49]**

Epoch 1/20, Train Acc: 0.89, Valid Acc: 0.90, Train F1: 0.89, Valid F1: 0.90, Train Precision: 0.89, Valid Precision: 0.90, Train Recall: 0.89, Valid Recall: 0.90
Epoch 2/20, Train Acc: 0.90, Valid Acc: 0.90, Train F1: 0.90, Valid F1: 0.90, Train Precision: 0.90, Valid Precision: 0.90, Train Recall: 0.90, Valid Recall: 0.90
Epoch 3/20, Train Acc: 0.90, Valid Acc: 0.91, Train F1: 0.90, Valid F1: 0.91, Train Precision: 0.90, Valid Precision: 0.91, Train Recall: 0.90, Valid Recall: 0.91
Epoch 4/20, Train Acc: 0.90, Valid Acc: 0.91, Train F1: 0.90, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.90, Valid Recall: 0.91
Epoch 5/20, Train Acc: 0.90, Valid Acc: 0.90, Train F1: 0.90, Valid F1: 0.90, Train Precision: 0.90, Valid Precision: 0.91, Train Recall: 0.90, Valid Recall: 0.90
Epoch 6/20, Train Acc: 0.91, Valid Acc: 0.89, Train F1: 0.91, Valid F1: 0.89, Train Precision: 0.91, Valid Precision: 0.89, Train Recall: 0.91, Valid Recall: 0.89
Epoch 7/20, Train Acc: 0.90, Valid Acc: 0.91, Train F1: 0.90, Valid F1: 0.91, Train Precision: 0.90, Valid Precision: 0.91, Train Recall: 0.90, Valid Recall: 0.91
Epoch 8/20, Train Acc: 0.90, Valid Acc: 0.91, Train F1: 0.90, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.90, Valid Recall: 0.91
Epoch 9/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91

Epoch 10/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 11/20, Train Acc: 0.90, Valid Acc: 0.90, Train F1: 0.90, Valid F1: 0.90, Train Precision: 0.90, Valid Precision: 0.90, Train Recall: 0.90, Valid Recall: 0.90
Epoch 12/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 13/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 14/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 15/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 16/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 17/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 18/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.90, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 19/20, Train Acc: 0.90, Valid Acc: 0.90, Train F1: 0.90, Valid F1: 0.90, Train Precision: 0.90, Valid Precision: 0.90, Train Recall: 0.90, Valid Recall: 0.90
Epoch 20/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
...
MEAN VALIDATION PRECISION: 0.91
MEAN TRAIN RECALL: 0.90
MEAN VALIDATION RECALL: 0.91

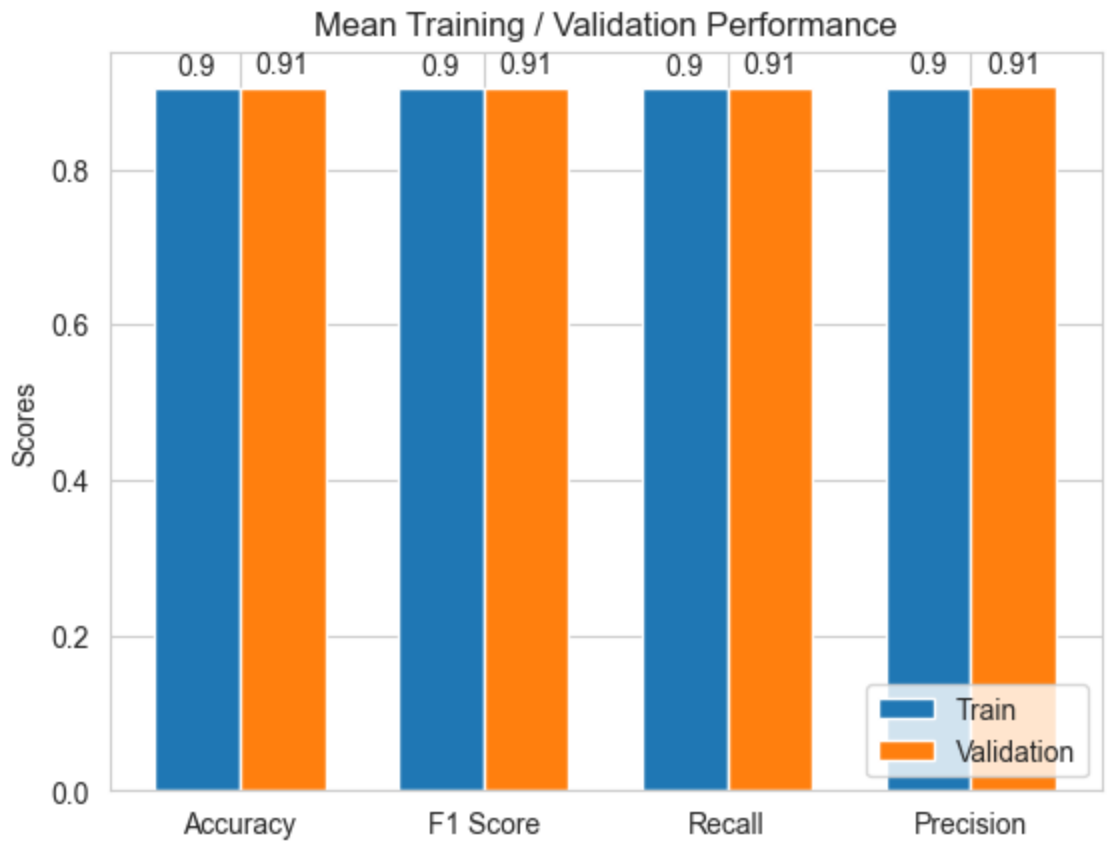


*Figure 32 Training / Validation (RMSprop)*

- **OPTIMIZER = Adagrad**

Reference: **[50]**

Epoch 1/20, Train Acc: 0.88, Valid Acc: 0.89, Train F1: 0.88, Valid F1: 0.89, Train Precision: 0.88, Valid Precision: 0.89, Train Recall: 0.88, Valid Recall: 0.89
Epoch 2/20, Train Acc: 0.89, Valid Acc: 0.89, Train F1: 0.89, Valid F1: 0.89, Train Precision: 0.89, Valid Precision: 0.89, Train Recall: 0.89, Valid Recall: 0.89
Epoch 3/20, Train Acc: 0.89, Valid Acc: 0.90, Train F1: 0.89, Valid F1: 0.90, Train Precision: 0.89, Valid Precision: 0.90, Train Recall: 0.89, Valid Recall: 0.90
Epoch 4/20, Train Acc: 0.90, Valid Acc: 0.90, Train F1: 0.90, Valid F1: 0.90, Train Precision: 0.90, Valid Precision: 0.90, Train Recall: 0.90, Valid Recall: 0.90
Epoch 5/20, Train Acc: 0.90, Valid Acc: 0.90, Train F1: 0.90, Valid F1: 0.90, Train Precision: 0.90, Valid Precision: 0.90, Train Recall: 0.90, Valid Recall: 0.90
Epoch 6/20, Train Acc: 0.90, Valid Acc: 0.90, Train F1: 0.90, Valid F1: 0.90, Train Precision: 0.90, Valid Precision: 0.90, Train Recall: 0.90, Valid Recall: 0.90
Epoch 7/20, Train Acc: 0.90, Valid Acc: 0.90, Train F1: 0.90, Valid F1: 0.90, Train Precision: 0.90, Valid Precision: 0.90, Train Recall: 0.90, Valid Recall: 0.90
Epoch 8/20, Train Acc: 0.90, Valid Acc: 0.90, Train F1: 0.90, Valid F1: 0.90, Train Precision: 0.90, Valid Precision: 0.90, Train Recall: 0.90, Valid Recall: 0.90
Epoch 9/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91,

Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 10/20, Train Acc: 0.91, Valid Acc: 0.90, Train F1: 0.90, Valid F1: 0.90, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.90
Epoch 11/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 12/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 13/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 14/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 15/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 16/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 17/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 18/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 19/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
Epoch 20/20, Train Acc: 0.91, Valid Acc: 0.91, Train F1: 0.91, Valid F1: 0.91, Train Precision: 0.91, Valid Precision: 0.91, Train Recall: 0.91, Valid Recall: 0.91
...
MEAN VALIDATION PRECISION: 0.90
MEAN TRAIN RECALL: 0.90
MEAN VALIDATION RECALL: 0.90

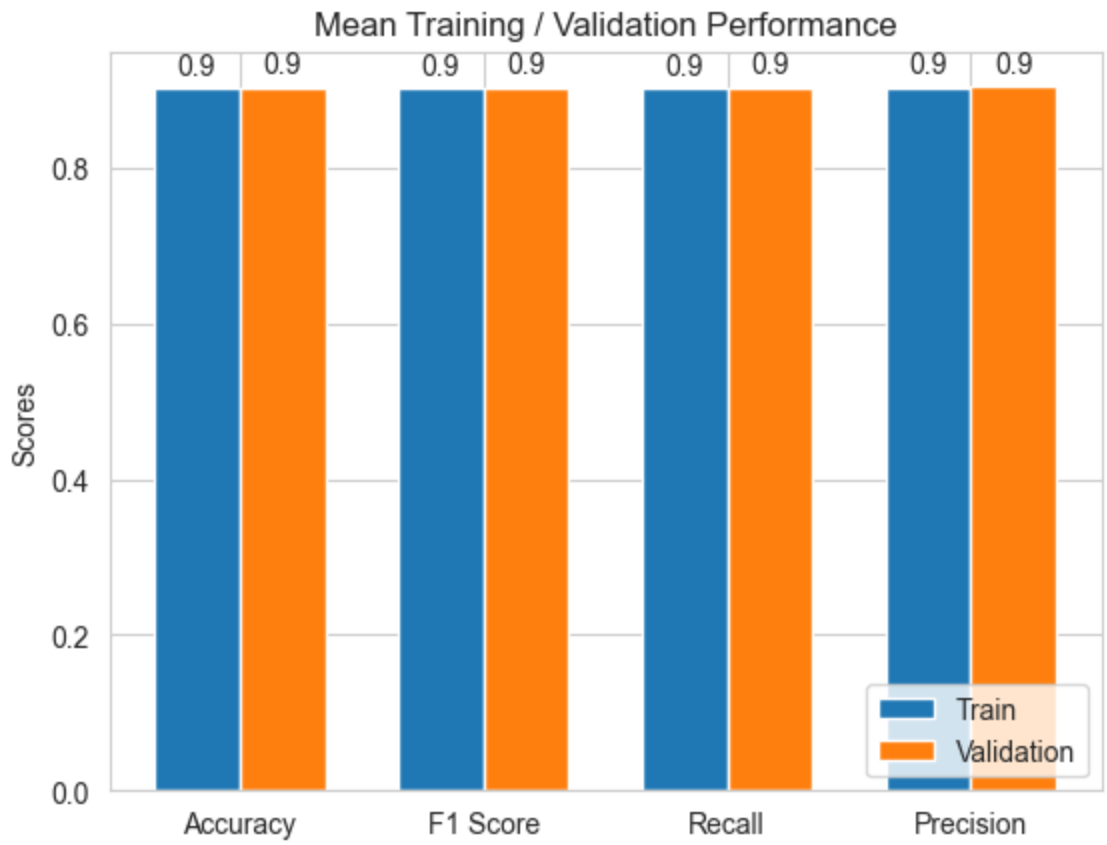


*Figure 33 Training / Validation (Adagrad)*

**The Optimizer that gives best results is RMSprop with Validation Accuracy: 0.9088235294117647**

# 6. Επίλογος

Η παρούσα διπλωματική εργασία ανέδειξε την αξία των σύγχρονων τεχνικών μηχανικής μάθησης και deep learning στην πρόβλεψη του διαβήτη, χρησιμοποιώντας ένα εκτενές σύνολο δεδομένων που περιλάμβανε 100,000 παρατηρήσεις και 9 χαρακτηριστικά. Μέσα από την ανάλυση πολλαπλών αλγορίθμων, από κλασικές προσεγγίσεις όπως Logistic Regression και KNN έως πιο εξελιγμένες τεχνικές όπως AdaBoost και νευρωνικά δίκτυα, αξιολογήθηκαν τα αποτελέσματα και η αποτελεσματικότητα των μοντέλων.

Η συνδυαστική χρήση διαφορετικών μεθόδων επέτρεψε την εις βάθος διερεύνηση των παραγόντων που επηρεάζουν την πρόβλεψη του διαβήτη και παρείχε χρήσιμες πληροφορίες για τη βέλτιστη εφαρμογή τεχνολογιών πρόβλεψης στην ιατρική πρακτική. Τα αποτελέσματα έδειξαν ότι τα νευρωνικά δίκτυα με τον optimizer Adam και learning rate 0.001 προσέφεραν την καλύτερη απόδοση, με υψηλό Validation Accuracy και F1 score, καθιστώντας αυτή την προσέγγιση την πιο κατάλληλη για το συγκεκριμένο πρόβλημα.

Η ανάλυση αυτή επιβεβαίωσε ότι η αξιοποίηση πολυδιάστατων μοντέλων και η προσεκτική επιλογή παραμέτρων μπορεί να οδηγήσει σε σημαντικές βελτιώσεις στην ακρίβεια πρόβλεψης. Επίσης, υπογράμμισε την ανάγκη για προσεκτική επιλογή μεθόδων, καθώς τεχνικές όπως το ElasticNet απέδειξαν περιορισμένη ικανότητα πρόβλεψης για το συγκεκριμένο dataset.

Συμπερασματικά, αυτή η έρευνα συμβάλλει στον τομέα της πρόβλεψης διαβήτη, θέτοντας τις βάσεις για περαιτέρω αναλύσεις και βελτιστοποιήσεις. Η χρήση εξελιγμένων μεθόδων μηχανικής μάθησης μπορεί να έχει σημαντική επίδραση στη βελτίωση της ποιότητας ζωής των ατόμων που διατρέχουν κίνδυνο, υποστηρίζοντας πιο ακριβείς και έγκαιρες διαγνωστικές προσεγγίσεις.

# 7. Βιβλιογραφία


[1] "Machine learning based diabetes prediction and development of smart web application," *ScienceDirect*. [Online]. Διαθέσιμο: https://www.sciencedirect.com/science/article/pii/S1877050920300557.

[2] "Types of Diabetes," *Diabetes UK*. [Online]. Διαθέσιμο: https://www.diabetes.org.uk/diabetes-the-basics/types-of-diabetes.

[3] "Παγκόσμια Ημέρα Διαβήτη: Απειλή για το 10% του ελληνικού πληθυσμού," *Η Καθημερινή*. [Online]. Διαθέσιμο: https://www.kathimerini.gr/life/health/561583198.

[4] "Machine learning based diabetes prediction and development of smart web application," *ScienceDirect*. [Online]. Διαθέσιμο: https://www.sciencedirect.com/science/article/pii/S1877050920300557.

[5] "A survey on semi-supervised learning," *PMC*. [Online]. Διαθέσιμο: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/.

[6] "A Survey of Ensemble Learning Techniques," *IEEE Xplore*. [Online]. Διαθέσιμο: https://ieeexplore.ieee.org/document/9952107.

[7] "Supervised Machine Learning," *DataCamp*. [Online]. Διαθέσιμο: https://www.datacamp.com/courses/supervised-learning.

[8] "Unsupervised learning," *Wikipedia*. [Online]. Διαθέσιμο: https://en.wikipedia.org/wiki/Unsupervised_learning.

[9] "Reinforcement learning," *Wikipedia*. [Online]. Διαθέσιμο: https://en.wikipedia.org/wiki/Reinforcement_learning.

[10] "A survey on semi-supervised learning," *Springer*. [Online]. Διαθέσιμο: https://link.springer.com/article/10.1007/s42979-021-00815-1.

[11] "Self-supervised Learning: A Succinct Review," *PMC*. [Online]. Διαθέσιμο: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10107388/.

[12] "A Comprehensive Review on Multiple Instance Learning," *MDPI*. [Online]. Διαθέσιμο: https://www.mdpi.com/topics/computer-science/few-shot-learning.

[13] "Few-Shot Learning," *ScienceDirect*. [Online]. Διαθέσιμο: https://www.sciencedirect.com/topics/computer-science/few-shot-learning.

[14] *arXiv*. [Online]. Διαθέσιμο: https://arxiv.org/abs/1707.00600.

[15] "Ensemble Learning," *Wikipedia*. [Online]. Διαθέσιμο: https://en.wikipedia.org/wiki/Ensemble_learning.

[16] "Gini Index for Decision Trees," *UpGrad*. [Online]. Διαθέσιμο: https://www.upgrad.com/blog/gini-index-for-decision-trees/.

[17] "What is Overfitting?" *AWS*. [Online]. Διαθέσιμο: https://aws.amazon.com/what-is/overfitting/.

[18] "RandomForestClassifier," *Scikit-Learn Documentation*. [Online]. Διαθέσιμο: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html.

[19] "Support Vector Machine," *Wikipedia*. [Online]. Διαθέσιμο: https://en.wikipedia.org/wiki/Support_vector_machine.

[20] "A Deep Learning-Based System for Predicting Diabetes," *IEEE Xplore*. [Online]. Διαθέσιμο: https://ieeexplore.ieee.org/abstract/document/10199539.

[21] "Types of Neural Networks," *My Great Learning*. [Online]. Διαθέσιμο: https://www.mygreatlearning.com/blog/types-of-neural-networks/.

[22] "Types of Neural Networks," *KnowledgeHut*. [Online]. Διαθέσιμο: https://www.knowledgehut.com/blog/data-science/types-of-neural-networks.

[23] "The Mostly Complete Chart of Neural Networks Explained," *Towards Data Science*. [Online]. Διαθέσιμο: https://towardsdatascience.com/the-mostly-complete-chart-of-neural-networks-explained-3fb6f2367464.

[24] "A Review on Deep Learning Approaches in Diabetes Prediction," *Springer*. [Online]. Διαθέσιμο: https://link.springer.com/article/10.1007/s42979-021-00815-1.

[25] *Pandas Documentation*. [Online]. Διαθέσιμο: https://pandas.pydata.org.

[26] *Matplotlib Documentation*. [Online]. Διαθέσιμο: https://matplotlib.org.

[27] *Numpy Documentation*. [Online]. Διαθέσιμο: https://numpy.org.

[28] *Scikit-Learn Documentation*. [Online]. Διαθέσιμο: https://scikit-learn.org.

[29] *Pytorch Documentation*. [Online]. Διαθέσιμο: https://pytorch.org.

[30] "What is Correlation?" *JMP*. [Online]. Διαθέσιμο: https://www.jmp.com/en_au/statistics-knowledge-portal/what-is-correlation.html.

[31] "Diabetes Prediction," *Kaggle*. [Online]. Διαθέσιμο: https://www.kaggle.com/code/mvanshika/diabetes-prediction.

[32] "Feature Selection Techniques in Machine Learning," *Towards Data Science*. [Online]. Διαθέσιμο: https://towardsdatascience.com/feature-selection-techniques-in-machine-learning-with-python-f24e7da3f36e.

[33] "What is Gradient Descent?" *Khan Academy*. [Online]. Διαθέσιμο: https://www.khanacademy.org/math/multivariable-calculus/applications-of-multivariable-derivatives/optimizing-multivariable-functions/a/what-is-gradient-descent.

[34] "Min-Max Scaling," *Mlxtend Documentation*. [Online]. Διαθέσιμο: https://rasbt.github.io/mlxtend/user_guide/preprocessing/minmax_scaling.

[35] "Logistic Regression," *ScienceDirect*. [Online]. Διαθέσιμο: https://www.sciencedirect.com/topics/computer-science/logistic-regression.

[36] *Διπλωματική Εργασία*, Πανεπιστήμιο Αιγαίου. [Online]. Διαθέσιμο: https://hellanicus.lib.aegean.gr/bitstream/handle/11610/18309.

[37] "Math Behind Logistic Regression," *LinkedIn*. [Online]. Διαθέσιμο: https://www.linkedin.com/pulse/math-behind-logistic-regression-how-fine-tune-santosh-kamble.

[38] "Fine-tuning Logistic Regression," *LinkedIn*. [Online]. Διαθέσιμο: https://www.linkedin.com/pulse/math-behind-logistic-regression-how-fine-tune-santosh-kamble.

[39] "AdaBoostClassifier," *Scikit-Learn Documentation*. [Online]. Διαθέσιμο: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html.

[40] "AdaBoost in Python," *DataCamp*. [Online]. Διαθέσιμο: https://www.datacamp.com/tutorial/adaboost-classifier-python.

[41] "BernoulliNB," *Scikit-Learn Documentation*. [Online]. Διαθέσιμο: https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.BernoulliNB.html.

[42] "Elastic Net Regularization," *Wikipedia*. [Online]. Διαθέσιμο: https://en.wikipedia.org/wiki/Elastic_net_regularization.

[43] "Unlocking the Full Potential of LLM," *Databricks*. [Online]. Διαθέσιμο: https://www.databricks.com/resources/ebook/tap-full-potential-llm.

[44] "One Hidden Layer Neural Networks," *Medium*. [Online]. Διαθέσιμο: https://medium.com/data-science-365/one-hidden-layer-shallow-neural-network-architecture-d45097f649e6.

[45] "Deep Neural Networks with Multiple Layers," *Medium*. [Online]. Διαθέσιμο: https://medium.com/data-science-365/two-or-more-hidden-layers-deep-neural-network-architecture-9824523ab903.

[46] "Intro to Neural Networks," *Brilliant*. [Online]. Διαθέσιμο: https://brilliant.org/courses/intro-neural-networks.

[47] "Adam Optimization Algorithm for Deep Learning," *Machine Learning Mastery*. [Online]. Διαθέσιμο: https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning.

[48] "Stochastic Gradient Descent," *GeeksforGeeks*. [Online]. Διαθέσιμο: https://www.geeksforgeeks.org/ml-stochastic-gradient-descent-sgd.

[49] "RMSprop Optimizer," *Keras Documentation*. [Online]. Διαθέσιμο: https://keras.io/api/optimizers/rmsprop.

[50] "Adagrad Overview," *Databricks*. [Online]. Διαθέσιμο: https://www.databricks.com/glossary/adagrad.