ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ

UNIVERSITY OF PIRAEUS

«Σχολή»

«Πρόγραμμα Σπουδών»

Μεταπτυχιακή Διπλωματική Εργασία

Υλοποίηση Πολυεπίπεδων Perceptrons για Διάγνωση Καρκίνου του Μαστού

«Κωνσταντία Μαρία Σιακαβέλλα»

AM:ME1823

Επιβλέπων καθηγητής: «Μιχαήλ Φιλιππάκης»

«Αττική», «Ιούλιος» «2021»

«Ευχαριστίες»

Η παρούσα εργασία αποτελεί διπλωματική εργασία στα πλαίσια του μεταπτυχιακού προγράμματος «Πληροφοριακά Συστήματα & Υπηρεσίες» του τμήματος Ψηφιακών Συστημάτων. Πριν την παρουσίαση των αποτελεσμάτων της παρούσας διπλωματικής εργασίας, αισθάνομαι την υποχρέωση να ευχαριστήσω ορισμένους από τους ανθρώπους που γνώρισα, συνεργάστηκα μαζί τους και έπαιξαν πολύ σημαντικό ρόλο στην πραγματοποίηση της. Πρώτο από όλους θέλω να ευχαριστήσω τον επιβλέποντα καθηγητή της διπλωματικής εργασίας, Καθηγητή Μιχαήλ Φιλιππάκη για την πολύτιμη καθοδήγηση , εμπιστοσύνη και εκτίμηση που μου έδειξε, καθώς και ότι μου στάθηκε ως σημαντικός αρωγός στην προσπάθειά μου και με υποστήριξε σε κάθε φάση της πορείας μου. Τις ευχαριστίες μου εκφράζω και στους καθηγητές Δημοσθένη Κυριαζή και Μενύχτα Ανδρέα που δέχτηκαν να είναι μέλη της τριμελής επιτροπής αξιολόγησης της μεταπτυχιακής εργασίας. Τέλος, να ευχαριστήσω τους γονείς μου Χαράλαμπο και Θεοδώρα, που με υπομονή και κουράγιο πρόσφεραν την απαραίτητη ηθική συμπαράσταση για την ολοκλήρωση της μεταπτυχιακής μου εργασίας.

Περίληψη

Τα τελευταία χρόνια η Μηχανική Μάθηση και ιδιαίτερα η Βαθιά Μάθηση, η οποία βασίζεται στα Τεχνητά Νευρωνικά Δίκτυα αναπτύσσονται ραγδαία, παράλληλα με την εξέλιξη των δυνατοτήτων των υπολογιστικών συστημάτων σε υλικό και λογισμικό. Συγχρόνως, στην εποχή των Μεγάλων Δεδομένων, όπου ιδιαίτερο βάρος έχει η στατιστική εκτίμηση και η εξαγωγή συμπερασμάτων, η Βαθιά Μάθηση βρίσκει ιδιαίτερη εφαρμογή γιατί με παρατήρηση ενός μικρού ποσοστού δεδομένων μπορεί να εξαχθεί η απαιτούμενη πληροφορία για πολύ μεγάλα σύνολα δεδομένων.

Τα Τεχνητά Νευρωνικά Δίκτυα, που είναι ο πυρήνας της Βαθιάς Μάθησης, βρίσκουν εφαρμογή στην επίλυση περίπλοκων προβλημάτων σε ένα ευρύ φάσμα εφαρμογών, όπως η ιατρική διάγνωση ασθενειών του καρκίνου. Ο καρκίνος του μαστού αποτελεί την πρώτη αιτία καρκίνου στις γυναίκες. Για να βελτιωθεί το ποσοστό μακροπρόθεσμης επιβίωσης για τους ασθενείς, οι βασικοί παράγοντες είναι η έγκαιρη ανίχνευση και η ακριβής διάγνωση για την ύπαρξη κακοήθειας. Η δημιουργία αξιόπιστων συστημάτων διάγνωσης με τη βοήθεια του υπολογιστή και της Βαθιάς Μάθησης είναι σημαντική βοήθεια για τον ιατρικό κόσμο, ώστε η διάγνωση να είναι ταχύτερη και ευκολότερη, και μάλιστα χωρίς να απαιτείται θεωρητικό και τεχνητό υπόβαθρο για τα Τεχνητά Νευρωνικά Δίκτυα.

Σκοπός της παρούσας διπλωματικής εργασίας είναι η υλοποίηση Πολυεπίπεδων Perceptrons με την χρήση της γλώσσας προγραμματισμού Python και της βιβλιοθήκης Keras για τη διάγνωση καρκίνου του μαστού, με βάση σύνολο δεδομένων του Ουισκόνσιν (Wisconsin Breast Cancer Diagnostic -WBCD) που διατίθεται στο αποθετήριο Μηχανικής Μάθησης UCI .

Λέξεις – Κλειδιά

Μηχανική Μάθηση, Βαθιά Μάθηση, Πολυεπίπεδα Perceptrons, Καρκίνος του Μαστού, WDBC, Ταξινόμηση , Συντελεστής συσχέτισης, Επαναλήψεις, Οπισθόδρομη / Πρόσθια Τροφοδότηση, Keras.

Abstract

In recent years Machine Learning and, especially Deep Learning, which is based on Artificial Neural Networks, are developing rapidly along with the evolution of computing capabilities of computers and software libraries. In the era of Big Data, it is becoming more necessary to extract statistical information and data insights. Deep Learning became a powerful tool because by observing a small percentage of data, the required information can be extracted for huge datasets.

Artificial Neural Networks are the core of Deep Learning and implemented to solve complex problems in a wide range of applications, such as medical diagnosis of cancer. Breast cancer is the leading cause of cancer in women. To improve the long-term survival rate for patients, the key factors are early detection and accurate diagnosis for malignancy. Creating reliable diagnostic systems based on Deep Learning algorithms may be consider of great assistance for medical practioners, so that the diagnosis is faster and easier, even if they do not know anything about artificial neural networks.

The objective of this thesis is to implement Multi-Layer Perceptrons using the Python programming language and the Keras library for breast cancer diagnosis, based on the Wisconsin Breast Cancer Diagnostic Database (WBCD) provided by UCI Machine Learning repository.

Keywords

Machine Learning, Deep Learning, Multi-Layer Perceptrons, Breast Cancer, WDBC, Classification, Correlation coefficient, Epochs, Feedforward / Back Propagation supply, Keras.

Περιεχόμενα

Πίνακας Εικόνων

Πίνακας Πινάκων

Πίνακας Συντομογραφιών & Ακρωνυμίων

AI	Artificial Intelligence
ANN	Artificial Neural Network
API	Application Programming Interface
AUC	Area Under the Curve
CAD	Computer-Aided Detection and Diagnosis
CADe	Computer-Aided Detection Systems
CADx	Computer-Aided Diagnosis Systems
CNN	Convolutional Neural Network
CV	Cross Validation
DL	Deep Learning
ML	Machine Learning
MLP	Multi-Layer Perceptron
ReLU	Rectified Linear Unit
ROC	Receiving Operating Characteristic
RNN	Recurrent Neural network
SGD	Stochastic Gradient Descent
WBCD	Wisconsin Breast Cancer Diagnostic
BM	Βαθιά Μάθηση
ΜΔΕ	Μεταπτυχιακή Διπλωματική Εργασία
ΜΜ	Μηχανική Μάθηση
ΤΝ	Τεχνητή Νοημοσύνη
ΤΝΔ	Τεχνητό Νευρωνικό Δίκτυο

1 Εισαγωγή

1.1 Εισαγωγή

Τα τελευταία χρόνια η Τεχνητή Νοημοσύνη -ΤΝ (Artificial Intelligence – AI) η Μηχανική Μάθηση -ΜΜ (Machine Learning – ML) και ιδιαίτερα η Βαθιά Μάθηση -ΒΜ (Deep Learning – DL), η οποία βασίζεται στα Τεχνητά Νευρωνικά Δίκτυα -ΤΝΔ (Artificial Neural Networks - ANN) αναπτύσσονται ραγδαία, παράλληλα με την εξέλιξη των δυνατοτήτων των υπολογιστικών συστημάτων σε υλικό και λογισμικό. Τα ΤΝΔ είναι εμπνευσμένα από τη λειτουργία και τη διαδικασία μάθησης του ανθρώπινου εγκεφάλου. Παράλληλα, στην εποχή των Μεγάλων Δεδομένων (Big Data), όπου ιδιαίτερο βάρος έχει η στατιστική εκτίμηση και η εξαγωγή συμπερασμάτων, η ΒΜ βρίσκει ιδιαίτερη εφαρμογή γιατί με παρατήρηση ενός μικρού ποσοστού δεδομένων μπορεί να εξαχθεί η απαιτούμενη πληροφορία για πολύ μεγάλα σύνολα δεδομένων. Από το 2016 έχει διαπιστωθεί ότι, ένας αλγόριθμος ΒΜ με επίβλεψη (εποπτευόμενη μάθηση) πετυχαίνει υψηλή απόδοση με διαθέσιμα περίπου 5.000 παραδείγματα ανά κατηγορία και ταυτίζεται ή υπερβαίνει την ανθρώπινη απόδοση όταν εκπαιδεύεται με ένα σύνολο δεδομένων που περιέχει τουλάχιστον 10 εκατομμύρια παραδείγματα [1].

Η ΜΜ βρίσκει ευρεία εφαρμογή σε διάφορους τομείς όπως οι τηλεπικοινωνίες, τα οικονομικά, οι μηχανές αναζήτησης, ο ιατρικός τομέας. Στον ιατρικό τομέα, η ΜΜ μπορεί να διευκολύνει πολύ περίπλοκες και χρονοβόρες εργασίες. Τα ΤΝΔ βρίσκουν εφαρμογή στην επίλυση περίπλοκων προβλημάτων σε ένα ευρύ φάσμα εφαρμογών, όπως η ιατρική διάγνωση ασθενειών του καρκίνου του μαστού [2]. Για να βελτιωθεί το ποσοστό μακροπρόθεσμης επιβίωσης για τους ασθενείς με καρκίνο του μαστού, οι βασικοί παράγοντες είναι η έγκαιρη ανίχνευση και η ακριβής διάγνωση για την ύπαρξη κακοήθειας. Ωστόσο, η αναντιστοιχία μεταξύ των αυξανόμενων ασθενών και της έλλειψης έμπειρων ιατρών δημιουργεί πολλές προκλήσεις για την ακριβή διάγνωση. Η άνιση κατανομή των ιατρικών πόρων σε σχέση με το πλήθος των ασθενών, αυξάνει επίσης την πιθανότητα εσφαλμένης διάγνωσης. Σε συνδυασμό με την αυξανόμενη τάση περιπτώσεων καρκίνου του μαστού, παράγονται επίσης Μεγάλα Δεδομένα (Big Data) τα οποία έχουν σημαντική χρήση στην προώθηση της κλινικής και ιατρικής έρευνας, και πολύ περισσότερο στην εφαρμογή της Επιστήμης των Δεδομένων (Data Science) και της ΜΜ στον προαναφερόμενο τομέα [2]. Ως εκ τούτου, η δημιουργία αξιόπιστων συστημάτων διάγνωσης με τη βοήθεια του

υπολογιστή και της ΜΜ είναι σημαντική βοήθεια για τον ιατρικό κόσμο, ώστε η διάγνωση να είναι ταχύτερη και ευκολότερη, προκειμένου να λαμβάνονται όσο το δυνατόν πιο σωστές αποφάσεις για την υγεία του ασθενούς και την εφαρμογή της κατάλληλης θεραπείας, και μάλιστα χωρίς να απαιτείται θεωρητικό και τεχνητό υπόβαθρο για την ΜΜ και τα ΤΝΔ από τον ιατρό.

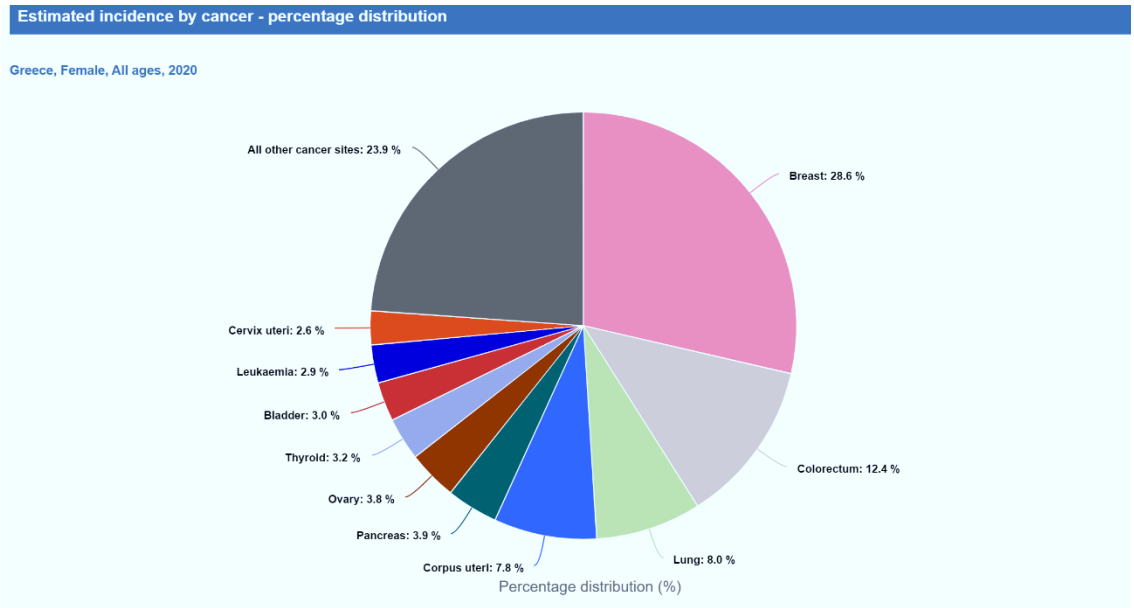
Το θέμα της διάγνωσης του καρκίνου του μαστού με ΜΜ και ιδιαίτερα με ΒΜ και ΤΝΔ , έχει απασχολήσει, ιδιαίτερα τα τελευταία χρόνια την ακαδημαϊκή κοινότητα, όπως προκύπτει από τις μελέτες [3], [4], [5], [6]. Επίσης, σε διάφορες μελέτες διαπιστώνεται ότι, μεταξύ των σύγχρονων τεχνολογιών, τα ΤΝΔ παρέχουν ένα πολύ υψηλό ποσοστό ακρίβειας στη διαδικασία της διάγνωσης για την ταξινόμηση των όγκων [7], [8].

Το σύνολο δεδομένων που χρησιμοποιήθηκε στην παρούσα Μεταπτυχιακή Διπλωματική Εργασία (ΜΔΕ) για την υλοποίηση ενός από τα είδη των ΤΝΔ, του πολυεπίπεδου perceptron (MLP), είναι το σύνολο δεδομένων για τη διάγνωση του καρκίνου του μαστού του Ουισκόνσιν (Wisconsin Breast Cancer Diagnostic -WBCD) [9], ο οποίο διατίθεται υπάρχει στο αποθετήριο μηχανικής μάθησης UCI [10]. Τα δεδομένα συλλέχθηκαν από τους Wolberg, Street and Mangasarian από τα Νοσοκομεία του Πανεπιστημίου του Ουισκόνσιν [11] και περιλαμβάνουν αριθμητικά αποτελέσματα από εικόνες βιοψίας όγκων του μαστού.

1.2 Ορισμός του προβλήματος

Ο καρκίνος είναι μια μεγάλη ομάδα ασθενειών που μπορούν να ξεκινήσουν σχεδόν σε οποιοδήποτε όργανο ή ιστό του σώματος όταν ανώμαλα κύτταρα αναπτύσσονται ανεξέλεγκτα, υπερβαίνουν τα συνηθισμένα όριά τους για να εισβάλουν σε παρακείμενα μέρη του σώματος ή/και να εξαπλωθούν σε άλλα όργανα. Η τελευταία διαδικασία ονομάζεται μετάσταση και είναι μια σημαντική αιτία θανάτου από καρκίνο. Άλλες κοινές ονομασίες για τον καρκίνο είναι το νεόπλασμα και ο κακοήθης όγκος [12]. Σύμφωνα με τον Παγκόσμιο Οργανισμό Υγείας (World Health Organization - WHO), καρκίνος είναι η δεύτερη κύρια αιτία θανάτου παγκοσμίως, αντιπροσωπεύοντας περίπου 9,6 εκατομμύρια θανάτους, ή έναν στους έξι θανάτους, το 2018 [12]. Ο καρκίνος του πνεύμονα, του προστάτη, του παχέος εντέρου, του στομάχου και του ήπατος είναι οι πιο συνηθισμένοι τύποι καρκίνου στους άνδρες, ενώ ο καρκίνος του μαστού, παχέος εντέρου, του πνεύμονα, του τραχήλου της μήτρας και του θυρεοειδούς είναι οι πιο συνηθισμένοι τύποι στις γυναίκες.

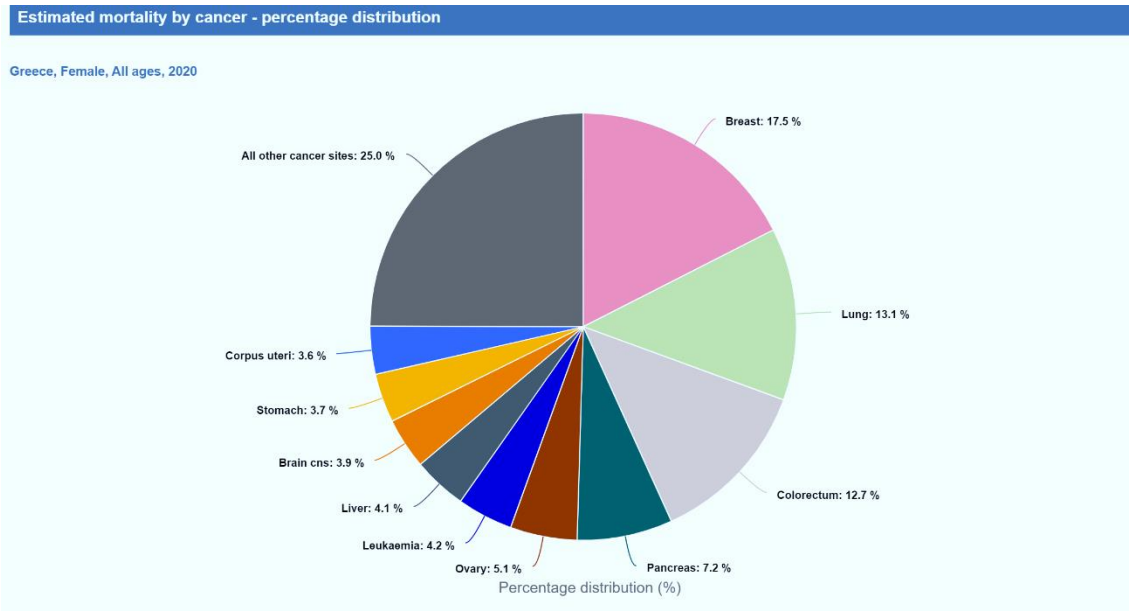
Σύμφωνα με τον Διεθνή Οργανισμό Έρευνας για τον Καρκίνο (International Agency for Research on Cancer – IACR) ο εκτιμώμενος αριθμός νέων περιπτώσεων καρκίνου του μαστού για το 2018 ανέρχεται στα 2.088.849 επί συνόλου 8.622.539 περιπτώσεων, καταλαμβάνοντας το ποσοστό του 24,2% επί του συνόλου των περιπτώσεων του γυναικείου πληθυσμού και το 11,6% επί του συνόλου του πληθυσμού, καθώς και το 15% των θανάτων γυναικών από καρκίνο [13].
Όσον αφορά την Ελλάδα, εκτιμάται για το 2020 από το Ευρωπαϊκό Σύστημα Πληροφοριών για τον Καρκίνο (European Cancer Information Systems – ECIS) ότι επί συνόλου 27.157 περιπτώσεων καρκίνου του γυναικείου πληθυσμού, 7.772 περιπτώσεις αφορούν καρκίνο του μαστού, καταλαμβάνοντας το ποσοστό του 28,6%, όπως φαίνεται στην Εικόνα 1-1.



Εικόνα 1-1. Ποσοστιαία εκτίμηση περιπτώσεων καρκίνου του μαστού στην Ελλάδα για το 2020.
Πηγή: European Cancer Information System, "ECIS - European Cancer Information System Measuring cancer burden and its time trends across Europe." [Online]. Available: https://ecis.jrc.ec.europa.eu/index.php [Accessed: 31-Aug-2020]

Επίσης, ο αριθμός των θανάτων από τον καρκίνο του μαστού εκτιμάται σε 2.333 καταλαμβάνοντας το ποσοστό του 17,5%, όπως φαίνεται στην Εικόνα 1-2.

Σύμφωνα με τις παραπάνω πηγές, σημαντικό ρόλο στην αντιμετώπιση της ασθένειας κατέχει η πρόληψη και η σωστή διάγνωση σε περίπτωση εμφάνισης όγκου. Το θέμα της σωστής διάγνωσης είναι σημαντικό για περαιτέρω αποφάσεις που αφορούν τη θεραπεία και κατ' επέκταση τη μείωση της θνησιμότητας.



Εικόνα 1-2. Ποσοστιαία εκτίμηση θανάτων από καρκίνο του μαστού στην Ελλάδα για το 2020.
Πηγή: European Cancer Information System, "ECIS - European Cancer Information System Measuring cancer burden and its time trends across Europe." [Online]. Available: https://ecis.jrc.ec.europa.eu/index.php [Accessed: 31-Aug-2020]

1.3 Δομή Μεταπτυχιακής Διπλωματικής Εργασίας

Το Κεφάλαιο 1 της εργασίας αποτελεί την εισαγωγή στο θέμα ΜΔΕ. Παρουσιάζεται σύντομα το πρόβλημα του καρκίνου του μαστού, η δομή της ΜΔΕ, καθώς και η συνεισφορά της.

Στο Κεφάλαιο 2 παρουσιάζεται συνοπτικά η ασθένεια του καρκίνου του μαστού. Περιγράφεται η ασθένεια, οι παράγοντες κινδύνου και τα συμπτώματα, καθώς και οι μέθοδοι διάγνωσης και βιοψίας.

Στα Κεφάλαια 3 και 4 παρουσιάζεται το θεωρητικό υπόβαθρο που απαιτείται για την υλοποίηση του ΤΝΔ. Έστω και εάν στις μέρες μας οι περισσότερες διαδικασίες για την υλοποίηση μιας εφαρμογής ΤΝΔ παρέχονται μέσω βιβλιοθηκών λογισμικού γλωσσών προγραμματισμού, η κατανόηση των βασικών όρων και εννοιών είναι απαραίτητη για τη σωστή χρήση τους.

Στο Κεφάλαιο 3 αφορά τις βασικές έννοιες και όρους για τα Τεχνητά Νευρωνικά Δίκτυα (ΤΝΔ). Δίνονται οι ορισμοί της Τεχνητής Νοημοσύνης (ΤΝ), της Μηχανικής Μάθησης (ΜΜ) και της

Βαθιάς Μάθησης (ΒΜ), μια σύντομη ιστορική αναδρομή για τα ΤΝΔ και οι σύγχρονες τάσεις. Τέλος, παρουσιάζεται η εργασία της ταξινόμησης, εφόσον το πρόβλημα της διάγνωσης του καρκίνου του μαστού ανήκει σε αυτή την κατηγορία εργασιών.

Στο Κεφάλαιο 4 παρουσιάζονται τα βασικά στοιχεία των ΤΝΔ. Αρχικά, παρουσιάζονται ο βιολογικός νευρώνας και ο τεχνητός νευρώνας, καθώς και τα είδη των ΤΝΔ. Στη συνέχεια, αναλύεται αρχιτεκτονική των MPLs και η διαδικασία της εκπαίδευσής τους. Κατόπιν παρουσιάζονται οι κυριότερες συναρτήσεις ενεργοποίησης, οι συνηθισμένες συναρτήσεις κόστους, οι αλγόριθμοι βελτιστοποίησης και οι συχνότερες μέθοδοι εξομάλυνσης. Τέλος, ανακεφαλαιώνεται συνοπτικά η διαδικασία εκπαίδευσης.

Στο Κεφάλαιο 5 παρουσιάζεται το περιβάλλον υλοποίησης σε Python, όπου αναφέρονται οι κυριότερες βιβλιοθήκες, με έμφαση στην Keras και η μεθοδολογία για την ανάπτυξη του έργου. Στη συνέχεια, γίνεται μια παρουσίαση του συνόλου δεδομένων WBCD, προκειμένου να έχουμε μια εικόνα για τα δεδομένα και, τέλος, γίνεται μια ανασκόπηση της βιβλιογραφίας, όπου παρουσιάζονται τρεις μελέτες που αφορούν την υλοποίηση MLPs για το ίδιο σύνολο δεδομένων με Python και τις ίδιες σχετικές βιβλιοθήκες.

Στο Κεφάλαιο 6 παρουσιάζονται οι υλοποιήσεις διάφορων μοντέλων MLP με τη γλώσσα προγραμματισμού Python και τις συναφείς βιβλιοθήκες. Αρχικά, παρουσιάζεται το στάδιο της εισόδου και της προετοιμασίας των δεδομένων σε μορφή κατάλληλη για τις επόμενες εργασίες που αφορούν την ανάπτυξη ενός MLP μοντέλου. Στη συνέχεια, παρουσιάζεται η γενική ροή των εργασιών για την ανάπτυξη, καθώς και οι αρχικές επιλογές για τις υπερπαραμέτρους της βελτιστοποίησης και της ομαλοποίησης. Σύμφωνα με τις αυτές τις επιλογές, παρουσιάζονται πρώτα για κάθε μοντέλο MLP τα αποτελέσματα της υλοποίησης με πειραματισμό όσον αφορά την επιλογή υπερπαραμέτρων των μοντέλων MLPs. Τέλος, παρουσιάζονται τα αποτελέσματα για διάφορα μοντέλα με αυτόματη ρύθμιση υπερπαραμέτρων και παρατίθεται η σύγκριση τους.

Τέλος, στο Κεφάλαιο 7 παρουσιάζονται τα συμπεράσματα που εξήχθησαν από την παρούσα ΜΔΕ, καθώς και οι μελλοντικές κατευθύνσεις.

1.4 Συνεισφορά Μεταπτυχιακής Διπλωματικής Εργασίας

Ο καρκίνος του μαστού είναι μια εξαιρετικά σοβαρή ασθένεια και είναι ο πιο κοινός καρκίνος που πλήττει τον γυναικείο πληθυσμό. Αντιπροσωπεύοντας το 28,6% όλων των νέων περιπτώσεων καρκίνου του μαστού στην Ελλάδα, είναι ένα θέμα έρευνας με μεγάλη αξία.

Σε περίπτωση ανίχνευσης όγκου στον μαστό, η διάγνωση της κακοήθειας ή μη του όγκου μέσω της βιοψίας συμβάλλει στον μέγιστο βαθμό για την λήψη κλινικών αποφάσεων.

Η παρούσα ΜΔΕ συνεισφέρει στον τομέα της ιατρικής, όσον αφορά την αποτελεσματικότερη διάγνωση και ταξινόμηση του καρκίνου του μαστού ενός ασθενούς. Στις μέρες μας, ιδιαίτερη θέση στις μεθόδους ΜΜ κατέχουν τα ΤΝΔ. Η γλώσσα προγραμματισμού που κατέχει την κυρίαρχη θέση στην επιστήμη των δεδομένων και της ΒΜ είναι η Python.

Συνεπώς, η υλοποίηση ενός ΤΝΔ με υψηλό μέτρο απόδοσης με τη γλώσσα προγραμματισμού Python για τη διάγνωση της ασθένειας αποκτά ιδιαίτερη αξία.

2 Η Ασθένεια του Καρκίνου του Μαστού

Στο κεφάλαιο αυτό παρουσιάζεται συνοπτικά η ασθένεια του καρκίνου του μαστού. Περιγράφεται η ασθένεια, οι παράγοντες κινδύνου και τα συμπτώματα, καθώς και οι μέθοδοι διάγνωσης και βιοψίας.

2.1 Περιγραφή της Ασθένειας

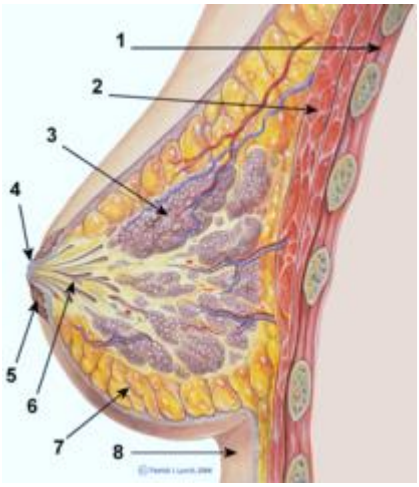
Ο καρκίνος του μαστού είναι μια ομάδα ασθενειών στις οποίες τα κύτταρα στον ιστό του μαστού αλλάζουν και διαιρούνται ανεξέλεγκτα, με αποτέλεσμα συνήθως ένα κομμάτι ή μάζα. Οι περισσότεροι καρκίνοι του μαστού ξεκινούν στους λοβούς (αδένες του γάλακτος) ή στους αγωγούς που συνδέουν τους λοβούς με τη θηλή.

2.1.1 Περιγραφή του μαστού

Οι μαστοί βρίσκονται στο πρόσθιο θωρακικό τοίχωμα και αναπτύσσονται ιδιαίτερα στις γυναίκες κατά την εφηβεία, όταν αρχίζουν να παράγονται οι γυναικείες ορμόνες, δηλαδή τα οιστρογόνα και η προγεστερόνη από τις ωοθήκες. Η εξωτερική μορφολογία του μαστού περιλαμβάνει τη θηλή, τη θηλαία άλω και τα αλωαία οζίδια. Η θηλή αποτελεί καστανέρυθρο έπαρμα του δέρματος του μαστού που βρίσκεται λίγο πιο κάτω και έξω από το μέσο του μαστού. Στην κορυφή της υπάρχουν 15-20 στόμια (λοβοί) όπου καταλήγουν οι γαλακτοφόροι πόροι. Η θηλαία άλως είναι υποστρόγγυλη και ελαφρά υψωμένη περιοχή γύρω από την θηλή. Εσωτερικά ο μαστός αποτελείται από τον μαστικό αδένα και από λίπος. Ο μαστικός αδένας βρίσκεται πίσω από την θηλαία άλω και αποτελείται από τους λοβούς όπου παράγεται το γάλα και τους γαλακτοφόρους πόρους που μεταφέρουν το γάλα στους γαλακτοφόρους κόλπους και από εκεί στη θηλή. [15]. Στην Εικόνα 2-1 φαίνεται η διατομή του μαστού και τα κυριότερα στοιχεία του.

2.1.2 Βασικοί τύποι της ασθένειας

Οι βασικοί τύποι της ασθένειας του καρκίνου του μαστού είναι ο πορογενής καρκίνος, ο οποίος προέρχεται από τους γαλακτοφόρους πόρους και ο λοβιακός καρκίνος που προέρχεται από τους λοβούς. Ο καρκίνος μπορεί να είναι διηθητικός ή μη διηθητικός (in situ). Στην περίπτωση του διηθητικού καρκίνου υπάρχουν μεταστάσεις, ενώ στον μη διηθητικό τύπο καρκίνου δυνητικά δεν υπάρχουν μεταστάσεις και θεωρείται αρχόμενος (Προ καρκινικό στάδιο) [16].



1. Θωρακικό τοίχωμα
2. Θωρακικοί μύες
3. Λοβοί
4. Θηλή
5. Θηλαία άλως
6. Γαλακτοφόρος πόρος
7. Λιπώδης ιστός
8. Δέρμα

Εικόνα 2-1. Διατομή του μαστού.
Πηγή: Wikipedia, "Breast." [Online]. Available: https://en.wikipedia.org/w/index.php?title=Breast&oldid=968115085 [Accessed: 31-Aug-2020]

2.1.3 Παράγοντες κινδύνου

Οι κυριότεροι παράγοντες κινδύνου οι οποίοι έχουν εντοπισθεί και προκαλούν τον καρκίνο του μαστού, είναι [17]:

- Η ηλικία: Περίπου το 80% των περιπτώσεων καρκίνου του μαστού συμβαίνουν σε γυναίκες άνω των 50 ετών.
- Το ατομικό ιστορικό: Το ιστορικό καρκίνου στον ένα μαστό συνεπάγεται αυξημένο κίνδυνο και στον άλλο μαστό.
- Οικογενειακό ιστορικό: Η ύπαρξη σε συγγενείς πρώτου βαθμού καρκίνου του μαστού ή καρκίνου του τραχήλου της μήτρας.
- Η γενετική προδιάθεση: Το 5-10% των περιπτώσεων είναι κληρονομικοί.
- Η έκθεση σε ακτινοβολία.
- Τα υψηλό ανάστημα, η παχυσαρκία μετα-εμμηνοπαυσιακών γυναικών και η μεγάλη μάζα του αδένα του μαστού.
- Η πρώιμη αρχή έμμηνης ρύσης και η όψιμη εμμηνόπαυση.
- Η ηλικία γέννησης του πρώτου παιδιού: Σε μεγαλύτερες ηλικίες, η τεκνοποίηση συνεπάγεται μεγαλύτερο κίνδυνο.
- Η θεραπευτική λήψη ορμονών, οιστρογόνων και προγεστερόνης για διάστημα μεγαλύτερο των τεσσάρων ετών.
- Η ινοκυστική μαστοπάθεια.
- Αντισυλληπτικά δισκία, αν και η αύξηση του κινδύνου είναι πολύ μικρή και έχει βραχεία διάρκεια.

- Η αυξημένη κατανάλωση οινοπνευματωδών ποτών, το κάπνισμα, καθώς και αυξημένη ύπαρξη λίπους, ιδιαίτερα στο άνω μέρος του σώματος.

2.1.4 Συμπτώματα

Το πιο σύνηθες προειδοποιητικό σύμπτωμα του καρκίνου του μαστού είναι ένας συμπαγής, συνήθως σταθερός όγκος. Μπορεί να υπάρχει πόνος, μπορεί και να μην υπάρχει. Άλλα σημεία μπορεί να είναι [17]:

- Έκκριση ορώδους ή αιμορραγικού υγρού από τη θηλή.
- Η θηλή μπορεί να είναι στραμμένη προς τα μέσα (εισολκή) ή να εκκρίνει σκουρόχρωμο υγρό.
- Το δέρμα πάνω από τον όγκο μπορεί έχει την όψη της «φλούδας πορτοκαλιού» ή να παρουσιάζει κοιλότητες στις περιοχές όπου έχει εξαπλωθεί ο καρκίνος.
- Αλλαγή στο σχήμα ή στην εξωτερική καμπύλη του μαστού, καθώς και ασσυμετρία μεταξύ των δύο μαστών.

Αξίζει να σημειωθεί πως, η αλλαγή στο μέγεθος ή στην υφή του μαστού μπορεί να οφείλεται σε πληθώρα άλλων καταστάσεων, όπως οι αλλαγές στον ιστό του μαστού που συμβαίνουν φυσιολογικά κατά τη διάρκεια της εγκυμοσύνης, του καταμήνιου κύκλου και της εμμηνόπαυσης, οι κύστεις, τα αδενώματα κ.λπ. [17]. Οι μη φυσιολογικές αλλαγές θα πρέπει σε κάθε περίπτωση να διερευνώνται.

2.2 Διάγνωση και Μέθοδοι Βιοψίας

Η διάγνωση του καρκίνου του μαστού σε προ καρκινικό στάδιο (in situ) αυξάνεται όλο και περισσότερο λόγω της ευαισθητοποίησης των γυναικών όσον αφορά τον προληπτικό έλεγχο. Ο έλεγχος με απεικονιστικές μεθόδους γίνεται με υπερηχογράφημα, μαστογραφία, τρισδιάστατη μαστογραφία και μαγνητική τομογραφία [18]. Επίσης, μπορεί να γίνει έλεγχος με κλινική ψηλάφηση από ιατρό καθώς και με την αυτό ψηλάφηση.

Όταν μετά τη διενέργεια των ελέγχων υπάρχουν ενδείξεις για ύπαρξη καρκίνου του μαστού, χρειάζεται να διενεργηθεί βιοψία. Η διενέργεια της βιοψίας, δεν συνεπάγεται απαραίτητα την ύπαρξη καρκίνου. Οι πλειοψηφία των βιοψιών έχουν ως αποτέλεσμα τη μη ύπαρξη καρκίνου, αλλά αυτός είναι ο μόνος σίγουρος τρόπος διάγνωσης. Κατά τη διάρκεια της βιοψίας ο ιατρός

αφαιρεί μικρά τμήματα από την ύποπτη περιοχή, έτσι ώστε στη συνέχεια να γίνει η εξέτασή τους σε εργαστήριο προκειμένου να εντοπισθεί η ύπαρξη καρκινικών κυττάρων.
Οι τρόποι βιοψίας είναι οι εξής [19]:

- **Αναρρόφηση με λεπτή βελόνα (FNA - Fine Needle Aspiration):** Χρησιμοποιείται μια πολύ λεπτή, κοίλη βελόνα που προσαρμόζεται σε σύριγγα για την αναρρόφησης μιας μικρής ποσότητας ιστού από την ύποπτη περιοχή.
- **Βασική βιοψία με βελόνα (Core needle biopsy)**: Χρησιμοποιείται μια μεγαλύτερη βελόνα για τη δειγματοληψία των αλλαγών του μαστού που έγιναν αισθητές από τον γιατρό ή παρατηρήθηκε σε υπερηχογράφημα, μαστογραφία ή μαγνητική τομογραφία. Αυτός ο τρόπος και ο προηγούμενος είναι οι προτιμώμενοι τύποι βιοψίας εάν υπάρχει υποψία για καρκίνο του μαστού.
- **Χειρουργική (ανοιχτή) βιοψία (Surgical Breast Biopsy)**: Σε σπάνιες περιπτώσεις, όταν για παράδειγμα δεν μπορεί ο ιατρός να εξάγει ασφαλή συμπεράσματα μέσω των μεθόδων με βελόνα, απαιτείται χειρουργική επέμβαση για την αφαίρεση όλου ή μέρους του όγκου για έλεγχο. Τις περισσότερες φορές, ο χειρουργός αφαιρεί ολόκληρη τη μάζα ή την ανώμαλη περιοχή, καθώς και ένα τμήμα του φυσιολογικού ιστού του μαστού.
- **Βιοψία λεμφαδένων (Lymph node biopsy):** Σε κάποιες περιπτώσεις μπορεί να χρειαστεί βιοψία των λεμφαδένων της μασχάλης, για να εξεταστεί εάν υπάρχει μετάσταση. Αυτή η βιοψία μπορεί να γίνει παράλληλα με τη βιοψία του μαστού ή κατά την χειρουργική αφαίρεση του όγκου.

2.2.1 Διάγνωση Μέσω Υπολογιστικών Συστημάτων

Η ανίχνευση και διάγνωση με τη βοήθεια υπολογιστή (Computer-Aided Detection and Diagnosis - CAD) είναι η κατηγορία συστημάτων υπολογιστών που χρησιμοποιούνται για να βοηθήσουν τους επαγγελματίες για την ανίχνευση ή/και τη διάγνωση ασθενειών. Ο σκοπός ενός συστήματος CAD για τις περιπτώσεις της ασθένειας του καρκίνου, είναι να βοηθήσει τους ιατρούς να ερμηνεύσουν σωστά τις ιατρικές εικόνες σε μικρότερο χρονικό διάστημα και με μεγαλύτερη βεβαιότητα. Τα συστήματα CAD ταξινομούνται σε δύο ομάδες [20]:

- **Συστήματα ανίχνευσης με τη βοήθεια υπολογιστή (CADe):** Τα συστήματα CADe ασχολούνται γενικά με τη θέση των αλλοιώσεων σε ιατρικές εικόνες.

- **Συστήματα διάγνωσης με τη βοήθεια υπολογιστή (CADx):** Τα συστήματα CADx ασχολούνται με την ταξινόμηση των αλλοιώσεων, για παράδειγμα, τη διάκριση μεταξύ καλοήθους ή κακοήθους όγκου.

Η διάγνωση μέσω υπολογιστή αποτελεί ένα σημαντικό τομέα της Τεχνητής Νοημοσύνης. Η ερμηνεία ιατρικών εικόνων είναι συχνά χρονοβόρα και απαιτεί σημαντική ανθρώπινη εμπειρία. Ως εκ τούτου, υπάρχει μια αυξανόμενη τάση για τη χρήση τεχνικών Μηχανικής Μάθησης, ιδιαίτερα των ΤΝΔ για την σωστή ταξινόμηση διαφορετικών ιατρικών εικόνων ή δεδομένων που έχουν εξαχθεί από ιατρικές εικόνες, που έχουν ληφθεί από διάφορες διαγνωστικές μεθόδους και μεθόδους βιοψίας.

3 Μηχανική Μάθηση και Βαθιά Μάθηση

Το κεφάλαιο αφορά τις βασικές έννοιες και όρους για τα Τεχνητά Νευρωνικά Δίκτυα (ΤΝΔ). Δίνονται οι ορισμοί της Τεχνητής Νοημοσύνης (ΤΝ), της Μηχανικής Μάθησης (ΜΜ) και της Βαθιάς Μάθησης (ΒΜ), μια σύντομη ιστορική αναδρομή για τα ΤΝΔ και οι σύγχρονες τάσεις. Τα MLP είναι υποτομέας της ΜΜ, συνεπώς η παρουσίαση βασικών εννοιών της ΜΜ κρίνεται απαραίτητη ως θεωρητικό υπόβαθρο. Τέλος, παρουσιάζεται η εργασία της ταξινόμησης, εφόσον το πρόβλημα της διάγνωσης του καρκίνου του μαστού ανήκει σε αυτή την κατηγορία εργασιών.

3.1 Τεχνητή Νοημοσύνη - Μηχανική Μάθηση - Βαθιά Μάθηση

Αρχικά, πρέπει να καθοριστεί με σαφήνεια σε τι αναφέρεται η Τεχνητή Νοημοσύνη (ΤΝ). Τι είναι η ΤΝ, η Μηχανική Μάθηση (ΜΜ) και η Βαθιά Μάθηση (ΒΜ) πως σχετίζονται μεταξύ τους και γιατί η ΒΜ είναι πλέον ο κυρίαρχος τομέας της ΤΝ.

Η **Τεχνητή Νοημοσύνη (Artificial Intelligence)** γεννήθηκε στην δεκαετία του 1950, όταν πρωτοπόροι στο νεοεμφανιζόμενο πεδίο της επιστήμης των υπολογιστών αναρωτήθηκαν εάν οι υπολογιστές μπορούν να σκεφθούν.

Ένας συνοπτικός ορισμός του πεδίου δόθηκε το 1990 από τους Rich & Knight [21]:

«Τεχνητή Νοημοσύνη είναι η μελέτη του πώς να κάνουμε τους υπολογιστές ικανούς να κάνουν πράγματα στα οποία προς το παρόν οι άνθρωποι τα καταφέρνουν καλύτερα»

Ουσιαστικά, η ΤΝ είναι η προσπάθεια αυτοματοποίησης των πνευματικών εργασιών που συνήθως εκτελούνται από ανθρώπους. Σε αυτή τη βάση, η ΤΝ είναι ένα γενικό πεδίο που περιλαμβάνει την ΜΜ και την ΒΜ, αλλά επίσης περιλαμβάνει και πολλές ακόμη προσεγγίσεις, οι οποίες δεν περιλαμβάνουν μάθηση. Για παράδειγμα, τα πρώτα προγράμματα για το σκάκι, περιλάμβαναν τους κανόνες σε προγραμματιστικό κώδικα χωρίς να εμπλέκεται η μηχανική μάθηση. Για ένα μεγάλο χρονικό διάστημα, οι ειδικοί πίστευαν πως, η ΤΝ σε ανθρώπινο επίπεδο θα μπορούσε να επιτευχθεί με τη συγγραφή προγραμμάτων, όπου θα περιλαμβανόταν ένα σύνολο ρητών κανόνων για τη διαχείριση της γνώσης. Αυτή η προσέγγιση είναι γνωστή ως **συμβολική ΤΝ (symbolic AI)** ή **ΤΝ βασισμένη στη γνώση (knowledge base AI)** και ήταν το κυρίαρχο παράδειγμα από τη

δεκαετία του 1950 έως τα τέλη της δεκαετίας του 1980. Αν και η συμβολική ΤΝ αποδείχθηκε ότι είναι κατάλληλη για την επίλυση σαφώς καθορισμένων λογικών προβλημάτων, όπως το παιχνίδι του σκάκι, όπου οι κανόνες είναι συγκεκριμένοι και μπορούν εύκολα να διατυπωθούν, απέτυχε στην επίλυση πιο περίπλοκων προβλημάτων του πραγματικού κόσμου, όπως είναι η ιατρική διάγνωση, η αναγνώριση εικόνων και η μετάφραση ομιλίας.

Ετσι, προέκυψε μια νέα προσέγγιση ΤΝ: η **Μηχανική Μάθηση** (**Machine Learning**). Ουσιαστικά, η ΜΜ προέκυψε από το ερώτημα εάν ένας υπολογιστής θα μπορούσε να προγραμματιστεί έτσι ώστε να εκτελεί μια συγκεκριμένη εργασία μαθαίνοντας τους κανόνες εξετάζοντας μόνο τα δεδομένα. Ο σχετικός γενικός ορισμός δόθηκε το 1959 από τον Arthur Samuel [22]:

«Η Μηχανική μάθηση είναι το πεδίο μελέτης το οποίο δίνει στους υπολογιστές τη δυνατότητα να μαθαίνουν χωρίς να έχουν ρητά προγραμματιστεί»

Στη συμβολική ΤΝ εισάγονται οι κανόνες από τους ανθρώπους (μέσω ενός προγράμματος) και τα δεδομένα που πρέπει να επεξεργαστούν αυτοί οι κανόνες, ώστε να εξαχθούν οι απαντήσεις. Στην ΜΜ, οι άνθρωποι εισάγουν τα δεδομένα και τις απαντήσεις που αναμένονται από αυτά τα δεδομένα και εξάγονται οι κανόνες [23], όπως φαίνεται στην Εικόνα 3-1, δημιουργώντας έτσι νέα μοντέλα (models) προγραμματισμού.

Εικόνα 3-1. Κλασσικός προγραμματισμός και μηχανική μάθηση

Αυτά τα μοντέλα, μπορούν στη συνέχεια να εφαρμοστούν σε νέα δεδομένα για να παράγουν αποτελέσματα. Στην πραγματικότητα, η βασική ιδέα πίσω από τη ΜΜ είναι ότι, είναι δυνατή η δημιουργία αλγορίθμων που μαθαίνουν και κάνουν προβλέψεις σχετικά με τα δεδομένα. Ένας πιο περιεκτικός ορισμός με βάση αυτή την προσέγγιση, και ο οποίος ουσιαστικά ορίζει τι είναι μάθηση, δόθηκε το 1997 από τον Tom Mitchell [1]:

*«Ένα πρόγραμμα υπολογιστή λέγεται ότι μαθαίνει από την εμπειρία **E** (Experience) σε σχέση με κάποια εργασία **T** (Task) και κάποιο μέτρο απόδοσης **P** (Performance), εάν βελτιωθεί η απόδοσή του στο **T**, όπως μετράτε από το **P**, με εμπειρία **E**».*

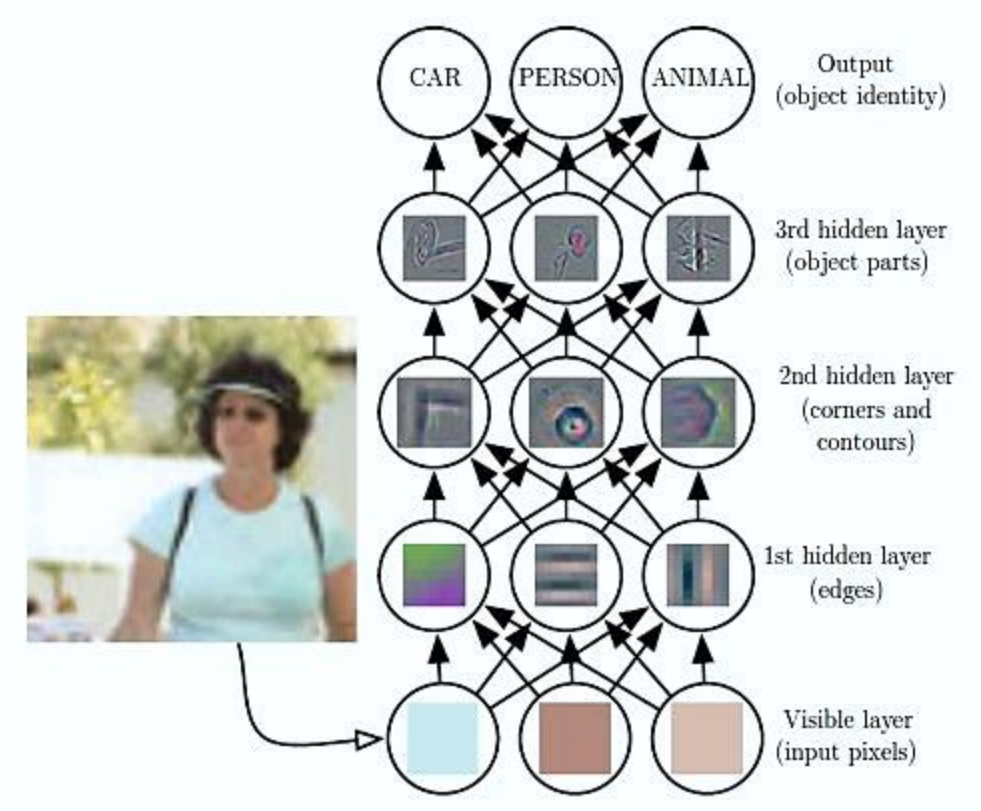
Για παράδειγμα, ας δούμε το πρόβλημα της διάγνωσης μιας ασθένειας, όπου **στόχος (target)** είναι η απάντηση εάν ένας άνθρωπος έχει μια ασθένεια ή όχι. Για μια τέτοια εργασία **T**, οι άνθρωποι – ειδικοί στον τομέα συλλέγουν πολλές περιπτώσεις, δηλαδή **παραδείγματα (examples)** όπου ένας ασθενής είχε ή δεν είχε την εν λόγω ασθένεια έχοντας καταγράψει μια σειρά **χαρακτηριστικών (features)** που θα βοηθούσαν στην πρόβλεψη, όπως, την ηλικία του ασθενούς, το φύλο και τα αποτελέσματα από μια σειρά διαγνωστικών εξετάσεων όπως η αρτηριακή πίεση, το σάκχαρο του αίματος και ούτω καθεξής και θέτοντας για κάθε παράδειγμα μια **ετικέτα (label)** όπου δηλώνεται το εάν υπάρχει η ασθένεια ή όχι. Για τα δεδομένα αυτά, μπορεί να γίνει η κατάλληλη **αναπαράσταση (representation)**, έτσι ώστε να είναι σε μορφή κατάλληλη για τον υπολογιστή. Το σύνολο των παραδειγμάτων σε αυτή τη μορφή αποτελούν ένα **σύνολο δεδομένων (dataset)** ή αλλιώς **σύνολο εκπαίδευσης (training set)** μέσω του οποίου αποκτάται η εμπειρία **E**. Ένας αλγόριθμος ΜΜ θα μπορούσε να προσδιορίσει πώς να συμπεράνει εάν ο ασθενής έχει την ασθένεια ή όχι με τη γενίκευση από τα δεδομένα. Έτσι, στην ουσία ο υποκείμενος αλγόριθμος ΜΜ βρίσκει μια μαθηματική συνάρτηση που μπορεί να παράγει με βάση τα χαρακτηριστικά το σωστό αποτέλεσμα για τον στόχο (ασθένεια ή όχι) λαμβάνοντας υπόψη τα παραδείγματα. Η εύρεση της απλούστερης μαθηματικής συνάρτησης που προβλέπει τα αποτελέσματα με το απαιτούμενο επίπεδο ακρίβειας είναι η καρδιά της ΜΜ. Το απαιτούμενο επίπεδο ορθότητας του αλγόριθμου εξαρτάται από την εκάστοτε εργασία και είναι το μέτρο απόδοσης **P** του αλγόριθμου. Στο συγκεκριμένο παράδειγμα, μέτρο απόδοσης θα μπορούσε να οριστεί η ορθότητα (accuracy) ως το ποσοστό των σωστών προβλέψεων για ύπαρξη ασθένειας σε σχέση με το σύνολο των παραδειγμάτων που αντιστοιχούν σε ύπαρξη ασθένειας.

Κατά μία έννοια, η πραγματική ευφυΐα είναι στον προσδιορισμό των χαρακτηριστικών και στην αναπαράσταση των δεδομένων και αυτό που κάνει ο αλγόριθμος ΜΜ είναι απλά να μάθει πώς να συνδυάζει αυτά τα χαρακτηριστικά για να φτάσει στη σωστή απάντηση.

Για πολλές όμως περιπτώσεις, όπως η αναγνώριση μιας εικόνας, ο προσδιορισμός των χαρακτηριστικών είναι δύσκολος και συνεπώς είναι δύσκολη και η αναπαράσταση των δεδομένων. Εδώ έρχεται να δώσει την απάντηση η **Βαθιά Μάθηση (Deep Learning)** [1]**.**

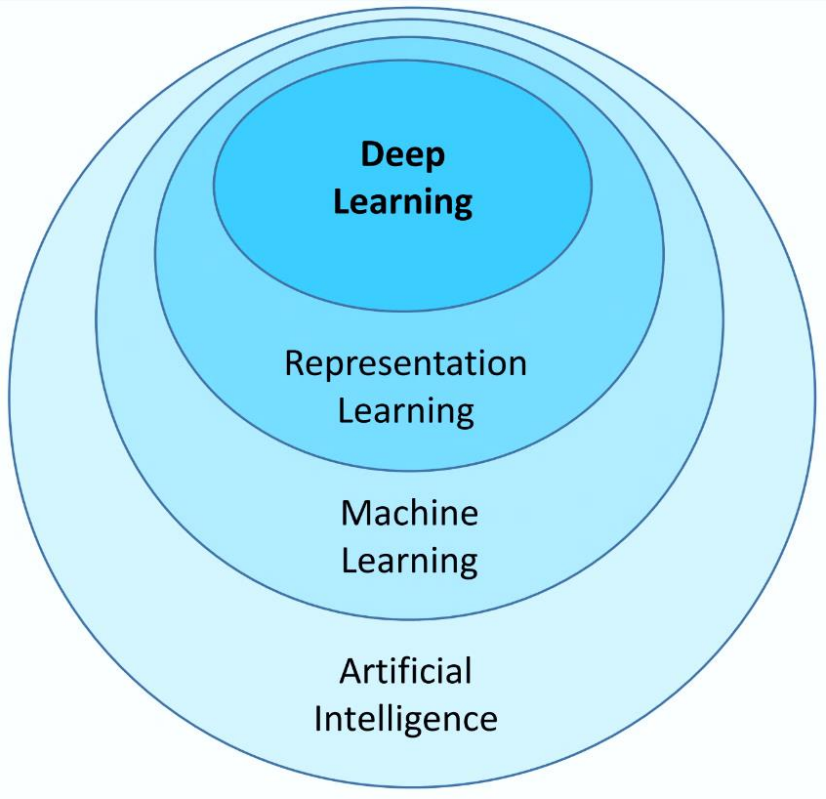
Τα ανθρώπινα όντα μαθαίνουν την κατάλληλη αναπαράσταση των δεδομένων από τα ίδια τα δεδομένα. Επιπλέον, οργανώνουν έννοιες ως ιεραρχία όπου οι περίπλοκες έννοιες εκφράζονται χρησιμοποιώντας πρωτόγονες έννοιες. Το πεδίο της ΒΜ επικεντρώνεται στην εκμάθηση κατάλληλων αναπαραστάσεων δεδομένων έτσι ώστε αυτά να μπορούν να χρησιμοποιηθούν για την εξαγωγή συμπερασμάτων. Η λέξη «βαθιά» στη βαθιά μάθηση αναφέρεται στην ιδέα της εκμάθησης της ιεραρχίας των εννοιών απευθείας από δεδομένα. Η ΒΜ είναι **«μάθηση με αναπαράσταση» (representation learning)** εισάγοντας αναπαραστάσεις που εκφράζονται με όρους άλλων απλούστερων αναπαραστάσεων, όπως φαίνεται ενδεικτικά στην Εικόνα 3-2, όπου παρουσιάζεται ο τρόπος με τον οποίο τα δεδομένα μιας εικόνας ταξινομούνται σε απλούστερα τμήματα ιεραρχικά [1], [23].

Η ΒΜ είναι μια κατηγορία αλγορίθμων ΜΜ εμπνευσμένη από τη δομή ενός ανθρώπινου εγκεφάλου που δίνει τη δυνατότητα στους υπολογιστές να βελτιώνονται με την εμπειρία και τα δεδομένα.



Εικόνα 3-2. Απεικόνιση ενός μοντέλου βαθιάς μάθησης.
Πηγή: I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016

Στην Εικόνα 3-3 φαίνεται περιγραφικά πως ο κάθε τομέας αποτελεί υποσύνολο της ΤΝ.



Εικόνα 3-3. Τεχνητή Νοημοσύνη, Μηχανική Μάθηση, Αναπαράσταση Γνώσης και Βαθιά Μάθηση
Πηγή: L. Fridman, "MIT Deep Learning Basics: Introduction and Overview with TensorFlow." [Online]. Available: https://medium.com/tensorflow/mit-deep-learning-basics-introduction-and-overview-with-tensorflow-355bcd26baf0 [Accessed: 31-Aug-2020]

Τα **Τεχνητά Νευρωνικά Δίκτυα (Artificial Neural Networks - ANN)** αποτελούν τον πυρήνα της ΒΜ. Είναι ευέλικτα, ισχυρά και επεκτάσιμα, καθιστώντας τα ιδανικά για την αντιμετώπιση μεγάλων και πολύπλοκων εργασιών ΜΜ, όπως ταξινόμηση δισεκατομμυρίων εικόνων (π.χ. Google Images), υπηρεσίες με αναγνώριση ομιλίας (π.χ. Apple's Siri), προτάσεις για τα καλύτερα βίντεο για παρακολούθηση σε εκατοντάδες εκατομμυρίων χρηστών κάθε μέρα (π.χ. YouTube), διάγνωση ασθενειών όπως ο καρκίνος (πχ Google Health) και πολλές άλλες [22].

Ο όρος νευρωνικό δίκτυο ξεκινά από τη νευροβιολογία, αλλά στην ουσία μόνο οι βασικές έννοιες της ΒΜ αναπτύχθηκαν αντλώντας την έμπνευση από τη λειτουργία του ανθρώπινου εγκεφάλου. Τα σημερινά μοντέλα της ΒΜ δεν είναι μοντέλα του εγκεφάλου. Η βασική αρχή για την ΒΜ είναι η μάθηση μέσω πολλαπλών επιπέδων σύνθεσης από μη γραμμικούς μετασχηματισμούς δεδομένων εισόδου που μπορούν να εφαρμοστούν σε δομές ΜΜ που δεν είναι εμπνευσμένες από τον ανθρώπινο εγκέφαλο.

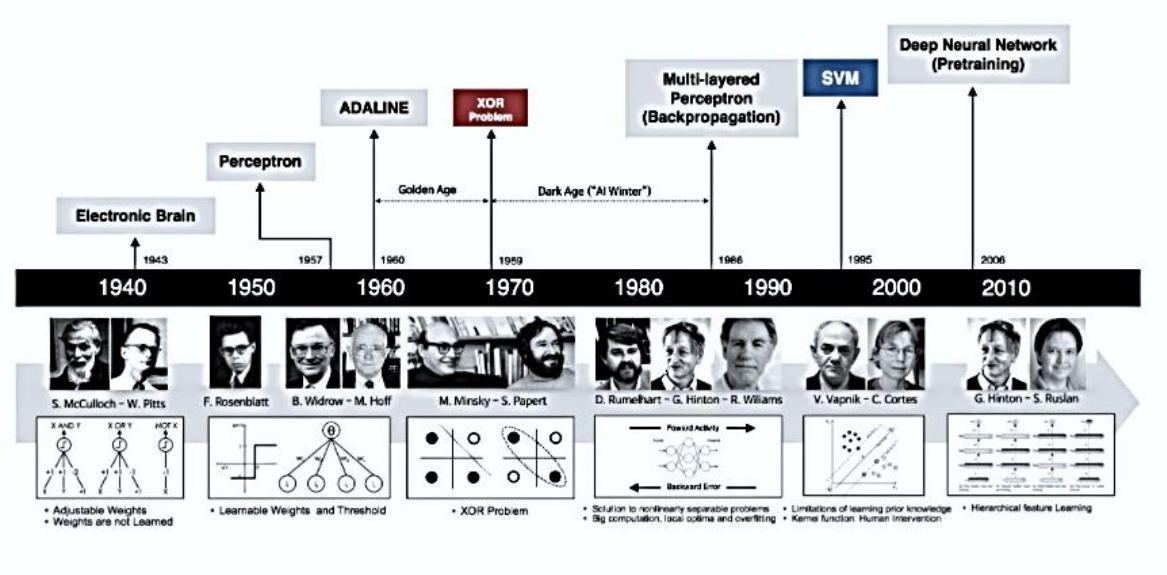
Σε ένα υψηλό επίπεδο αφαίρεσης τα ΤΝΔ είναι είτε κωδικοποιητές (encoders), είτε κωδικοποιητές (decoders) ή συνδυασμός και των δύο [24]:

- Οι κωδικοποιητές βρίσκουν μοτίβα σε πρωτογενή δεδομένα για να σχηματίσουν συμπαγείς και χρήσιμες αναπαραστάσεις.
- Οι αποκωδικοποιητές δημιουργούν υψηλής ανάλυσης δεδομένα από αυτές τις αναπαραστάσεις.

Τα δεδομένα που δημιουργούνται είναι είτε νέα παραδείγματα, είτε περιγραφική γνώση (descriptive knowledge).

3.1.1 Ιστορική Αναδρομή των ΤΝΔ

Τα ΤΝΔ είναι μία από τις πρώτες ιδέες στην ΤΝ και την ΜΜ. Οι σημαντικότερες στιγμές στην ιστορία των ΤΝΔ, παρατίθενται περιληπτικά [1], [21], [25] και αποτυπώνονται στην Εικόνα 3-4.



Εικόνα 3-4. Ιστορική αναδρομή των ΤΝΔ.
Πηγή: https://slides.com/beamandrew/deep-learning-101/

Η ιστορία των ΤΝΔ ξεκινά από το 1943, όταν ο νευροβιολόγος Warren McCulloch και ο μαθηματικός Walter Pitts το πρώτο υπολογιστικό μοντέλο τεχνητού νευρώνα εμπνευσμένο από τη λειτουργία του ανθρώπινου εγκεφάλου. Το γραμμικό τους μοντέλο αναγνώριζε δύο διαφορετικές κατηγορίες εισόδων *x* ελέγχοντας εάν μια συνάρτηση *f(x, w)* είναι θετική ή αρνητική. Το μοντέλο αυτό, δεν μπορούσε να εκπαιδευτεί για να μαθαίνει την *w*, έπρεπε η *w* να δοθεί με κώδικα. Ήταν μια απλή λογική μονάδα κατωφλίου (Thresholded Logic Unit – TLU) που προγραμματίζεται, ικανή να αναπαραστήσει τις λογικές πύλες AND, OR, NOT.

Το 1957 ο Rosenblatt παρουσίασε το Perceptron, ένα βελτιωμένο μοντέλο τεχνητού νευρώνα, με αποδεδειγμένα σωστό αλγόριθμο μάθησης της μεταβλητής *w* που μπορούσε να αναγνωρίσει γράμματα και αριθμούς.

Το 1960 η προσαρμοστική γραμμική μονάδα (ADAptive LINEar unit – ADALINE) των Widrow και Hoff, απλά επέστρεφε την τιμή της *f(x)* για να προβλέψει έναν αριθμό και επίσης μάθαινε να προβλέπει αυτόν τον αριθμό από τα δεδομένα. Για την εκπαίδευση της ADALINE χρησιμοποιήθηκε ένας αλγόριθμος, η στοχαστική κάθοδος με βάση την κλίση (stochastic gradient descent), που ακόμη και στις μέρες μας παραμένει ο βασικός αλγόριθμος εκπαίδευσης εκπαίδευσης για κάποια μοντέλα ΒΜ.

Το πρώτο κίνημα της ανάπτυξης των ΤΝΔ στις δεκαετίες 1940-1960 είναι γνωστό ως **cybernetics** (από την ελληνική λέξη κυβερνήτης).

Το 1969 οι Minsky και Papert απέδειξαν ότι τα γραμμικά μοντέλα, όπως το perceptron, δεν μπορούν να λύσουν προβλήματα άλλων συναρτήσεων, όπως το πρόβλημα της λογικής πύλης XOR. Αυτή η απόδειξη ήταν ένα πλήγμα για την δημοφιλία των ΤΝΔ και το ενδιαφέρον για αυτά περιορίστηκε μια μεγάλη χρονική περίοδο, τον «χειμώνα» των ΤΝΔ όπως αποκαλείται.

Το 1986 οι Rumelhart Hinton, και Willams προτείνουν την οργάνωση των perceptron σε δίκτυο με επίπεδα, τα **πολυεπίπεδα perceptron (MultiLayer Perceptrons – MLPs)** και έναν νέο αλγόριθμο εκπαίδευσης, τον αλγόριθμο οπισθοδιάδοσης (backpropagation), ο οποίος ακόμη και σήμερα είναι ο βασικός αλγόριθμος εκπαίδευσης των ΤΝΔ.

Το ενδιαφέρον για τα ΤΝΔ αναθερμαίνεται εγκαινιάζοντας έτσι το δέυτερο νέο κίνημα για τα ΤΝΔ, που είναι γνωστό ως **connectionism**. Η κεντρική ιδέα του connectionism είναι ότι, ένας μεγάλος αριθμός απλών υπολογιστικών μονάδων μπορούν να αποκτήσουν έξυπνη συμπεριφορά όταν εργάζονται μαζί. Πολλές βασικές αρχές θεμελιώθηκαν κατά το κίνημα του connectionism, όπως αυτή του κατανεμημένης αναπαράστασης (distributed representation). Αυτή είναι η ιδέα όπου, κάθε είσοδος σε ένα σύστημα μπορεί να αναπαρασταθεί με πολλά χαρακτηριστικά και κάθε χαρακτηριστικό εμπλέκεται στην αναπαράσταση πολλών πιθανών εισόδων. Από το 1986 μέχρι τα μέσα της δεκαετίας του 1990, γίνονται σημαντικές πρόοδοι στα ΤΝΔ, αλλά παράλληλα αναπτύσσονται άλλοι αλγόριθμοι ΜΜ, όι οποίοι έδιναν καλύτερες λύσεις σε θέματα ΤΝ, και το ενδιαφέρον για τα ΤΝΔ μειώνεται.

Το 2006, ξεκινά το τρίτο κίνημα ανάπτυξης- το οποίο εξακολουθεί να υφίσταται στις μέρες μας, όταν οι Hinton και Ruslan αποδεικνύουν πως, ένα είδος ΤΝΔ, που το ονόμασαν δίκτυο βαθειάς

πεποίθησης (Deep Believe Network – DBN), θα μπορούσε να εκπαιδευτεί αποτελεσματικά με μια διαφορετική στρατηγική, βασισμένη στη θεωρία των πιθανοτήτων και την εξαναγκασμένη μάθηση, έτσι ώστε να γίνεται η αναγνώριση προτύπων και πιο σύνθετοι υπολογισμοί. Στη συνέχεια, αποδείχθηκε από τους ερευνητές ότι, αυτή η στρατηγική μπορεί να εφαρμοστεί και σε άλλα βαθιά δίκτυα, αναθερμαίνοντας έτσι το ενδιαφέρον για τα ΤΝΔ. Από το 2012 και μετά, η ΒΜ απογειώνεται πλέον, τα βαθιά ΤΝΔ υπερτερούν έναντι όλων των υπόλοιπων συστημάτων ΜΜ.

3.1.2 Οι Λόγοι Ανάπτυξης των Νευρωνικών Δικτύων

Οι βασικές ιδέες θεμελιώδεις ιδέες για την ΒΜ, όπως για παράδειγμα ο αλγόριθμος οπισθοδιάδοσης που διατυπώθηκε το 1986 και η υπολογιστική όραση με τη βοήθεια των Συνελικτικών Νευρωνικών Δικτύων που υλοποιήθηκε το 1989. Η ΒΜ όμως, ουσιαστικά άρχισε να αναπτύσσεται μετά το 2006 και απογειώνεται μετά το 2012. Γενικά, οι βασικές τάσεις που επικρατούν και ωθούν την ανάπτυξη της ΒΜ είναι [1], [22], [26]:

- **Η πρόοδος των υπολογιστών όσον αφορά το υλικό (hardware):** Η τεράστια αύξηση της υπολογιστικής ισχύος από τη δεκαετία του 1990 καθιστά δυνατή την εκπαίδευση μεγάλων νευρωνικών δικτύων σε εύλογο χρονικό διάστημα. Η εξάπλωση του διαδικτύου, σε συνδυασμό με την ανάπτυξη των παιχνιδιών μέσω υπολογιστών (gaming) οδήγησε στην παραγωγή ισχυρότερων CPU και GPU
- **Η αύξηση των συνόλων των δεδομένων:** Σήμερα, λόγω της ψηφιοποίησης της κοινωνίας υπάρχει μια τεράστια ποσότητα δεδομένων για την εκπαίδευση νευρωνικών δικτύων και τα ΤΝΔ ξεπερνούν πλέον άλλες τεχνικές ΜΜ σε πολύ μεγάλα και πολύπλοκα προβλήματα.
- **Οι αλγοριθμικές πρόοδοι**: αν και οι πρόοδοι στον τομέα αυτόν δεν είναι τόσο μεγάλες όσο στους δύο προηγούμενους τομείς, κάποιες μικρές αλλαγές βελτίωσαν σημαντικά την απόδοση.

Επειδή το πεδίο της ΒΜ βασίζεται σε πειραματικά ευρήματα και όχι στη θεωρία, οι αλγόριθμοι εξελίσσονται μόνο όταν είναι διαθέσιμα κατάλληλα δεδομένα και υλικό. Η ΜΜ είναι μια επιστήμη μηχανικής. Τέλος, ένας σημαντικός παράγοντας είναι η δημιουργία και η συνεχής ανάπτυξη βιβλιοθηκών λογισμικού γλωσσών προγραμματισμού, όπως η Theano, η PyTorch, η TensorFlow, η Keras της γλώσσας προγραμματισμού Python και πολλές άλλες για την γλώσσα προγραμματισμού R.

3.2 Τα βασικά της Μηχανικής Μάθησης

Η ΒΜ, ως τομέας-υποσύνολο της ΜΜ, βασίζεται σε αρχές που διέπουν την ΜΜ. Ως εκ τούτου, για την κατανόηση της ΒΜ και κατ' επέκταση των ΤΝΔ, είναι απαραίτητο να είναι γνωστές οι βασικές αρχές της ΜΜ που αφορούν την εμπειρία, τις εργασίες και το μέτρο απόδοσης. Με βάση μια εργασία και ένα σύνολο σχετικών δεδομένων ως παραδείγματα σε κατάλληλη αναπαράσταση, ένας αλγόριθμος ΜΜ καλείται να αποκτήσει εμπειρία πάνω στα δεδομένα της εκπαίδευσης για να μάθει με διάφορους τρόπους, έτσι ώστε να μπορεί να δώσει απαντήσεις και προβλέψεις σε καινούρια δεδομένα που αφορούν τη συγκεκριμένη εργασία. Η απόδοση ενός αλγορίθμου αποτιμάται με βάση την ακρίβεια των προβλέψεων στα νέα δεδομένα.

3.2.1 Κατηγορίες Εργασιών

Μια εργασία της ΜΜ συνήθως περιγράφονται με όρους για το πώς ένα σύστημα ΜΜ θα επεξεργαστεί ένα **παράδειγμα (example)**[1]. Το παράδειγμα είναι ένα σύνολο **χαρακτηριστικών (features)** που έχουν μετρηθεί ποσοτικά από ένα αντικείμενο ή γεγονός το οποίο θέλουμε το σύστημα της ΜΜ να επεξεργαστεί. Το παράδειγμα τυπικά αναπαριστάνεται από ένα διάνυσμα $\boldsymbol{x} \in \mathbb{R}^n$ όπου κάθε στοιχείο του x_i είναι και ένα διαφορετικό χαρακτηριστικό. Πχ, τα χαρακτηριστικά μιας εικόνας είναι συνήθως οι τιμές των pixels της εικόνας,

Υπάρχουν πολλές εργασίες που μπορούν να πραγματοποιηθούν με τη χρήση της ΜΜ. Οι πιο κοινές είναι οι εξής [1]:

- Ταξινόμηση
- Ταξινόμηση με ελλιπείς εισόδους
- Παλινδρόμηση
- Απομαγνητοφώνηση
- Μετάφραση
- Ανίχνευση ανωμαλιών
- Σύνθεση και δειγματοληψία

3.2.2 Μέτρα απόδοσης

Για να αποτιμηθούν οι δυνατότητες ενός αλγόριθμου ΜΜ, θα πρέπει να οριστεί ένα ποσοτικό μέτρο της απόδοσής του κατά την εκπαίδευση. Το μέτρο απόδοσης είναι σχετικό με την εργασία.

[1] Αναφέρεται συχνά στη βιβλιογραφία ως παρατήρηση (observation), ως στιγμιότυπο (instance) και ως σημείο δεδομένων (data point) και στην παρούσα ΜΔΕ αυτοί οι όροι θα χρησιμοποιούνται εναλλακτικά.

Για παράδειγμα, σε εργασίες όπως αυτή της ταξινόμησης, συνήθως μετράτε η ορθότητα (accuracy) του μοντέλου. Ορθότητα είναι η αναλογία των παραδειγμάτων για τα οποία το μοντέλο παράγει τη σωστή έξοδο. Μια ισοδύναμη πληροφορία μπορεί να εξαχθεί και από το εύρος λάθους (error rate), την αναλογία των παραδειγμάτων για τα οποία παράγει μια λάθος έξοδο.

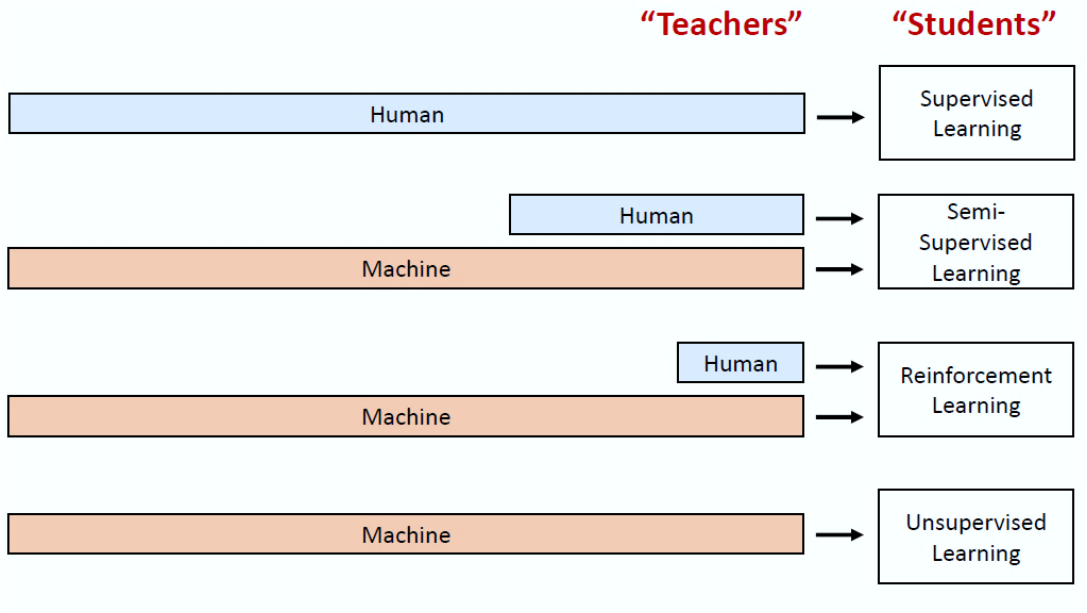
Συνήθως ενδιαφερόμαστε για το πόσο καλά αποδίδει ένας αλγόριθμος σε δεδομένα που δεν έχει ξαναδεί, καθώς αυτό είναι που καθορίζει το πόσο καλά θα λειτουργήσει όταν θα χρησιμοποιηθεί στον πραγματικό κόσμο για προγνώσεις. Συνεπώς, η απόδοση αποτιμάται με τη χρήση ενός **συνόλου δοκιμής (test set)**, το οποίο είναι διαφορετικό από αυτό που χρησιμοποιείται για την εκπαίδευση του συστήματος ΜΜ.

3.2.3 Κατηγορίες Εμπειρίας - Μάθησης

Η βασική κατηγοριοποίηση των αλγόριθμων ΜΜ γίνεται ανάλογα με το είδος της εμπειρίας που επιτρέπεται να έχουν κατά τη διάρκεια της διαδικασίας της μάθησης, δηλαδή της εκπαίδευσης. Οι αλγόριθμοι αποκτούν την εμπειρία, δηλαδή εκπαιδεύονται, με βάση το σύνολο εκπαίδευσης (training set). Έτσι, κατηγοριοποιούνται ανάλογα με το εάν η εκπαίδευση γίνεται με ανθρώπινη επίβλεψη (human) ή όχι (machine), κατά αναλογία πόσο ένας δάσκαλος είναι ο άνθρωπος ή η μηχανή που δείχνει σε έναν μαθητή τι πρέπει να κάνει. Σύμφωνα με αυτό το κριτήριο, υπάρχουν τέσσερις κύριες κατηγορίες αλγορίθμων μάθησης[2] [1], [21], [22], [23], [27], όπως φαίνεται στην Εικόνα 3-5:

- **Επιβλεπόμενη ή εποπτευόμενη μάθηση (supervised learning)**
- **Μη επιβλεπόμενη ή μη εποπτευόμενη μάθηση (unsupervised learning)**
- **Ημι- επιβλεπόμενη ή ημι- εποπτευόμενη μάθηση (semi-supervised learning)**
- **Ενισχυτική μάθηση (reinforcement learning)**

[2] Στην ελληνική βιβλιογραφία οι όροι εποπτεία και επίβλεψη χρησιμοποιούνται εναλλακτικά αποδίδοντας την ίδια έννοια. Το ίδιο ισχύει και στην παρούσα ΜΔΕ.



Εικόνα 3-5. Η ανθρώπινη παρέμβαση σε σχέση με την μηχανική στους τρόπους μάθησης
Πηγή: L. Fridman, "Deep Learning Basics." [Online]. Available: https://www.dropbox.com/s/c0g3sc1shi63x3q/deep_learning_basics.pdf?dl=0 [Accessed: 30-Sep-2020]

3.2.3.1 Επιβλεπόμενη Μάθηση

Στην **επιβλεπόμενη μάθηση** το σύνολο δεδομένων με βάση το οποίο θα αποκτήσει εμπειρία ο αλγόριθμος, δηλαδή θα εκπαιδευτεί για να μάθει, περιέχει τα παραδείγματα με τα χαρακτηριστικά τους και σε παράδειγμα - στοιχείο του συνόλου συνδέεται με ένα **στόχο (target)** ή αλλιώς **ετικέτα (label)**. Τέτοιου είδους δεδομένα, ονομάζονται **δεδομένα με ετικέτα (labeled)**. Πχ, στο πρόβλημα της διάγνωσης μιας ασθένειας που αναφέραμε στην παράγραφο 3.1, κάθε παράδειγμα έχει τις τιμές των βιομετρικών χαρακτηριστικών σε μορφή διανύσματος $\boldsymbol{x}$ και μια συνδεμένη ετικέτα-στόχο y με διακριτή τιμή, πχ 0 εάν για το παράδειγμα δεν υπήρχε ασθένεια και 1 εάν υπήρχε. Ο όρος επιβλεπόμενη μάθηση πηγάζει από την θεώρηση ότι ο στόχος y δίνεται από έναν δάσκαλο που δείχνει κατά κάποιο τρόπο στον μαθητή- σύστημα της ΜΜ- τι ακριβώς περιμένει από αυτόν να κάνει. Οι πιο συνηθισμένες εργασίες επιβλεπόμενης μάθησης είναι η **ταξινόμηση ή κατηγοριοποίηση (classification)**, όπου από το σύστημα αναμένεται η πρόβλεψη για την ταξινόμηση των δεδομένων σε διάφορες εκ των προτέρων γνωστές κλάσεις και η **παλινδρόμηση (regression)** όπου από το σύστημα αναμένεται η πρόβλεψη μιας τιμής. Τα MLP είναι πλέον ο κυρίαρχος αλγόριθμος ΜΜ για εργασίες επιβλεπόμενης μάθησης.

3.2.3.2 Μη-επιβλεπόμενη Μάθηση

Στην **μη-επιβλεπόμενη μάθηση** ο αλγόριθμος εκπαιδεύεται στο σύνολο των δεδομένων που περιέχει τα χαρακτηριστικά, μαθαίνοντας τις χρήσιμες ιδιότητες της δομής του συνόλου δεδομένων. Τα παραδείγματα δεν περιέχουν ετικέτες – στόχους (unlabeled). Η μη επιβλεπόμενη μάθηση αφορά την παρατήρηση διάφορων παραδειγμάτων και προσπαθεί έμμεσα ή άμεσα να μάθει την κατανομή πιθανότητας *p(x)* ή τις ενδιαφέρουσες ιδιότητες αυτής της κατανομής. Οι πιο συνηθισμένες εργασίες μη επιβλεπόμενης μάθησης είναι η **συσταδοποίηση (clustering)** όπου από το σύστημα αναμένεται ο διαχωρισμός των δεδομένων σε ομοειδείς συστάδες και η **μείωση διαστάσεων (dimensionality reduction)** όπου από το σύστημα αναμένεται η εξαγωγή χρήσιμων ιδιοτήτων των δεδομένων.

3.2.3.3 Ενισχυτική Μάθηση

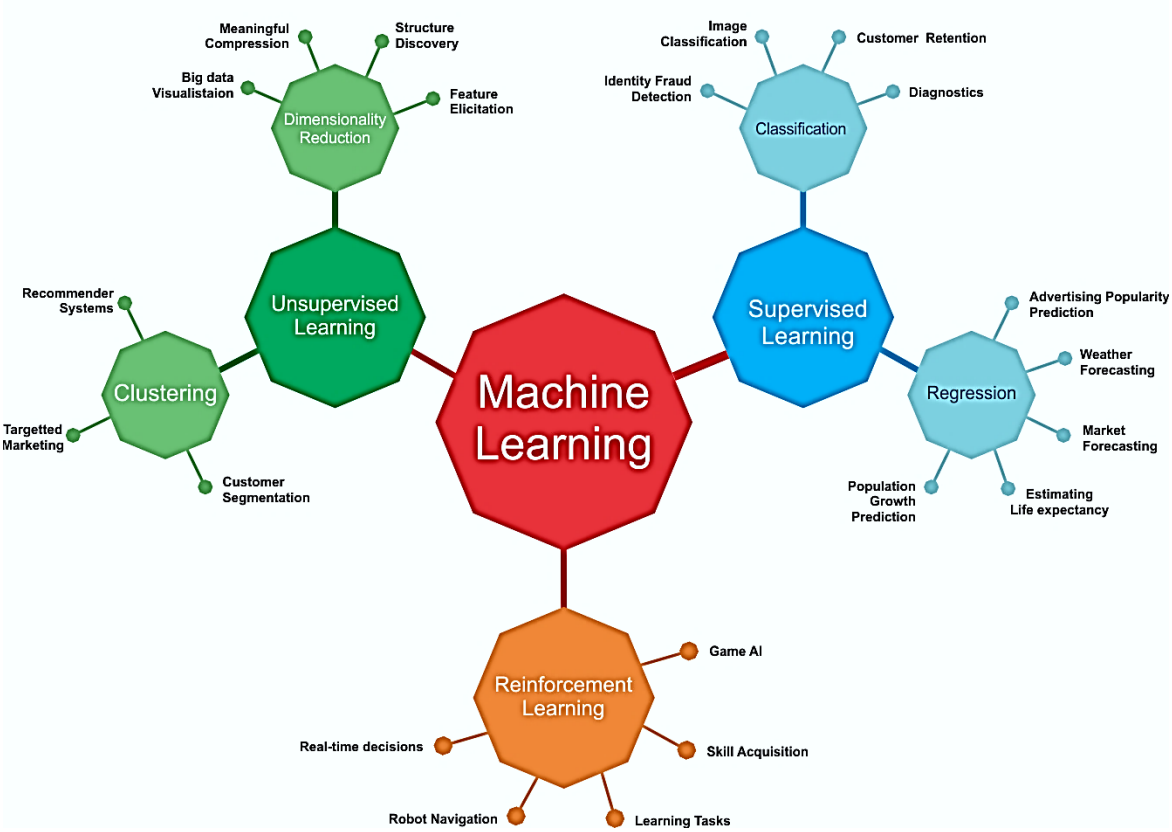
Η **ενισχυτική μάθηση** είναι μια τελείως διαφορετική κατηγορία αλγορίθμων. Ένα σύστημα εκμάθησης που ονομάζεται πράκτορας (agent), μπορεί να παρατηρεί το περιβάλλον, να επιλέγει και να εκτελεί ενέργειες και να επιβραβεύεται ή να τιμωρείται ανάλογα με την ορθότητα των ενεργειών του. Ο πράκτορας θα πρέπει να μαθαίνει μόνος του με στόχο να επιβραβεύεται συνεχώς. Οι εργασίες ενισχυτικής μάθησης είναι διαφόρων ειδών, όπως η εκμάθηση πλοήγησης στα ρομπότ και η εκμάθηση παιχνιδιών.

3.2.3.4 Ημι-επιβλεπόμενη Μάθηση

Τέλος, η ημι-επιβλεπόμενη μάθηση είναι ένας συνδυασμός επιβλεπόμενης και μη επιβλεπόμενης μάθησης, όπου ο αλγόριθμος καλείται να εκπαιδευτεί σε σύνολο δεδομένων που περιέχουν παραδείγματα με ετικέτα και παραδείγματα χωρίς ετικέτα.

3.2.4 Εφαρμογές Μηχανικής Μάθησης

Στις μέρες μας, η ΒΜ και ΤΝΔ αποτελούν την κύρια τάση στην ΜΜ [28] και καλύπτουν κάθε τύπο μάθησης σε ένα ευρύ πεδίο της επιστήμης και της τεχνολογίας, εκεί όπου εφαρμόζονται οι σύγχρονες αναπτυσσόμενες τεχνολογίες όπως τα Μεγάλα Δεδομένα (Big Data), ο Υπολογισμός Νέφους και Άκρων (Cloud Computing and Edge Computing), ο Κινητός Υπολογισμός (Mobile Computing), το διαδίκτυο των Πραγμάτων (Internet of Things) και η Κυβερνοασφάλεια (Cybersecurity) [29].



Εικόνα 3-6. Εφαρμογές μηχανικής μάθησης ανάλογα με την κατηγορία μάθησης και εργασίας
Πηγή: https://www.oreilly.com/library/view/java-deep-learning/9781788997454/a8fce962-51dd-4e29-a7f9-9bf4fd245b1d.xhtml

Ουσιαστικά, οι τομείς όπου αναπτύσσεται η ΒΜ είναι όλοι οι τομείς της επιστήμης και της τεχνολογίας του πραγματικού κόσμου, όπως η ιατρική, το εμπόριο, η εκπαίδευση, η ψυχαγωγία κλπ. Στην Εικόνα 3-6, φαίνονται κάποιες ενδεικτικές εφαρμογές των αλγορίθμων ΜΜ, ανάλογα με τον τύπο της μάθησης και την κατηγορία των εργασιών του κάθε τύπου μάθησης.

Στο σημείο αυτό, σημειώνεται ότι το πρόβλημα της διάγνωσης του καρκίνου του μαστού είναι ένα πρόβλημα επιβλεπόμενης μάθησης για δυαδική ταξινόμηση δεδομένων (binary classification). Συνεπώς, από εδώ και πέρα τα παραδείγματα θα αφορούν κατά κύριο λόγο το συγκεκριμένο θέμα.

3.2.5 Εκπαίδευση, Δοκιμή και Αποτίμηση Μοντέλου Επιβλεπόμενης Μάθησης

Ο πυρήνας του συστήματος που θα υλοποιηθεί για μια εφαρμογή ΜΜ, είναι ο αλγόριθμος που θα επιλεγεί και το **προγνωστικό μοντέλο (predictive model)** που θα δημιουργηθεί με βάση τα δεδομένα της εφαρμογής. Ο υποκείμενος αλγόριθμος ΜΜ βρίσκει μια μαθηματική συνάρτηση που μπορεί να παράγει με βάση τα χαρακτηριστικά το σωστό αποτέλεσμα για τον στόχο της

πρόγνωσης μαθαίνοντας από τα παραδείγματα. Η εύρεση της απλούστερης μαθηματικής συνάρτησης που προβλέπει τα αποτελέσματα με το απαιτούμενο επίπεδο ακρίβειας με την εκπαίδευση είναι η καρδιά της ΜΜ [1].

Στην παρούσα ΜΔΕ διερευνάται το πρόβλημα του καρκίνου του μαστού, όσον αφορά τη διάγνωση καλοήθειας ή μη ενός όγκου. Είναι μία εργασία δυαδικής ταξινόμησης, όπου καλούμαστε με βάση ένα σύνολο δεδομένων από κλινικές παρατηρήσεις για την μορφολογία των όγκων και τα αντίστοιχα αποτελέσματα για ύπαρξη καρκίνου ή όχι, δηλαδή ένα σύνολο δεδομένων με ετικέτα, να υλοποιήσουμε ένα σύστημα ΜΜ που για νέες κλινικές παρατηρήσεις θα μας δώσει προβλέψεις για την ύπαρξη καρκίνου ή όχι με όσο το δυνατόν μεγαλύτερη ακρίβεια. Συνεπώς, με δεδομένο το πρόβλημα και το σύνολο των παραδειγμάτων που διαθέτουμε εξαρχής, καλούμαστε να επιλέξουμε πρώτα έναν αλγόριθμο πάνω στον οποίο θα βασιστούμε για να δημιουργήσουμε το προγνωστικό μοντέλο. Η διαδικασία δημιουργίας του μοντέλου είναι η **εκπαίδευση (training)** [1]**.**

Υπάρχουν πολλοί αλγόριθμοι επιβλεπόμενης μάθησης την εργασία της δυαδικής ταξινόμησης, όπως μεταξύ των άλλων η λογιστική παλινδρόμηση (logistic regression), οι μηχανές διανυσματικής υποστήριξης (Support Vector Machines - SVM), τα δένδρα απόφασης (Decision Trees) και βέβαια τα MLPs [22].

Έχοντας ορίσει με σαφήνεια το πρόβλημα, διαθέτοντας ένα αρκετά αντιπροσωπευτικό σύνολο εκπαίδευσης με τα παραδείγματα και τους στόχους και γνωρίζοντας ποιοι αλγόριθμοι είναι διαθέσιμοι, η διαδικασία της εκπαίδευσης του μοντέλου ακολουθεί μια συγκεκριμένη ροή εργασιών ανεξάρτητα από την επιλογή του αλγόριθμου με στόχο την βελτιστοποίηση του μοντέλου και στη συνέχεια την επικύρωσή του μέσω δοκιμών και την αποτίμησή του.

3.2.5.1 Βελτιστοποίηση

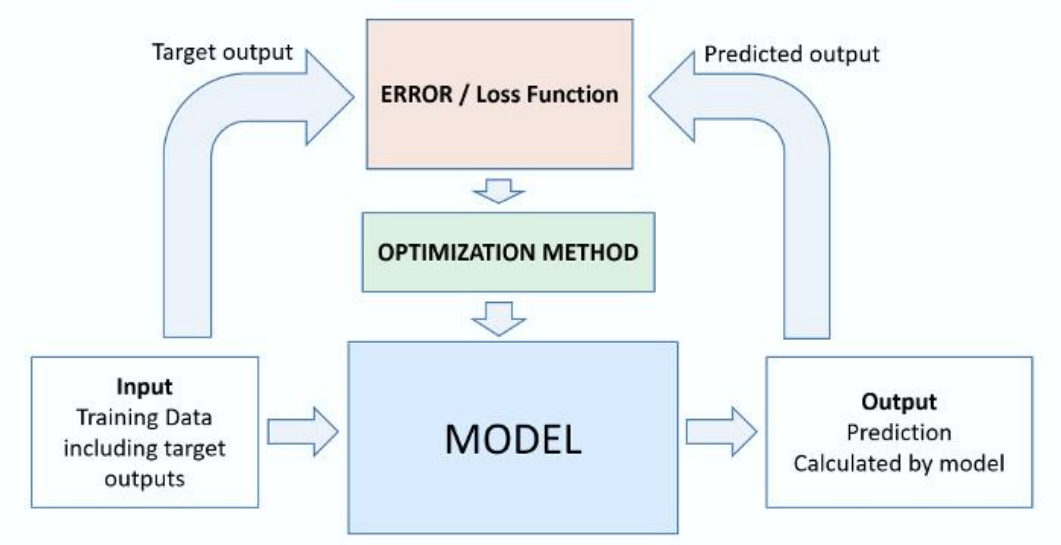
Η ροή εργασιών ροή των εργασιών για την εκπαίδευση ενός μοντέλου ΜΜ για όποιον αλγόριθμο επιλεγεί, όπως περιγράφεται και διαγραμματικά στην Εικόνα 10, έχει ως εξής [1], [22], [23]:

- *Είσοδος:* Παραδείγματα που συμπεριλαμβάνουν τις ετικέτες σε μορφή κατάλληλη για τον υπολογιστή
- *Εφαρμογή του αλγόριθμου στο μοντέλο*: Ουσιαστικά εδώ επιλέγεται η μορφή που θα έχει η συνάρτηση που θα υπολογίσει την πρόβλεψη για την έξοδο με βάση συγκεκριμένες παραμέτρους που εξαρτώνται από τον αλγόριθμο
- *Έξοδος*: Υπολογισμός πρόβλεψης από το μοντέλο με τυχαίες παραμέτρους

- *Απόδοση του μοντέλου*: Πρόκειται για την αποτίμηση του **λάθους εκπαίδευσης (training error)** που προκύπτει από την έξοδο σε σχέση με την τιμή της ετικέτας. Εδώ, εισάγεται ο όρος της **συνάρτησης κόστους (loss function)**. Η συνάρτηση αυτή μετρά την απόδοση του μοντέλου και επιλέγεται ανάλογα με το είδος του προβλήματος. Η συνάρτηση κόστους είναι αυτή που θα οδηγήσει το μοντέλο προς τη σωστή κατεύθυνση για τη συνέχεια της εκπαίδευσης.
- *Βελτιστοποίηση*: Στόχος της βελτιστοποίησης (optimization) είναι η ελαχιστοποίηση της συνάρτησης κόστους, προκειμένου να βρεθούν οι ιδανικές παράμετροι του μοντέλου που θα δίνουν προβλέψεις όσο το δυνατόν πλησιέστερα στις ετικέτες. Αυτό επιτυγχάνεται με υπολογιστικές μεθόδους για την ελαχιστοποίηση μαθηματικών συναρτήσεων, δηλαδή με **μεθόδους βελτιστοποίησης (optimization methods)**, τους **βελτιστοποιητές (optimizers)**.

Η εκπαίδευση είναι μια επαναληπτική διαδικασία η οποία επαναλαμβάνεται μέχρι να ικανοποιηθεί το κριτήριο της σύγκλισης, δηλαδή να ελαχιστοποιηθεί η συνάρτηση κόστους ή αλλιώς το λάθος να τείνει στο μηδέν. Κάθε επανάληψη, ονομάζεται **εποχή (epoch).**

Υπερπαράμετροι (hyperparameters) είναι οι ρυθμίσεις που γίνονται από τον χρήστη για τον έλεγχο της συμπεριφοράς του μοντέλου. Μία σημαντική υπερπαράμετρος είναι η **χωρητικότητα (capacity)** του μοντέλου, που είναι η ικανότητά του μοντέλου να προσαρμόζεται σε μια ποικιλία συναρτήσεων [1]. Επίσης, οι συναρτήσεις κόστους και οι βελτιστοποιητές μπορούν να θεωρηθούν ως **υπερπαράμετροι** του μοντέλου, εφόσον είναι διαθέσιμες διάφορες συναρτήσεις κόστους και μέθοδοι βελτιστοποίησης για τον σχεδιαστή του μοντέλου.



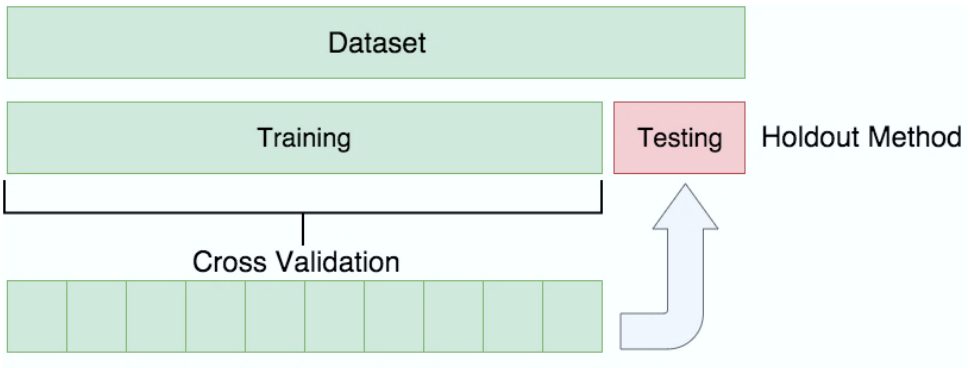
Εικόνα 3-7. Ροή εργασιών διαδικασίας εκπαίδευσης
Πηγή: https://www.deepnetts.com/blog/from-basic-machine-learning-to-deep-learning-in-5-minutes.html

Η κύρια πρόκληση όμως στην ΜΜ είναι ότι, ο αλγόριθμος θα πρέπει να αποδίδει καλά σε νέα δεδομένα που δεν έχει δει ξανά, και όχι μόνο για τα δεδομένα με βάση τα οποία έχει εκπαιδευτεί. Η δυνατότητα να αποδίδει καλά σε νέες εισόδους δεδομένων ονομάζεται **γενίκευση (generalization).** Συνεπώς, κατά την εκπαίδευση το μοντέλο πρέπει να αποτιμάται μέσω της **δοκιμής (testing)** .

3.2.5.2 Δοκιμή και Επικύρωση

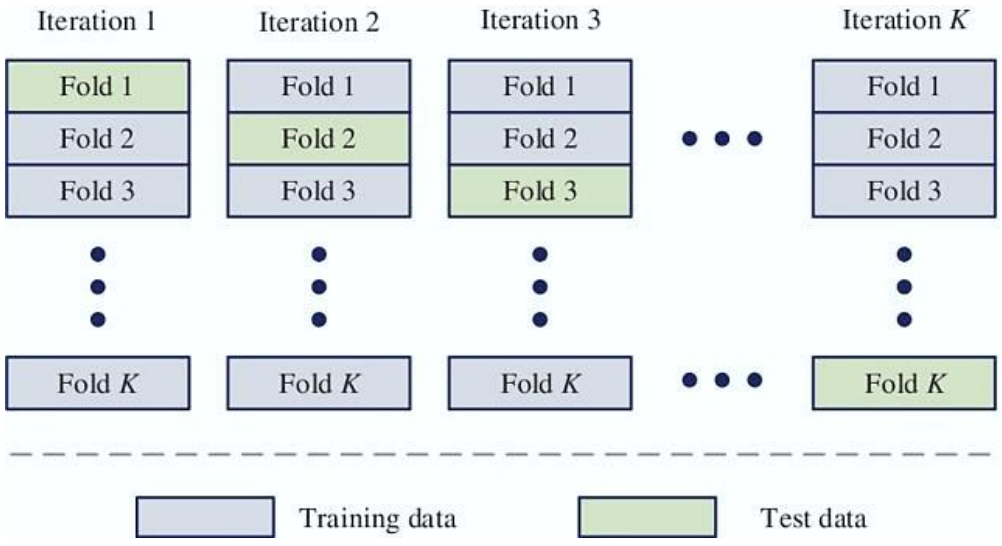
Η εκπαίδευση του μοντέλου αποτελεί ένα πρόβλημα βελτιστοποίησης. Αυτό όμως που διαχωρίζει την ΜΜ από την βελτιστοποίηση είναι το λάθος της **γενίκευσης (generalization error)**, που ονομάζεται επίσης και **λάθος δοκιμής (test error)** που πρέπει επίσης να είναι μικρό. Ως λάθος γενίκευσης ορίζεται η αναμενόμενη τιμή λάθους σε μια νέα είσοδο. Για τον υπολογισμό του λάθους γενίκευσης χρησιμοποιείται η ίδια συνάρτηση κόστους που χρησιμοποιείται στο σύνολο δοκιμής.

Τυπικά, η εκτίμηση του λάθους γενίκευσης μετράτε από την απόδοσή του σε ένα **σύνολο επικύρωσης (validation set)** παραδειγμάτων. Μια κοινή πρακτική όταν σχεδιάζουμε ένα μοντέλο είναι το σύνολο των δεδομένων των παραδειγμάτων της δοκιμής να χωρίζεται σε δύο υποσύνολα σε μια αναλογία πχ 80%-20% , όπου το 80% των παραδειγμάτων θα αποτελεί το σύνολο της εκπαίδευσης και το 20% θα αποτελεί το σύνολο της επικύρωσης. Αυτή η τεχνική διαχωρισμού ονομάζεται **hold-out validation** (Εικόνα 3-8). Ουσιαστικά, επικυρώνουμε την απόδοση του μοντέλου μέσω της δοκιμής, γι' αυτό χρησιμοποιείται και ο όρος **επικύρωση (validation)** [23]**.**



Εικόνα 3-8. Hold-out validation
Πηγή: https://miro.medium.com/max/1896/1*r73p1rxMZWnZLoYi5Odf4A.png

Η τεχνική της διαίρεσης του συνόλου δεδομένων σε ένα σταθερό σύνολο εκπαίδευσης και ένα σταθερό σύνολο επικύρωσης, εάν το σύνολο δοκιμής είναι μικρό εισάγει στατιστική αβεβαιότητα στο εκτιμώμενο μέσο λάθος επικύρωσης και αυτό δυσκολεύει τον ισχυρισμό για το εάν ένας Α αλγόριθμος είναι καλύτερος από τον Β. Όταν ένα σύνολο δεδομένων είναι μεγάλο και έχει χιλιάδες παραδείγματα, δεν υφίσταται τέτοιο πρόβλημα. Αντίθετα, όταν το σύνολο δεδομένων είναι μικρό, τότε εναλλακτικές διαδικασίες δίνουν τη δυνατότητα της εκτίμησης του μέσου λάθους επικύρωσης, έστω και εάν αυξάνεται το υπολογιστικό κόστος [1]. Αυτές οι διαδικασίες βασίζονται στην ιδέα στο να επαναλαμβάνονται οι υπολογισμοί της εκπαίδευσης και της επικύρωσης σε διαφορετικά επιλεγμένα τυχαία υποσύνολα ή διαχωρισμούς του πρωτογενούς συνόλου δεδομένων. Μια από τις πιο συνηθισμένες τέτοιες διαδικασίες είναι η **διασταυρωμένη επικύρωση k τμημάτων (k-fold Cross Validation - CV)**. Το σύνολο δεδομένων χωρίζεται σε k υποσύνολα. Το λάθος επικύρωσης εκτιμάται παίρνοντας το μέσο λάθος για k δοκιμασίες-επαναλήψεις (iterations). Στη δοκιμασία i το i-οστό υποσύνολο των δεδομένων χρησιμοποιείται ως σύνολο επικύρωσης και τα υπόλοιπα δεδομένα χρησιμοποιούνται ως σύνολο εκπαίδευσης [1], [23]. Η τεχνική αυτή αποτυπώνεται στην Εικόνα 3-9.



Εικόνα 3-9. k-fold Cross Validation
Πηγή:
https://www.researchgate.net/profile/Mingchao_Li/publication/331209203/figure/fig2/AS:728070977748994@1550597056956/K-fold-cross-validation-method_W640.jpg

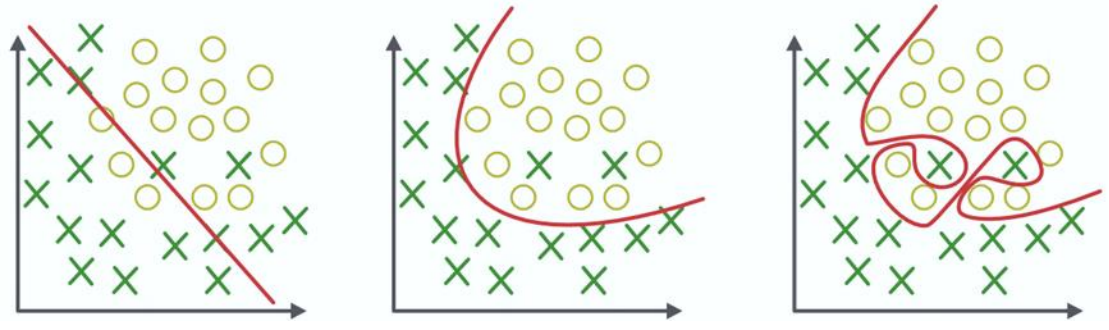
3.2.5.3 Αποτίμηση

Η εκπαίδευση έχει ως απώτερο στόχο την δημιουργία προγνώσεων. Όταν ολοκληρωθεί η εκπαίδευση, χρειάζεται ένα ποσοτικό μέτρο για την **αποτίμηση (evaluation)** του μοντέλου όσον αφορά τις προγνώσεις. Αυτό είναι το **μέτρο απόδοσης (performance measure)** και το πιο απλό μέτρο είναι η **ορθότητα (accuracy)**. Η ορθότητα είναι το ποσοστό των ορθών προβλέψεων του μοντέλου σε σχέση με τα δεδομένα των ετικετών του συνόλου εκπαίδευσης [1], [22]. Σε πολλές εργασίες, όπως αυτή της ταξινόμησης, δεν αρκεί μόνο η ακρίβεια ως μέτρο, όπως θα δούμε αναλυτικότερα όταν θα παρουσιάσουμε τη δυαδική ταξινόμηση. Σε κάθε περίπτωση, τα μέτρα απόδοσης εξαρτώνται από το είδος της εφαρμογής.

Σε περίπτωση που το μέτρο απόδοσης δεν είναι ικανοποιητικό, σημαίνει ότι υπάρχουν προβλήματα κατά τη διάρκεια της εκπαίδευσης, όπως θα δούμε στην αμέσως επόμενη παράγραφο.

3.2.5.4 Το Πρόβλημα του Underfitting και Overfitting

Κατά τη διάρκεια της εκπαίδευσης μπορεί να εμφανιστούν προβλήματα στη γενίκευση τα οποία οφείλονται στην κακή επιλογή των υπερπαραμέτρων του μοντέλου ή στον αριθμό και την ποιότητα των δεδομένων που περιέχονται στο σύνολο εκπαίδευσης. Τα προβλήματα αυτά έχουν μελετηθεί με τη βοήθεια της θεωρίας της στατιστικής μάθησης (η ΜΜ είναι στατιστική μάθηση) [1] και θα τα παρουσιάσουμε πολύ απλά και συνοπτικά με ένα παράδειγμα μέσω της Εικόνας 3-10, όπου θέλουμε να διαχωρίσουμε δύο κλάσεις αντικειμένων που απεικονίζονται σε ένα καρτεσιανό επίπεδο (x, y) και με βάση όσα περιγράφουν οι συγγραφείς στις [1], [22].



Εικόνα 3-10. Underfitting – Fitting - Overfitting

Αριστερά στην εικόνα, φαίνεται η προσπάθεια διαχωρισμού με μια ευθεία γραμμή η οποία είναι μια πρώτου βαθμού πολυωνυμική συνάρτηση του y ως προς το x και προφανώς το μοντέλο αυτό

δεν επαρκεί για τη λύση του προβλήματος. Στο κέντρο της εικόνας ο διαχωρισμός γίνεται με μια κυρτή καμπύλη η οποία ουσιαστικά είναι μία δεύτερου βαθμού συνάρτηση του y ως προς το x (δεύτερου βαθμού πολυώνυμο) και το μοντέλο ταιριάζει ικανοποιητικά, έχουμε αυξήσει τον βαθμό του πολυωνύμου του μοντέλου. Δεξιά της εικόνας, ο διαχωρισμός είναι μια πολύ μεγαλύτερου βαθμού συνάρτηση του y ως προς το x και το μοντέλο ταιριάζει απόλυτα και με μεγάλη λεπτομέρεια και η χωρητικότητά του έχει αυξηθεί πολύ. Η υπερπαράμετρος σε αυτό το μοντέλο είναι ο βαθμός του πολυωνύμου και χαρακτηρίζει την χωρητικότητά του.

Στην πρώτη περίπτωση, έχουμε την περίπτωση που στην ΜΜ ονομάζεται **υποταίριασμα ή υποπροσαρμογή (underfitting).** Το μοντέλο δεν μπορεί να συλλάβει την τάση της κατανομής των δεδομένων και δεν ταιριάζει για αυτά τα δεδομένα, οπότε δεν μπορεί να επιτευχθεί ο στόχος της εκπαίδευσης. Στην τρίτη περίπτωση έχουμε το **υπερταίριασμα ή υπερπροσαρμογή (overfitting)** όπου το μοντέλο διαχωρίζει μεν πολύ σωστά τα δεδομένα, αλλά με μεγάλη λεπτομέρεια και θα αποτύχει στην περίπτωση εισαγωγής νέων δεδομένων όπου θα δημιουργηθεί μεγάλο λάθος γενίκευσης γιατί εισάγεται και ο θόρυβος (λέγοντας θόρυβο εννοούμε νέα δεδομένα που δεν αντιπροσωπεύουν τις πραγματικές ιδιότητες αλλά είναι τυχαία). Τέλος, έχουμε την δεύτερη περίπτωση ενός καλού προσαρμοσμένου μοντέλου, δηλαδή ενός μοντέλου που θα αποδίδει καλά στην πρόβλεψη για νέα δεδομένα [1], [30].

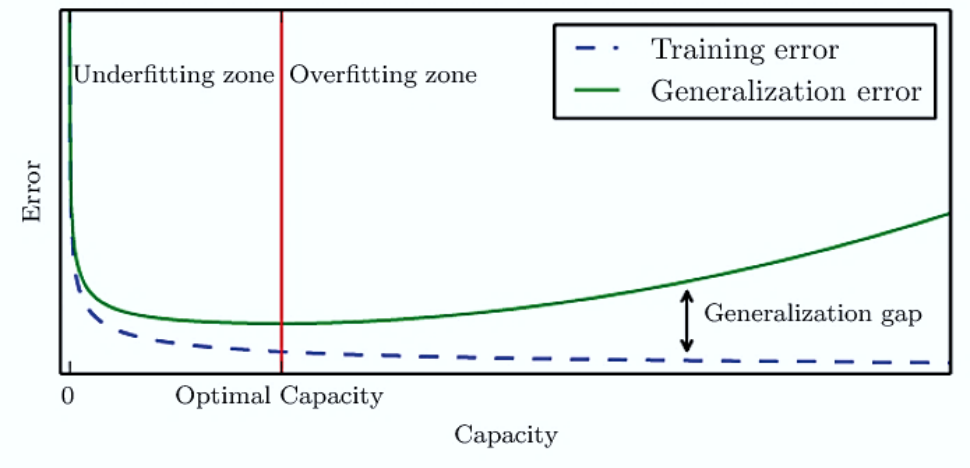
Συνεπώς, και σύμφωνα με τους συγγραφείς στην [1], οι δύο παράγοντες που ορίζουν πόσο καλά αποδίδει ένα μοντέλο είναι η δυνατότητά του:

1. Να κάνει το λάθος εκπαίδευσης μικρό
2. Να μικραίνει τη διαφορά μεταξύ του λάθους εκπαίδευσης και του λάθους

Αυτοί οι δύο παράγοντες αντιστοιχούν σε δύο βασικές προκλήσεις στην ΜΜ: την **υπερεκπαίδευση (overtraining)** που προκαλεί το overfitting και την **υποεκπαίδευση (uundertraining)** που προκαλεί το underfitting. Το underfitting συμβαίνει όταν το μοντέλο δεν είναι ικανό να παράγει μικρή τιμή λάθους στο σύνολο εκπαίδευσης. Το overfitting συμβαίνει όταν το διάστημα μεταξύ του λάθους εκπαίδευσης και του λάθους δοκιμής είναι πολύ μεγάλο. Ο έλεγχος για το underfitting και το overfitting επιτυγχάνεται με την μεταβολή της χωρητικότητας του μοντέλου, δηλαδή τη ρύθμιση των παραμέτρων του.

Στην Εικόνα 3-11 δίνεται η σχέση μεταξύ της χωρητικότητας και του λάθους. Το λάθος εκπαίδευσης και το λάθος γενίκευσης συμπεριφέρονται διαφορετικά. Αριστερά στην γραφική

παράσταση φαίνεται ότι και τα δύο λάθη είναι μεγάλα, είναι η ζώνη του underfitting. Καθώς αυξάνεται η χωρητικότητα το λάθος εκπαίδευσης μειώνεται, αλλά αυξάνεται το λάθος γενίκευσης, η διαφορά μεταξύ τους αυξάνεται οπότε περνάμε στη ζώνη του overfitting. Η γραμμή που οριοθετεί τις δύο ζώνες αντιστοιχεί στη βέλτιστη χωρητικότητα (optimal capacity). Η χωρητικότητα του μοντέλου συχνά αναφέρεται και ως **πολυπλοκότητα (complexity).**



Εικόνα 3-11. Σχέση χωρητικότητας και λάθους
Πηγή: I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016

Ο περιορισμός της χωρητικότητας του μοντέλου προκειμένου να γίνει πιο απλό και να μειωθεί η πιθανότητα του overfitting ονομάζεται **εξομάλυνση**[3] **(regularization)** [22]. Η εξομάλυνση είναι κάθε τροποποίηση που γίνεται σε ένα μοντέλο μάθησης και που έχει ως σκοπό την μείωση του λάθους γενίκευσης και όχι του λάθους εκπαίδευσης. Είναι μία από τις κύριες έννοιες στο πεδίο της ΜΜ που συναγωνίζεται την σπουδαιότητα της βελτιστοποίησης [1], [23]. Οι μέθοδοι εξομάλυνσης εξαρτώνται από τον αλγόριθμο της ΜΜ που έχει επιλεγεί για τη δημιουργία του μοντέλου.

[3] Ο όρος regularization μπορεί να αποδοθεί και ως κανονικοποίηση. Επιλέγεται η απόδοση εξομάλυνση για διαχωρισμό με τον όρο normalization, ο οποίος χρησιμοποιείται στην ΜΜ για μετασχηματισμούς δεδομένων.

3.3 Δυαδική Ταξινόμηση (Binary Classification)

Η **ταξινόμηση ή κατηγοριοποίηση (classification)** είναι μια ομάδα προβλημάτων της επιβλεπόμενης μάθησης, όπου σκοπός είναι η πρόγνωση της διακριτής τάξης που ανήκει ένα νέο δεδομένο με βάση την εμπειρία που έχει αποκτηθεί από προηγούμενα δεδομένα.

Στην παράγραφο 3.1 αναφέραμε ως παράδειγμα μια εργασία ταξινόμησης, που αφορούσε τη διάγνωση μιας ασθένειας. Σε αυτή την περίπτωση, το σύνολο εκπαίδευσης περιέχει παραδείγματα με ετικέτα, όπου σε κάθε παράδειγμα υπάρχουν τα χαρακτηριστικά που είναι σχετικά με την ασθένεια και η ετικέτα που χαρακτηρίζει την τάξη ως αρνητική ή θετική και παίρνει δύο διακριτές τιμές. Σε αυτή την περίπτωση έχουμε την **δυαδική ταξινόμηση (binary classification)**.

Σε περιπτώσεις όπου οι διακριτές τιμές των ετικετών είναι περισσότερες από δύο, τότε αναφερόμαστε σε προβλήματα **ταξινόμησης πολλών κλάσεων (multiclass classification)**. Το προγνωστικό μοντέλο που μαθαίνει από έναν αλγόριθμο εποπτευόμενης ΜΜ μπορεί να καταχωρήσει οποιαδήποτε από τις δυνατές κλάσεις που υπάρχουν στο σύνολο εκπαίδευσης σε ένα νέο στιγμιότυπο χωρίς ετικέτα.

Κλασσικό παράδειγμα μιας τέτοιας ταξινόμησης αποτελεί η αναγνώριση χειρόγραφων ψηφίων. Μπορούμε να συλλέξουμε πολλά παραδείγματα χειρόγραφων ψηφίων από το 0 έως το 9 για να δημιουργήσουμε ένα σύνολο εκπαίδευσης για το μοντέλο. Τα ψηφία (“0”, “1”, “2” … “9”) θα αναπαριστούν τις διαφορετικές διακριτές τιμές ή τις κλάσεις με ετικέτα που θέλουμε να προβλέπει το μοντέλο. Έτσι, εάν ένας χρήστης γράψει ένα ψηφίο μέσω μιας συσκευής εισόδου, το προγνωστικό μοντέλο θα είναι σε θέση να προβλέψει πιο είναι αυτό το ψηφίο με μεγάλη ακρίβεια. Προφανώς, το μοντέλο είναι σε θέση να αναγνωρίσει μόνο τα ψηφία και όχι τα γράμματα της αλφαβήτου, εφόσον αυτά δεν περιλαμβάνονται στο σύνολο εκπαίδευσης.

Στις επόμενες παραγράφους περιγράφονται οι βασικές έννοιες και στοιχεία του προβλήματος της δυαδικής ταξινόμησης που είναι σημαντικά για τους υπολογισμών στην διαδικασία εκπαίδευσης και αποτίμησης ενός μοντέλου για την επίλυση του προβλήματος. Η διαδικασία της εκπαίδευσης ξεκινά από την αναπαράσταση των δεδομένων που είναι η μεγάλη δύναμη της ΒΜ, άρα και των MLPs. Συνεπώς, είναι σημαντική η παρουσίαση του θεωρητικού υπόβαθρου που απαιτείται για την κατανόηση της αναπαράστασης των δεδομένων.

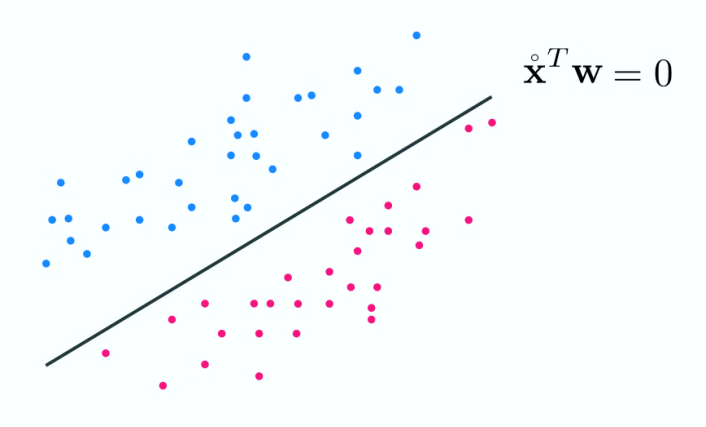
3.3.1 Γραμμικό και Μη-γραμμικό Μοντέλο Δυαδικής Ταξινόμησης

Γενικά, στη δυαδική ταξινόμηση έχουμε ένα σύνολο δεδομένων P παραδειγμάτων με ετικέτα $\{(\boldsymbol{x}_p, y_p)\}_{p=1}^{P}$ όπου κάθε στοιχείο του $\boldsymbol{x}_p$ είναι ένα διάνυσμα χαρακτηριστικών του οποίου κάθε διάσταση $j= 1,2, \dots N$ περιέχει μια τιμή που περιγράφει το παράδειγμα και το y_p είναι η ετικέτα που παίρνει μόνο δύο τιμές από το σύνολο {0,1}, δηλαδή αποτελείται από δύο τάξεις-κλάσεις για τις οποίες θέλουμε ο αλγόριθμος να μάθει πως θα τις διαχωρίζει αυτόματα[4]. Αυτό το σύνολο δεδομένων χρησιμοποιεί ο αλγόριθμος για να δημιουργήσει το μοντέλο. Έχουμε τον χαρακτηρισμό της εργασίας Τ: την πρόβλεψη της κλάσης *y* από το διάνυσμα $\boldsymbol{x}$.

Στο σημείο αυτό κάνουμε μια μικρή παρένθεση η οποία αφορά το σύνολο των δεδομένων στο οποίο αναφερόμαστε. Τα δεδομένα του συνόλου δεν είναι τα πρωτογενή δεδομένα του προβλήματος, αλλά έχουν μετατραπεί σε μορφή διανυσμάτων με ομοιογενείς τιμές. Έτσι, μαζί με το διάνυσμα των ετικετών συνιστούν ένα πίνακα- μητρώο διαστάσεων [P x (N+1] που ονομάζεται **πίνακας σχεδίασης (design matrix)**. Για παράδειγμα, τα χαρακτηριστικά μιας εικόνας είναι τριών διαστάσεων (μήκος x πλάτος x βάθος χρώματος) και με κατάλληλη μετατροπή, που ονομάζεται διανυσματοποίηση (vectorization) μπορούν να μετατραπούν σ διάνυσμα τιμών. Εάν το πρόβλημα αφορά για παράδειγμα μια ταξινόμηση εικόνων με ετικέτες σκύλος- γάτα, η μετατροπή θα είναι πχ σε 0, 1 αντίστοιχα. Δεν θα μπούμε σε περισσότερες λεπτομέρειες για τις μετατροπές, οι οποίες γίνονται πολύ απλά μέσω συναρτήσεων βιβλιοθηκών λογισμικού και δεν έχουν ιδιαίτερη σημασία στην παρούσα φάση για την κατανόηση των εννοιών. Για περισσότερες λεπτομέρειες, ο αναγνώστης παραπέμπεται ενδεικτικά στις πηγές [22], [23] και επιστρέφουμε στο πρόβλημα.

Στην απλούστερη περίπτωση, οι δύο τάξεις των δεδομένων διαχωρίζονται ιδανικά με μια γραμμή, ένα **γραμμικό όριο απόφασης (linear decision boundary)**, όπως φαίνεται στην Εικόνα 3-12, όπου με μπλε χρώμα είναι τα παραδείγματα όπου $y_p = 1$ και με κόκκινο χρώμα είναι τα παραδείγματα όπου $y_p = 0$.

[4] Η σύμβαση γραφής για πίνακες (arrays) είναι με έντονα και πλάγια γράμματα (πχ πίνακας $\boldsymbol{x}$), ενώ για βαθμωτά μεγέθη με πλάγια γράμματα (πχ μέγεθος *y*)



Εικόνα 3-12. Γραμμικό όριο απόφασης

Η γραμμή είναι ένα σημείο όταν η διάσταση της εισόδου είναι N=1 , μια γραμμή όταν N=2 και πιο γενικά για N ένα υπερ-επίπεδο που ορίζεται από το σύνολο των δεδομένων εισόδου. Ουσιαστικά, για το γραμμικό όριο απόφασης που είναι ένα υπερ επίπεδο, η εξίσωσή του δίνεται από τη σχέση:

$$w_0 + x_1 w_1 + \cdots + x_N w_N = 0 \qquad (3.3.1)$$

Ο όρος w_0 αντιπροσωπεύει την μετατόπιση ως προς την πηγή (0,0) του καρτεσιανού συστήματος συντεταγμένων και αναφέρεται συχνά ως **πόλωση (bias)** και συμβολίζεται συνήθως με *b*, αλλά προς το παρόν για απλούστευση κρατάμε το w_0.

Η εξίσωση μπορεί να γραφεί και με μορφή εσωτερικού γινομένου πινάκων ως:

$$\mathring{\boldsymbol{x}}^T\boldsymbol{w} = 0 \qquad (3.3.2)^5$$

θεωρώντας ως[6]:

[5] Για τον συμβολισμό του εσωτερικού γινομένου δύο πινάκων ***a***, **b** ακολουθείται η σύμβαση ***ab***. Ο ανάστροφος (transpose) ενός πίνακα ***a*** συμβολίζεται με $\boldsymbol{a}^T$.

[6] Η διαφορά του $\boldsymbol{x}$ σε σχέση με το $\dot{\boldsymbol{x}}$ είναι ότι έχει μπει ως στοιχείο το 1, που είναι ο συντελεστής του *x* για τον όρο w_0 της πόλωσης.

$$\boldsymbol{w} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_N \end{bmatrix} \kappa\alpha\iota\ \mathring{\boldsymbol{x}} = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} \tag{3.3.3}$$

Το διάνυσμα $\boldsymbol{w}$**,** είναι το διάνυσμα των **παραμέτρων,** των τιμών που καθορίζουν την συμπεριφορά του συστήματος. Το στοιχείο w_i είναι ο συντελεστής που πολλαπλασιάζεται με το χαρακτηριστικό x_ι στην εξίσωση 3.3.1. Το $\boldsymbol{w}$ μπορεί να θεωρηθεί ως ένα σύνολο **βαρών (weights)** που ορίζουν πως το κάθε χαρακτηριστικό επηρεάζει την πρόβλεψη y. Εάν ένα χαρακτηριστικό έχει θετικό βάρος, τότε αυξάνοντας την τιμή αυτού του χαρακτηριστικού επηρεάζει προς τη θετική κλάση. Εάν ένα χαρακτηριστικό έχει αρνητικό βάρος, τότε αυξάνοντας την τιμή αυτού του χαρακτηριστικού επηρεάζει προς τη αρνητική κλάση. Εάν ένα χαρακτηριστικό έχει μηδενικό βάρος, τότε δεν έχει καμιά επιρροή.

Στόχος του μοντέλου που θα επιλεγεί, είναι να μάθει το γραμμικό όριο απόφασης για να διαχωρίσει τις δύο κλάσεις. Έτσι, όπου $\mathring{\boldsymbol{x}}^T\boldsymbol{w} > \boldsymbol{0}$, θα αντιστοιχεί στην κλάση 1 (ή αλλιώς θετική κλάση) και όπου $\mathring{\boldsymbol{x}}^T\boldsymbol{w} < \boldsymbol{0}$, θα αντιστοιχεί στην κλάση 0 (ή αλλιώς αρνητική κλάση)

Οπότε, θεωρώντας την ιδανική περίπτωση έχουμε τη γνώση όλων των δυνατών πιθανών παραμέτρων το μοντέλο εκφράζεται από τη σχέση:

$$\text{model}(\boldsymbol{x}_p, \boldsymbol{w}) \approx y_p \tag{3.3.4}$$

Οι υπολογισμοί για το μοντέλο γίνονται με βάση την σχέση:

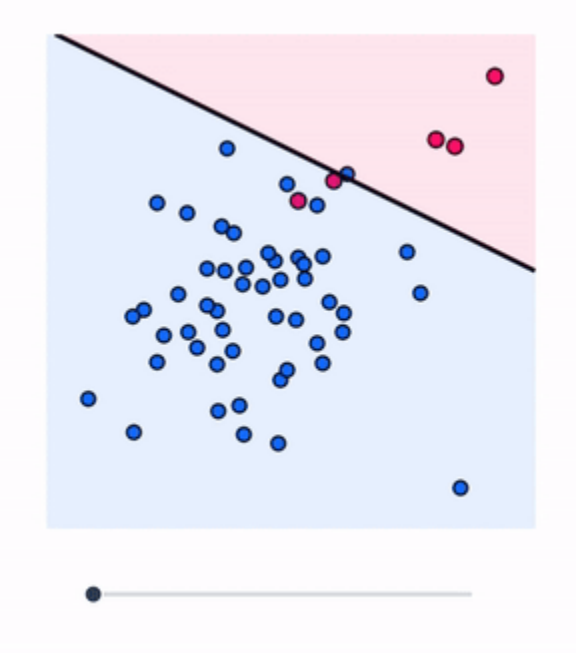
$$\text{model}(\boldsymbol{x}, \boldsymbol{w}) = w_0 + x_1 w_1 + \cdots + x_N w_N \tag{3.3.5}$$

Η σχέση 3.3.5 δεν είναι τίποτε άλλο, παρά η συνάρτηση $f(\boldsymbol{x})$ που καλείται να προσεγγίσει το μοντέλο με βάση τα δεδομένα μαθαίνοντας τις παραμέτρους **w.**

Θεωρώντας τα διανύσματα $\boldsymbol{w}$ και $\dot{\boldsymbol{x}}$ όπως στην 3.3.3 το μοντέλο με μορφή πινάκων είναι:

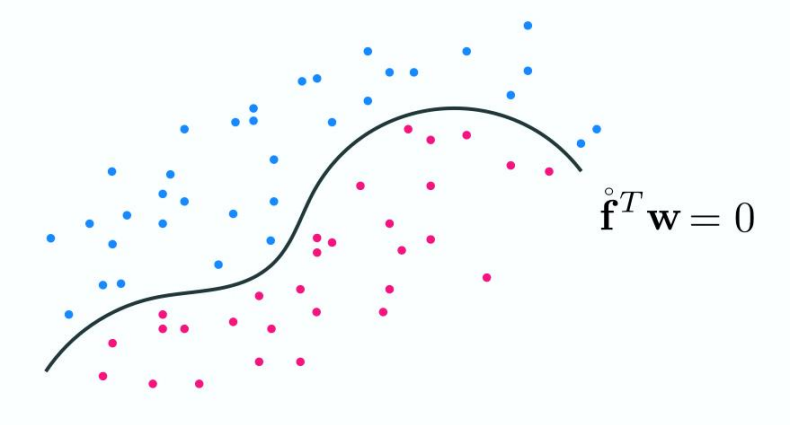
$$\text{model}(\boldsymbol{x}, \boldsymbol{w}) = \mathring{\boldsymbol{x}}^T\boldsymbol{w} \tag{3.3.6}$$

Στην Εικόνα 3-13 φαίνεται ένα οπτικοποιημένο παράδειγμα γραμμικού διαχωρισμού [31].



Εικόνα 3-13. Οπτικοποίηση γραμμικού διαχωρισμού

Όμως, στα περισσότερα προβλήματα του πραγματικού κόσμου, το όριο απόφασης σε ένα πρόβλημα δυαδικής ταξινόμησης δεν είναι γραμμικό, αλλά μη γραμμικό, όπως φαίνεται στην Εικόνα 3-14, οπότε χρειάζεται άλλο μοντέλο.



Εικόνα 3-14. Μη-γραμμικό όριο απόφασης

Το γραμμικό μοντέλο μπορεί να μετασχηματιστεί σε μη γραμμικό, με τη βοήθεια των μαθηματικών και συγκεκριμένα των μη γραμμικών συναρτήσεων που θα εφαρμόζονται σε κάθε είσοδο $\boldsymbol{x}$. Τέτοιες συναρτήσεις είναι, για παράδειγμα, η σιγμοειδής συνάρτηση και η συνάρτηση υπερβολικής εφαπτομένης. Στην ορολογία της ΜΜ μια τέτοια μη γραμμική συνάρτηση

ονομάζεται **μη γραμμικός μετασχηματισμός χαρακτηριστικού (nonlinear feature transformation)**, καθώς μετασχηματίζει τα χαρακτηριστικά της εισόδου $\boldsymbol{x}$.
Έτσι, το όριο απόφασης, όπως φαίνεται στην Εικόνα, παίρνει την αλγεβρική μορφή:

$$w_0 + f_1(\boldsymbol{x})w_1 + f_2(\boldsymbol{x})w_2 + \cdots + f_B(\boldsymbol{x})w_B = 0 \qquad (3.3.7)$$

Μπορούμε να αναπαραστήσουμε το σύνολο των παραμέτρων, δηλαδή τις εσωτερικές παραμέτρους της συνάρτησης f και τα βάρη τους στον γραμμικό μετασχηματισμό, μέσω ενός συνόλου $\boldsymbol{\theta}$. Για παράδειγμα, εάν η $f(x) = x$, τότε το μοντέλο είναι γραμμικό και το σύνολο των παραμέτρων είναι το σύνολο των βαρών του γραμμικού μετασχηματισμού. Όπως στην περίπτωση του γραμμικού μετασχηματισμού, που θεωρώντας την ιδανική περίπτωση έχουμε τη γνώση όλων των δυνατών πιθανών βαρών, έτσι και εδώ το μοντέλο θα είναι:

$$\text{model}(\boldsymbol{x}_p, \boldsymbol{\theta}) \approx y_p \qquad (3.3.8)$$

Μπορούμε γενικά να δημιουργήσουμε ένα μη γραμμικό μοντέλο που θα είναι το σταθμισμένο άθροισμα (weighted sum) B μη γραμμικών συναρτήσεων:

$$\text{model}(\boldsymbol{x}, \boldsymbol{\theta}) = w_0 + f_1(\boldsymbol{x})w_1 + f_2(\boldsymbol{x})w_2 + \cdots + f_B(\boldsymbol{x})w_B \qquad (3.3.9)$$

όπου $f_1, f_2, \ldots f_B$ είναι οι μετασχηματισμοί των χαρακτηριστικών και w_0 έως w_B και παριστάνονται από το σύνολο των παραμέτρων $\boldsymbol{\theta}$ που θα πρέπει να ρυθμιστούν κατάλληλα.
Σε αναλογία με τη γραμμική περίπτωση, μπορούμε να αναπαραστήσουμε το μοντέλο με μορφή πινάκων θέτοντας:

$$\boldsymbol{w} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_B \end{bmatrix} \quad \boldsymbol{f} = \begin{bmatrix} 1 \\ f_1(\boldsymbol{x}) \\ f_2(\boldsymbol{x}) \\ \vdots \\ f_B(\boldsymbol{x}) \end{bmatrix} \quad \boldsymbol{f}_p = \begin{bmatrix} 1 \\ f_1(\boldsymbol{x}_p) \\ f_2(\boldsymbol{x}_p) \\ \vdots \\ f_B(\boldsymbol{x}_p) \end{bmatrix} \qquad (3.3.10)$$

και να γράψουμε το μη γραμμικό μοντέλο ως:

$$\text{model}(\boldsymbol{x}, \theta) = \mathring{\boldsymbol{f}}^T \boldsymbol{w} \tag{3.3.11}$$

Το όριο απόφασης σε αυτή την περίπτωση για τις εισόδους $\boldsymbol{x}$ θα είναι $\mathring{\mathbf{f}}^T \mathbf{w} = 0$ και οι προβλέψεις δίνονται από τη σχέση (όπου sign = πρόσημο)

$$y = sign(\mathring{\boldsymbol{f}}^T \boldsymbol{w}) \tag{3.3.12}$$

3.3.2 Αποτίμηση Μοντέλου Δυαδικής Ταξινόμησης

Μετά την εκπαίδευση του μοντέλου δυαδικής ταξινόμησης, δηλαδή ενός **δυαδικού ταξινομητή (binary classifier)**, πρέπει να αποτιμήσουμε την ποιότητά του, ορίζοντας το μέτρο απόδοσης. Σε μια εργασία ταξινόμησης γενικά, ακολουθούνται οι εξής συμβάσεις για την ονοματολογία και τους συμβολισμούς για το σύνολο εκπαίδευσης και τις προβλέψεις:

- Ο κλάσεις είναι δύο: η ***θετική (Positive – P)*** και η ***αρνητική (Negative – N).***
- Ο αριθμός των παραδειγμάτων που ανήκουν στη *θετική* κλάση, ονομάζονται ***αληθώς θετικά (True Positives- TP)***, πχ υπάρχει ασθένεια.
- Ο αριθμός των παραδειγμάτων που ανήκουν στην αρνητική κλάση και έχουν προβλεφθεί σωστά, ονομάζονται ***αληθώς αρνητικά (True Negatives- TN)***, πχ δεν υπάρχει ασθένεια.
- Ο αριθμός προβλέψεων που ταξινομήθηκαν στη θετική κλάση ενώ στην πραγματικότητα ανήκουν στην αρνητική, ονομάζονται ***ψευδώς αρνητικά (False Positive- FP)***. Επίσης, ονομάζονται *λάθος τύπου I (Type I error).*
- Ο αριθμός προβλέψεων που ταξινομήθηκαν στην αρνητική κλάση ενώ στην πραγματικότητα ανήκουν στη θετική, ονομάζονται ***ψευδώς αρνητικά (False Negatives - FN)***. Επίσης, ονομάζονται *λάθος τύπου II (Type II error).*

3.3.2.1 Μετρικές Εκτίμησης Απόδοσης

Η **ορθότητα (accuracy)** είναι ένα συνηθισμένο μέτρο απόδοσης του μοντέλου. Απλά, η ορθότητα είναι η αναλογία των παραδειγμάτων για τα οποία έχει γίνει η σωστή πρόβλεψη επί του συνόλου των προβλέψεων και με βάση τις παραπάνω συμβάσεις και ισοδυναμεί με τον δείκτη λάθους (error rate) του μοντέλου [1]. Η ορθότητα δίνεται από τον τύπο:

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

Η ορθότητα ως μέτρο απόδοσης έχει μια διαισθητική απλή εξήγηση: η αναλογία των παραδειγμάτων για τα οποία έγινε σωστή πρόβλεψη. Όμως, σε προβλήματα του πραγματικού

κόσμου οι κλάσεις των δεδομένων στα σύνολα δεν κατανέμονται με ισορροπία. Για παράδειγμα, σε ένα σύνολο δεδομένων που αφορούν μια πολύ σπάνια περίπτωση καρκίνου μπορεί το 99.99% των παραδειγμάτων να ανήκουν στην αρνητική κλάση και μόνο το 0.01% στη θετική. Σε τέτοια περίπτωση, μπορεί να έχουμε ένα μοντέλο με υψηλή ορθότητα, αλλά το μοντέλο δεν έχει προγνωστική δύναμη. Έτσι, για το παράδειγμά μας, μπορεί στην εκπαίδευση το μοντέλο να έχει ορθότητα 95%. Όμως, το 99.99% των ανθρώπων δεν έχουν την ασθένεια. Εάν απλά δημιουργήσουμε ένα μοντέλο που «προβλέπει» ότι σχεδόν κανένας δεν έχει την ασθένεια, το απλοϊκό μοντέλο θα μπορούσε να είναι κατά 4.99% πιο ορθό, αλλά ξεκάθαρα δεν μπορεί να προβλέψει το καθετί. Να τονίσουμε ότι στο κόσμο των ιατρικών, είναι σημαντικότερο να υπάρχουν λίγα (FN) δείγματα. Αυτό, γιατί είναι προτιμότερο να γνωρίζουμε ότι κάποιος έχει μια ασθένεια, ενώ δεν έχει (FP) δείγματα παρά ότι δεν την έχει ενώ την έχει (FN) δείγματα. Συνεπώς, η ορθότητα ως μετρική απόδοσης δεν αρκεί. Για τον σκοπό αυτό, στην ταξινόμηση χρησιμοποιούμε και άλλες μετρικές: την ακρίβεια (precision)[7], την ανάκληση (recall), το $F1$ αποτέλεσμα ($F1$ score) και τον δείκτη ψευδώς αρνητικών (False Positive Rate- FPR) [1], [22].

Η **ακρίβεια (*precision*),** γνωστή και ως **θετική προγνωστική αξία (Positive Predictive Value-PPV)**, είναι η αναλογία του κάθε παραδείγματος που προβλέπεται ως θετικό και είναι πραγματικά θετικό. Μπορεί να θεωρηθεί ως μέτρο θορύβου στις προβλέψεις, δηλαδή όταν προβλέπουμε ότι κάτι είναι θετικό, πόσο πιθανό είναι να είμαστε σωστοί [22]. Η ακρίβεια δίνεται από τον τύπο:

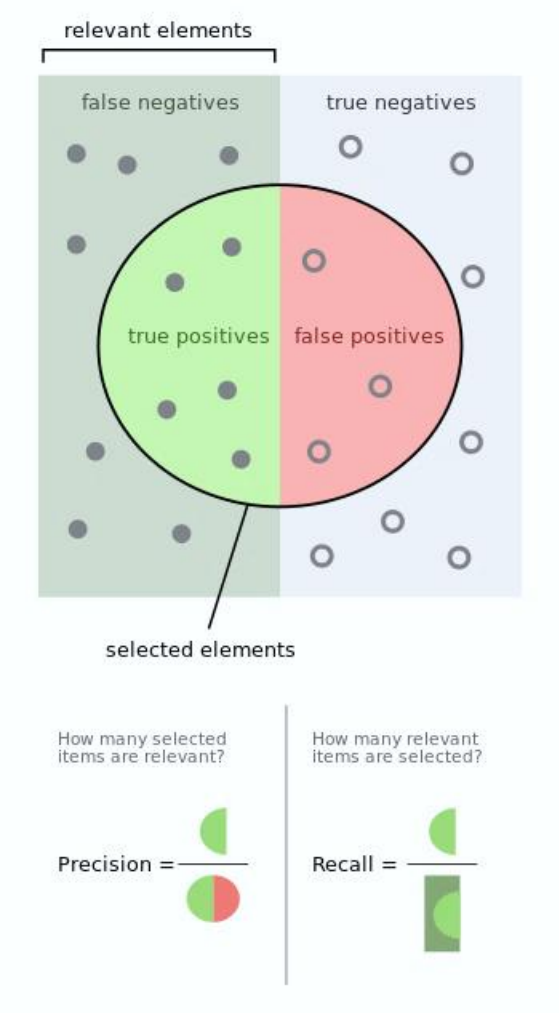
$$Precision = \frac{TP}{FP + TP}$$

Η **ανάκληση (*recall*),** γνωστή και ως **ευαισθησία (sensitivity)** καθώς και ως **δείκτης αληθώς θετικών (True Positive Rate – TPR)**, είναι η αναλογία κάθε θετικού παραδείγματος που είναι πραγματικά θετικό [22]. Η ανάκληση μετρά την ικανότητα του μοντέλου να αναγνωρίζει ένα παράδειγμα της θετικής κλάσης και δίνεται από τον τύπο:

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

[7] Στην στατιστική οι όροι accuracy και precision, οι οποίοι στα ελληνικά αποδίδονται με τη λέξη ακρίβεια, έχουν διαφορετική έννοια. Οπότε για τον διαχωρισμό ερμηνεύουμε τον όρο accuracy ως ορθότητα και τον όρο precision ως ακρίβεια.

Κοινώς, η ακρίβεια είναι το κλάσμα των προβλέψεων που αναφέρθηκαν από το μοντέλο ως σωστές και η ανάκληση είναι το κλάσμα των πραγματικών παραδειγμάτων που έχουν προβλεφθεί. Λόγω του ότι η διαίσθηση για την ακρίβεια και την ανάκληση δεν είναι τόσο προφανής όσο αυτή της ορθότητας, παραθέτουμε την επεξηγηματική Εικόνα 3-15.



Εικόνα 3-15. Precision και Recall
Πηγή: https://en.wikipedia.org/wiki/Precision_and_recall

Σε πολλές περιπτώσεις, εάν θέλουμε να συνοψίσουμε την απόδοση του μοντέλου, επιδιώκοντας ένα είδος ισορροπίας μεταξύ της ακρίβειας και την ανάκλησης, χρησιμοποιούμε μια άλλη μετρική που συνδυάζει τις δύο μετρικές, τον αρμονικό μέσο όρο[8] της ακρίβειας και της ανάκλησης που ονομάζεται ***F1* αποτέλεσμα (*F1* score)** [1]**.** Το $F1$ αποτέλεσμα, ή απλά $F1$, δίνεται από τον τύπο:

[8] Γενικότερα ο αρμονικός μέσος όρος στην περίπτωση δύο αριθμών, συμπίπτει με το τετράγωνο του γεωμετρικού μέσου διαιρούμενο με τον αριθμητικό μέσο.

$$F1 = \frac{2}{\frac{1}{Precision} + \frac{1}{Recall}} = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{TP}{TP + \frac{FN + FP}{2}}$$

Η προσδιοριστικότητα (specificity), γνωστή και ως **δείκτης αληθώς αρνητικών (True Negative Rate – TNR)** μετρά τις ορθά αρνητικές προβλέψεις στο σύνολο των ορθών αρνητικών δειγμάτων και δίνεται από τον τύπο:

$$Specificity = TNR = \frac{TN}{TN + FP}$$

Η μετρική του **δείκτη ψευδώς αρνητικών (False Positive Rate- *FPR*)** αντιστοιχεί στην αναλογία των αρνητικών δειγμάτων που θεωρήθηκαν ως θετικά, σε σχέση με όλα τα αρνητικά παραδείγματα. Με άλλα λόγια, όσο μεγαλύτερος ο *FPR*, πόσα περισσότερα αρνητικά δείγματα έχουν ταξινομηθεί λάθος [22]. Ο *FPR* υπολογίζεται από τον τύπο:

$$FPR = \frac{FP}{FP + TN} = 1 - specificity = 1 - TNR$$

3.3.2.2 Καμπύλες Εκτίμησης Απόδοσης

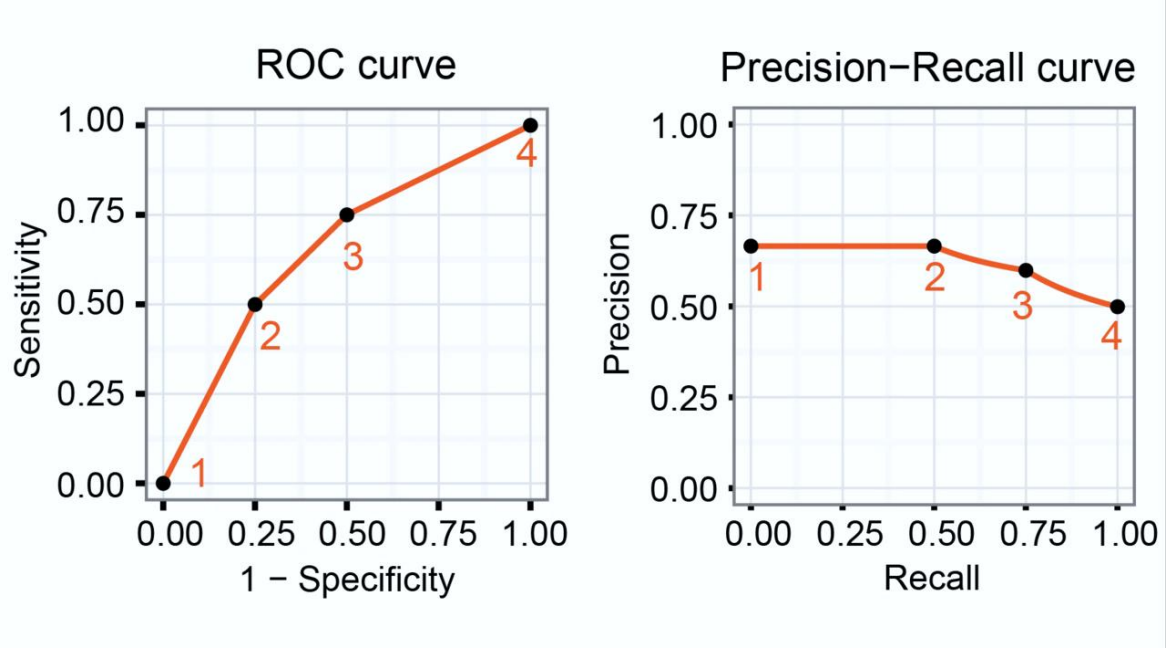
Όταν χρησιμοποιούνται ως μετρικές εκτίμησης απόδοσης του δυαδικού ταξινομητή η ακρίβεια και η ανάκληση, συνηθίζεται να απεικονίζονται γραφικά με την **PR καμπύλη (PR curve)** με την ακρίβεια να αντιστοιχεί στον άξονα των y και την ανάκληση στον άξονα των x [1].

Ένας άλλος, εύκολος σχετικά γραφικός τρόπος και ο πιο συνηθισμένος για την εκτίμηση της απόδοσης ενός ταξινομητή είναι η **χαρακτηριστική καμπύλη λειτουργίας δέκτη (Receiving Operating Characteristic curve – ROC curve).** Η ROC καμπύλη είναι η σχεδίαση της μετρικής *TPR*, δηλαδή της ανάκλησης, σε σχέση με την μετρική *FPR* και αναπαριστά τα *TP* και τα *FP* για

κάθε κατώφλι (threshold) [9], δηλαδή την πιθανότητα στην οποία ένα δείγμα προβλέπεται να ανήκει σε μια κλάση [22]. Πιο συγκεκριμένα:

- Μικρότερες τιμές στον άξονα των x δείχνουν χαμηλότερα FP και οι υψηλότερα TN.
- Μεγαλύτερες τιμές στον y άξονα δείχνουν υψηλά TP και οι μικρότερες χαμηλότερα FN.

Στην Εικόνα 3-16 φαίνονται παραδείγματα μιας καμπύλης ROC και της PR καμπύλης ενός δυαδικού ταξινομητή.



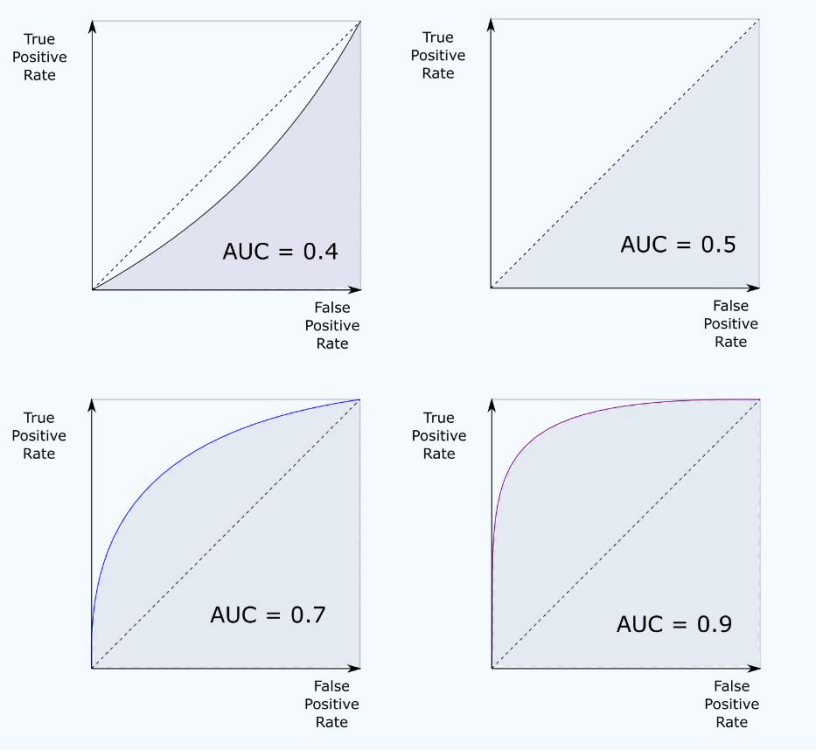
Εικόνα 3-16. Καμπύλες ROC και PR
Πηγή: https://www.linkedin.com/pulse/lets-evaluate-classification-model-roc-pr-curves-suravi-mahanta/

Επιπρόσθετα, η ROC καμπύλη χρησιμοποιείται ως μια γενική μετρική του μοντέλου. Όσο καλύτερο είναι ένα μοντέλο, τόσο υψηλότερα είναι η καμπύλη και συνεπώς τόσο μεγαλύτερη η επιφάνεια κάτω από την καμπύλη. Για αυτόν τον λόγο, συναντάται συχνά ο υπολογισμός της **επιφάνειας κάτω από την ROC καμπύλη (Area Under the Curve – AUC)**. Ο τέλειος

[9] Οι αλγόριθμοι ΜΜ για την ταξινόμηση εκτιμούν την πιθανότητα ένα παράδειγμα να ανήκει σε μια κλάση Για παράδειγμα, εάν στο μοντέλο έχει οριστεί ως κατώφλι το 0.5, τότε το μοντέλο θα ταξινομήσει ένα παράδειγμα στη θετική κλάση εάν υπολογίσει πιθανότητα >50%.

ταξινομητής έχει ROC AUC ίση με 1, ενώ ένας κακός ταξινομητής θα έχει 0.5, ίσως και λιγότερο. Συνεπώς, όσο πιο κοντά η ROC AUC στο 1, τόσο καλύτερος είναι ο ταξινομητής [22].

Στην Εικόνα 3.17 φαίνονται παραδείγματα για διάφορες τιμές της ROC AUC, όπου η διακεκομμένη γραμμή με κλίση 45^0 φαίνεται σε όλα τα γραφήματα για σύγκριση και απεικονίζει μοντέλο με AOC = 0.5.



Εικόνα 3-17. Παραδείγματα AUC
Πηγή: https://www.linkedin.com/pulse/lets-evaluate-classification-model-roc-pr-curves-suravi-mahanta/

3.3.2.3 Πίνακας Ταξινόμησης

Στην ΜΜ και ιδιαίτερα στην εργασία της ταξινόμησης όπου χρησιμοποιείται η στατιστική, ο **πίνακας σύγχυσης (confusion matrix)** - σε καλύτερη απόδοση **πίνακας ταξινόμησης** – είναι ένας δισδιάστατος πίνακας ο οποίος παρέχει τη δυνατότητα οπτικοποίησης της απόδοσης ενός μοντέλου. Στην περίπτωση της δυαδικής ταξινόμησης είναι ένα μητρώο 2x2, όπως δίνεται στον Πίνακα 3-1 και κάθε γραμμή παριστάνει τα παραδείγματα της κλάσης που έχει προβλέψει ο ταξινομητής (κλάση πρόγνωσης), ενώ κάθε στήλη παριστάνει τα παραδείγματα της κλάσης που ανήκουν πραγματικά τα παραδείγματα (πραγματική κλάση) [22].

		Πραγματική κλάση	
		P	**N**
Κλάση πρόγνωσης	**P**	**TP**	FP
	N	FN	**TN**

Πίνακας 3-1. Η γενική μορφή του πίνακα ταξινόμησης

Ο όρος σύγχυση προκύπτει από το γεγονός από το ότι μπορούμε να δούμε εύκολα εάν το μοντέλο συγχέει τις δύο κλάσεις, δηλαδή βάζει διαφορετική ετικέτα από αυτή που αντιστοιχεί πραγματικά. Ο πίνακας ταξινόμησης συνήθως οπτικοποιείται με τη μορφή heatmap[10] και παρέχει τη δυνατότητα να δούμε όχι μόνο εάν το μοντέλο είναι λάθος, αλλά και τι πήγε λάθος.

Στον Πίνακα 3-2 δίνονται οι μετρικές εκτίμησης της απόδοσης, σε συνδυασμό με τον δυαδικό πίνακα ταξινόμησης

		Πραγματική Κλάση		
	Συνολικός πληθυσμός	Κλάση θετική	Κλάση αρνητική	Accuracy $\frac{TP+TN}{TP+TN+FP+FN}$
Κλάση πρόγνωσης	Κλάση πρόγνωσης θετική	**True positive**	**False positive**, Type I error	Positive predictive value (PPV), Precision $\frac{TP}{FP+TP}$
	Κλάση πρόγνωσης αρνητική	**False negative**, Type II error	**True negative**	
		True positive rate (TPR), Recall, Sensitivity $\frac{TP}{TP+FN}$	False positive rate (FPR) $\frac{FP}{FP+TN}$	F1 score = 2 * Precision * Recall / Precision + Recall
		False negative rate (FNR) $1-TPR$	Specificity True negative rate (TNR)	

[10] Η μορφή heatmap (χάρτης θερμότητας) είναι τεχνική οπτικοποίησης των δεδομένων που απεικονίζει το μέγεθος των δεδομένων ως χρώμα σε δύο διαστάσεις.

	$\frac{TN}{TN + FP}$	

Πίνακας 3-2. Ο δυαδικός πίνακας ταξινόμησης και οι μετρικές εκτίμησης απόδοσης

4 Τεχνητά Νευρωνικά Δίκτυα

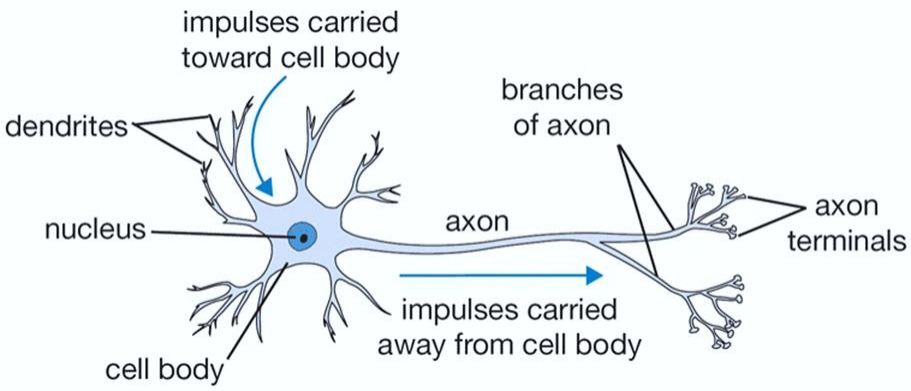
Στο κεφάλαιο αυτό παρουσιάζονται τα βασικά στοιχεία των ΤΝΔ. Αρχικά παρουσιάζονται ο βιολογικός νευρώνας και ο τεχνητός νευρώνας, καθώς και τα είδη των ΤΝΔ. Στη συνέχεια, αναλύεται αρχιτεκτονική των MPLs και η διαδικασία της εκπαίδευσής τους. Κατόπιν παρουσιάζονται οι κυριότερες συναρτήσεις ενεργοποίησης, οι συνηθισμένες συναρτήσεις κόστους, οι αλγόριθμοι βελτιστοποίησης και οι συχνότερες μέθοδοι εξομάλυνσης. Τέλος, ανακεφαλαιώνεται συνοπτικά η διαδικασία εκπαίδευσης.

4.1 Ο Βιολογικός και ο Τεχνητός Νευρώνας

Τα ΤΝΔ αποτελούν μια κατηγορία αλγορίθμων ΜΜ εμπνευσμένη από τη δομή του βιολογικού νευρικού συστήματος του ανθρώπινου εγκεφάλου. Η θεμελιώδης βασική μονάδα του εγκεφάλου είναι ο νευρώνας και κατά αντιστοιχία του ΤΝΔ ο τεχνητός νευρώνας.

4.1.1 Ο Βιολογικός Νευρώνας

Η επεξεργασία πληροφοριών που εκτελείται από τον ανθρώπινο εγκέφαλο πραγματοποιείται από βιολογικές συνιστώσες επεξεργασίας, που λειτουργούν παράλληλα για την παραγωγή κατάλληλων λειτουργιών όπως η σκέψη και η μάθηση. Το θεμελιώδες κύτταρο του κεντρικού νευρικού συστήματος είναι ο **νευρώνας (neuron)** και ο ρόλος του είναι να παράγει παλμούς κάτω από ορισμένες συνθήκες. Ο βιολογικός νευρώνας φαίνεται στην Εικόνα 4-1.



Εικόνα 4-1. Ο βιολογικός νευρώνας.

Πηγή: L. Fridman, “Deep Learning Basics.” [Online]. Available: https://www.dropbox.com/s/c0g3sc1shi63x3q/deep_learning_basics.pdf?dl=0 [Accessed: 30-Sep-2020]

Αποτελείται από το **κυτταρικό σώμα (cell body)** που περιέχει τον **πυρήνα (nucleus)**, διακλαδώσεις που ονομάζονται **δενδρίτες (dendrites)** και μια μακρύτερη προέκταση, τον **άξονα (axon)**. Στο άκρο του ο άξονας έχει διακλαδώσεις και στην κορυφή των διακλαδώσεων Κάθε νευρώνας δέχεται σήματα εισόδου από τους δενδρίτες και παράγει σήματα εξόδου διαμέσου του άξονα. Στο άκρο του άξονα, υπάρχουν διακλαδώσεις (branches) και στο τέρμα (terminal) της κάθε διακλάδωσης υπάρχουν μικροσκοπικές δομές, που ονομάζονται **συνάψεις (synapses)**. Οι συνάψεις συνδέονται με τους δενδρίτες ή τα κυτταρικά σώματα άλλων νευρώνων. Οι νευρώνες δέχονται μικρούς ηλεκτρικούς παλμούς, τα λεγόμενα σήματα, από τους από τους άλλους νευρώνες μέσω των συνάψεων. Οι δενδρίτες μεταφέρουν αυτά τα σήματα προς το κυτταρικό σώμα. Όταν το άθροισμα των σημάτων ξεπεράσει ένα κατώφλι ηλεκτρικού δυναμικού, το λεγόμενο δυναμικό ενεργοποίησης, ο νευρώνας πυροδοτείται και στέλνει τα δικό του σήμα προς τους άλλους νευρώνες. Το σήμα που εισέρχεται στον άλλο νευρώνα διαμορφώνεται κατά ένα ποσοστό και ονομάζεται συναπτικό δυναμικό, το οποίο μπορεί να είναι ενισχυτικό (excitory) ή κατασταλτικό (inhibitory) ως προς το σήμα εξόδου [22], [27] .

Στον ανθρώπινο εγκέφαλο υπάρχουν περίπου 100 εκατομμύρια νευρώνες που συνδέονται μεταξύ τους με περίπου 1000 τρισεκατομμύρια συνάψεις, σχηματίζοντας έτσι, ένα τεράστιο και πολυσύνθετο δίκτυο, το βιολογικό νευρικό δίκτυο, ικανό να εκτελεί τις πολύπλοκες μαθησιακές λειτουργίες. Η γνώση «αποθηκεύεται» στις τιμές των συναπτικών δυναμικών και η μάθηση στο βιολογικό σύστημα είναι η μεταβολή των συναπτικών δυναμικών [27]. Η αρχιτεκτονική του βιολογικού νευρωνικού δικτύου αποτελεί αντικείμενο της νευροεπιστήμης και ερευνάται συνεχώς.

4.1.2 Αντιστοιχία Βιολογικού και Τεχνητού Νευρώνα

Στο απλουστευμένο υπολογιστικό μοντέλο του **τεχνητού νευρώνα (artificial neuron),** όπως περιγράφεται στην [27], το οποίο φαίνεται στην Εικόνα 4-2, κατά αντιστοιχία με τον βιολογικό νευρώνα, τα σήματα, όπως το x_0, που μεταδίδονται διαμέσου των αξόνων αλληλοεπιδρούν πολλαπλασιαστικά, $w_0 x_0$, με τους δενδρίτες των άλλων νευρώνων με βάση το συναπτικό δυναμικό που αντιστοιχεί στο **βάρος (weight)**, w_0. Η βασική ιδέα είναι ότι, όπως τα συναπτικά δυναμικά στον βιολογικό νευρώνα, έτσι και τα βάρη προσαρμόζονται και ελέγχουν τη δύναμη της επιρροής ενός νευρώνα με τον άλλον. Έτσι, η ενίσχυση αντιστοιχεί σε θετικό βάρος και καταστολή σε αρνητικό βάρος. Όπως στον βιολογικό νευρώνα τα σήματα που μεταφέρονται στο

κυτταρικό σώμα αθροίζονται και αν το άθροισμα ξεπεράσει το δυναμικό ενεργοποίησης ο νευρώνας πυροδοτείται και στέλνει το σήμα στον άξονα, κατά αναλογία στον τεχνητό νευρώνα θεωρούμε ότι έχουμε το άθροισμα $z = \sum_\iota w_\iota x_\iota + b$, το οποίο ονομάζεται **σταθμισμένο άθροισμα (weighted sum)** και η πυροδότηση που μεταφέρει την πληροφορία μοντελοποιείται με τη συνάρτηση ενεργοποίησης f. Έτσι, η έξοδος του τεχνητού νευρώνα y κατά αντιστοιχία με την έξοδο του άξονα του βιολογικού δίνεται από τη σχέση:

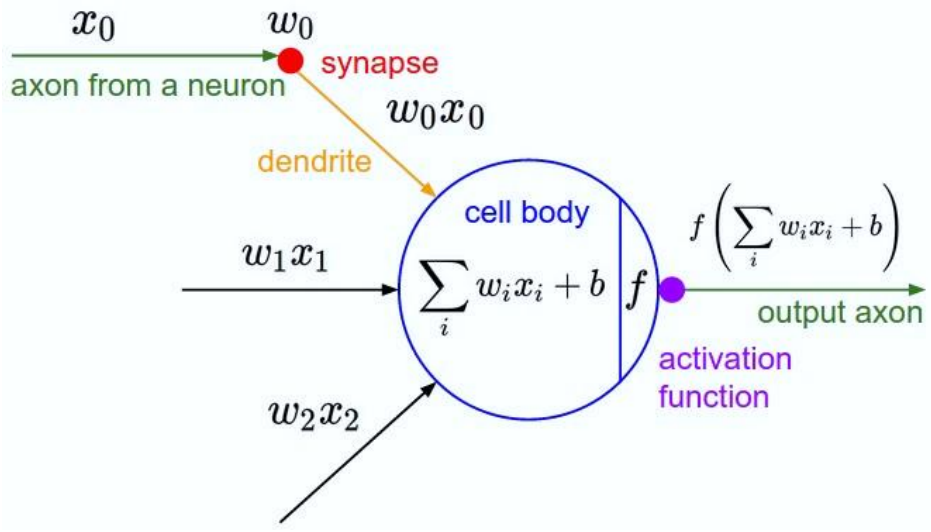
$$y = f(\sum_\iota w_\iota x_\iota + b) \qquad (4.1.1)$$

Ο όρος b, είναι η **πόλωση (bias)** και τίθεται προαιρετικά στο άθροισμα, όταν είναι επιθυμητή η μετατόπιση του συνολικού σήματος ενεργοποίησης κατά μία σταθερή τιμή (συνήθως b=1).
Σε μορφή διανυσμάτων, ο όρος $\sum_\iota w_\iota x_\iota + b$ εκφράζεται ως

$$\boldsymbol{z} = \boldsymbol{x}^T \boldsymbol{w} + b \qquad (4.1.2)$$

και η έξοδος ως

$$y = f(\boldsymbol{z}) \qquad (4.1.3)$$



Εικόνα 4-2. Το μοντέλο του τεχνητού νευρώνα.

Πηγή: L. Fridman, "Deep Learning Basics." [Online]. Available: https://www.dropbox.com/s/c0g3sc1shi63x3q/deep_learning_basics.pdf?dl=0 [Accessed: 30-Sep-2020]

Το παραπάνω μοντέλο του τεχνητού νευρώνα, είναι η πιο απλή μορφή ΤΝΔ, και αναφέρεται συχνά ως **perceptron ή λογική μονάδα κατωφλίου (threshold logic unit)** ή απλά **μονάδα (unit)**, μιας και βασίζεται στο αρχικό μοντέλο του Rosenblatt και περιγράφεται συνοπτικά στην επόμενη παράγραφο.

Συνοψίζοντας γενικά τα παραπάνω σε μαθηματική μορφή, έχουμε ότι:

- Η μονάδα είναι μια συνάρτηση που δέχεται ως είσοδο ένα διάνυσμα $\boldsymbol{x} \in \mathbb{R}^n$
- Η μονάδα παραμετροποιείται από ένα διάνυσμα βαρών $\boldsymbol{w} \in \mathbb{R}^n$ και την πόλωση b.
- Η έξοδος y της μονάδας είναι:
  $f(\sum_{i=1}^{n} x_i \cdot w_i + b)$ ή $αλλιώς\ f(\boldsymbol{x^T w} + b) = f(\boldsymbol{z})$
  όπου $f: \mathbb{R} \to \mathbb{R}$ είναι η συνάρτηση ενεργοποίησης
- Υπάρχουν διάφορες συναρτήσεις ενεργοποίησης f που εφαρμόζονται στο σταθμισμένο άθροισμα για να παραχθεί η έξοδος (θα παρουσιαστούν σε επόμενη παράγραφο).

Ως συμπέρασμα από τα παραπάνω προκύπτει ότι:

- Ο βιολογικός νευρώνας είναι η υπολογιστική μονάδα για τον εγκέφαλο
- Ο τεχνητός νευρώνας είναι η υπολογιστική μονάδα για το νευρωνικό δίκτυο

Οι βασικές διαφορές μεταξύ του ανθρώπινου εγκέφαλου και των ΤΝΔ δίνονται στον Πίνακα 4-1 [27].

Διαφορές	**Επεξήγηση**
Παράμετροι	Ο ανθρώπινος εγκέφαλος έχει περίπου 10.000.000 περισσότερες συνάψεις από τα νευρωνικά δίκτυα
Τοπολογία	Ο ανθρώπινος εγκέφαλος δεν έχει «επίπεδα» και δουλεύει ασύγχρονα. Τα ΤΝΔ έχουν επίπεδα και δουλεύουν σύγχρονα
Αλγόριθμος εκμάθησης	Τα ΤΝΔ χρησιμοποιούν συγκεκριμένους αλγορίθμους για να μαθαίνουν. Δεν ξέρουμε ακόμη τι αλγόριθμους χρησιμοποιεί ο ανθρώπινος εγκέφαλος
Κατανάλωση ενέργειας	Τα βιολογικά νευρικά δίκτυα καταναλώνουν πάρα πολύ μικρή ενέργεια σε σχέση με τα ΤΝΔ

Στάδια	Τα βιολογικά δίκτυα συνήθως δεν σταματούν ποτέ να μαθαίνουν. Τα ΤΝΔ πρώτα εκπαιδεύονται και μετά δοκιμάζονται
Συνάψεις	Ο ανθρώπινος εγκέφαλος έχει 1000 τρισεκατομμύρια συνάψεις. Ένα από τα μεγαλύτερα ΤΝΔ, το ResNet 152, έχει 60 εκατομμύρια συνάψεις

Πίνακας 4-1. Οι κύριες διαφορές μεταξύ του ανθρώπινου εγκέφαλου και των ΤΝΔ

4.1.3 Το απλό Perceptron

Στο πιο απλό ΤΝΔ, στο perceptron του Rosenblatt η συνάρτηση ενεργοποίησης είναι μια γραμμική συνάρτηση, όπως η βηματική συνάρτηση. Το απλό perceptron μπορεί να χρησιμοποιηθεί για ένα πρόβλημα δυαδικής ταξινόμησης, λύνοντας το πρόβλημα με γραμμικές εξισώσεις: υπολογίζεται το σταθμισμένο άθροισμα των εισόδων του από το σταθμισμένο άθροισμα $z = \sum_j w_j x_j + b$, και η έξοδος λαμβάνεται:

$$y = \begin{cases} 0 & \text{εάν } \sum_j w_j x_j + b \leq \text{ κατώφλι} \\ 1 & \text{εάν } \sum_j w_j x_j + b > \text{ κατώφλι} \end{cases} \qquad (4.1.4)$$

Η εκπαίδευση του perceptron αφορά την προσαρμογή των βαρών επαναληπτικά, έτσι ώστε σε περίπτωση ενός παραδείγματος με χαρακτηριστικά εισόδου $\boldsymbol{x}$ να υπολογιστεί στην έξοδο σωστά η τιμή στόχος *y*, η οποία λαμβάνει αποκλειστικά τις τιμές 0 ή 1 [22].

Όμως, το μειονέκτημα της βηματικής συνάρτησης είναι ότι δεν μπορεί να ανταποκριθεί σε μικρές αναπροσαρμογές των βαρών. Για να αντιμετωπιστεί αυτό το πρόβλημα, στο σταθμισμένο άθροισμα εφαρμόζεται μια άλλη συνάρτηση ενεργοποίησης, η σιγμοειδής συνάρτηση (περιγράφεται αναλυτικά σε επόμενη παράγραφο), και η έξοδος υπολογίζεται από τη σχέση:

$$y = \frac{1}{1 + e^{(-\sum_j w_j x_j - b)}} \qquad (4.1.5)$$

Όταν στο σταθμισμένο άθροισμα εφαρμόζεται η σιγμοειδής συνάρτηση, τότε το perceptron ονομάζεται σιγμοειδές perceptron. Ουσιαστικά, το σιγμοειδές perceptron δεν διαφέρει από το

απλό perceptron όταν έχουμε έναν τεχνητό νευρώνα [26]. Η διαφορά είναι σημαντική στα σύνθετα ΤΝΔ, τα Πολυεπίπεδα perceptron, όπως θα δούμε παρακάτω.

Προκειμένου να γίνει η αναπαράσταση της σύνδεσης της εξόδου με τις εισόδους, συνηθίζεται οι είσοδοι να αναπαριστάνονται ως νευρώνες και ονομάζονται νευρώνες εισόδου και αυτοί οι νευρώνες σχηματίζουν το **επίπεδο εισόδου (input layer)**. Όταν υπάρχει ο παράγοντας της πόλωσης, τότε στο επίπεδο προστίθεται ακόμη ένας νευρώνας, ο νευρώνας πόλωσης. Η λογική μονάδα του perceptron ονομάζεται **επίπεδο εξόδου (output layer)**.

Το σιγμοειδές perceptron είναι ένας **δυαδικός ταξινομητής (binary classifier)** και λειτουργεί όπως ο αλγόριθμος της λογιστικής παλινδρόμησης (logistic regression) και μπορεί να εκπαιδευτεί για σύνολα δεδομένων όπου μεταξύ των δεδομένων μπορεί να βρεθεί γραμμικό όριο απόφασης, αλλά όχι για την εύρεση μη-γραμμικού ορίου απόφασης, όπως είδαμε στην παράγραφο 3.3.1.

Προκειμένου να επιτευχθεί ο μη γραμμικός διαχωρισμός, η ιδέα ήταν να χρησιμοποιηθεί ένα ενδιάμεσο επίπεδο νευρώνων ανάμεσα στο επίπεδο εισόδου και εξόδου, ένα **κρυφό επίπεδο (hidden layer)** όπως ονομάζεται, όπου να μετασχηματίζονται οι γραμμικές είσοδοι κατάλληλα. Αυτή ήταν η αρχή για τη δημιουργία πολυεπίπεδων νευρωνικών δικτύων, που παρουσιάζονται στην επόμενη παράγραφο.

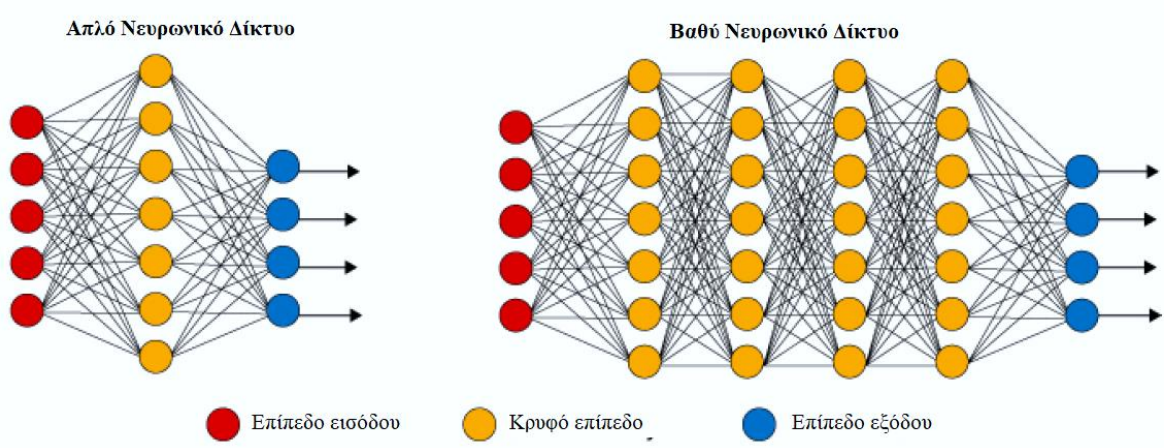
4.2 Είδη Τεχνητών Νευρωνικών Δικτύων

Τα Τεχνητά Νευρωνικά Δίκτυα (ΤΝΔ) προκύπτουν από τη διασύνδεση τεχνητών νευρώνων σε διάφορες τοπολογίες οργανωμένες σε **επίπεδα (layers)**. Τα **εμπρόσθιας τροφοδότησης νευρωνικά δίκτυα (feedforword neural networks)**, είναι η πεμπτουσία της ΒΜ. Ο στόχος ενός ΤΝΔ εμπρόσθιας τροφοδότησης είναι η προσέγγιση μιας συνάρτησης f^*. Η κύρια εφαρμογή τους είναι για εργασίες εποπτευόμενης μάθησης, όπως αυτή της ταξινόμησης. Για παράδειγμα, για έναν ταξινομητή $y= f^*(\boldsymbol{x})$, αντιστοιχίζεται μια είσοδος $\boldsymbol{x}$ σε μια κατηγορία y. Το ΤΝΔ ορίζει μια αντιστοίχιση $y = f(\boldsymbol{x},\boldsymbol{\theta})$ και μαθαίνει την τιμή των παραμέτρων $\boldsymbol{\theta}$ (βάρη και πολώσεις) που έχει ως αποτέλεσμα την καλύτερη προσέγγιση της συνάρτησης.

Ονομάζονται εμπρόσθιας τροφοδότησης γιατί η πληροφορία ρέει μόνο προς τα εμπρός μέσω της συνάρτησης που έχει αποτιμηθεί από το $\boldsymbol{x}$, μέσω των ενδιάμεσων υπολογισμών που εκτελούνται προς την έξοδο y [1]. Δεν υπάρχουν ανατροφοδούμενες συνδέσεις όπου οι έξοδοι του μοντέλου τροφοδοτούν προς τα πίσω τον εαυτό τους. Όταν υπάρχουν επεκτάσεις έτσι ώστε να

περιλαμβάνονται ανατροφοδοτούμενες συνδέσεις, τότε τα ΤΝΔ ονομάζονται ανατροφοδοτούμενα νευρωνικά δίκτυα (Recurrent Neural Networks – RNNs). Τα ΤΝΔ εμπρόσθιας τροφοδότησης είναι δύο ειδών: τα πολύ επίπεδα perceptron (MultiLayer Perceptrons - MPLs) και τα συνελικτικά νευρωνικά δίκτυα (Convolutional Neural Networks – CNNs) [1].

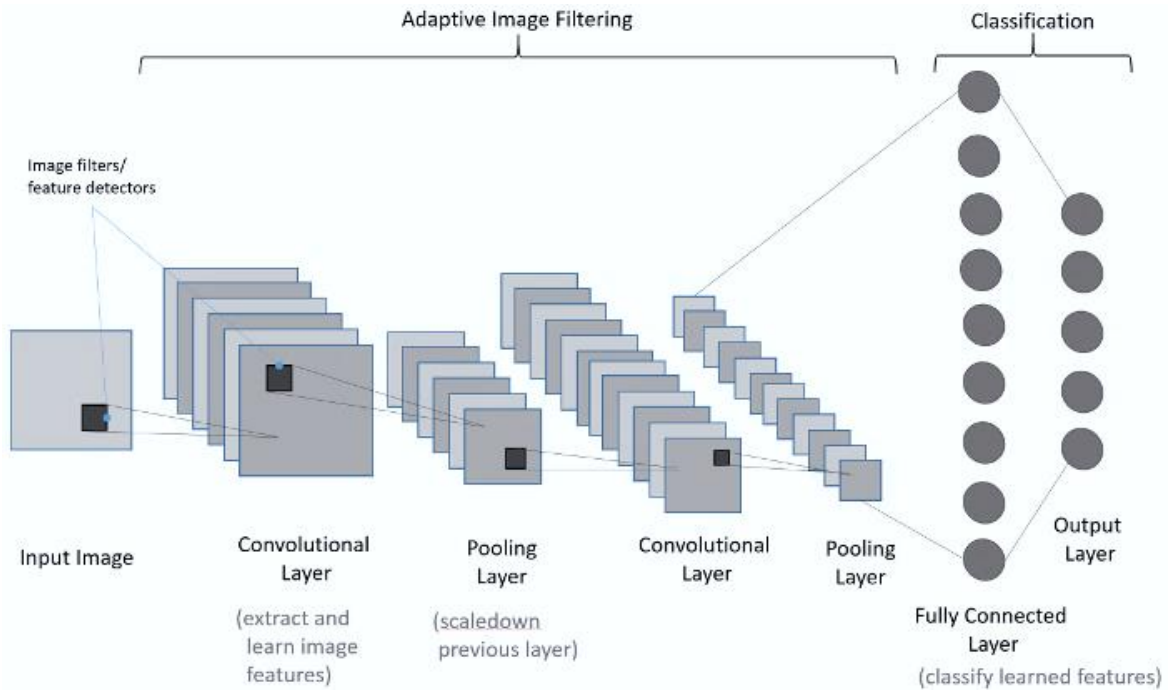
Τα MPLs συνήθως μοντελοποιούνται ως σύνολα τεχνητών νευρώνων-perceptron που συνδέονται μεταξύ τους σχηματίζοντας έναν άκυκλο γράφο, και οι έξοδοι κάποιων νευρώνων είναι είσοδοι για κάποιους άλλους. Οι κύκλοι δεν επιτρέπονται, καθώς μπορεί να δημιουργηθεί ένας ατέρμονας βρόγχος κατά το πέρασμα προς τα εμπρός. Τα MPLs οργανώνονται σε διακριτά επίπεδα νευρώνων. Ο πιο συνηθισμένος τύπος είναι τα **πλήρως διασυνδεμένα (fully-connected)** δίκτυα, στα οποία οι νευρώνες μεταξύ δύο γειτονικών επιπέδων, είναι πλήρως διασυνδεμένοι μεταξύ τους, αλλά οι νευρώνες του ίδιου επιπέδου δεν συνδέονται μεταξύ τους, όπως φαίνεται στην Εικόνα 4-3.



Εικόνα 4-3. Απλό νευρωνικό δίκτυο και βαθύ νευρωνικό δίκτυο

Πηγή: L. Fridman, "Deep Learning Basics." [Online]. Available: https://www.dropbox.com/s/c0g3sc1shi63x3q/deep_learning_basics.pdf?dl=0 [Accessed: 30-Sep-2020]

Τα CNNs είναι παρόμοια με τα MLPs, αλλά η αρχιτεκτονική τους είναι ειδικά σχεδιασμένη για την επεξεργασία εικόνων ως είσοδο, όπως φαίνεται ενδεικτικά στην Εικόνα 4-4.

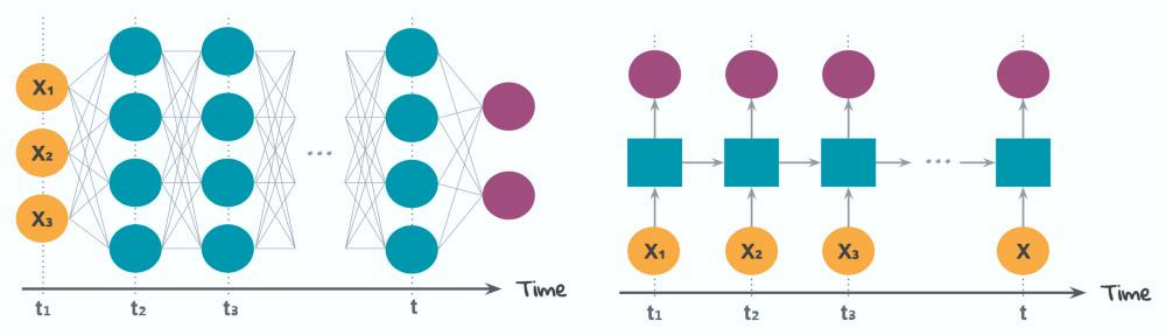


Εικόνα 4-4. Συνελικτικό νευρωνικό δίκτυο

Πηγή: https://www.deepnetts.com/blog/from-basic-machine-learning-to-deep-learning-in-5-minutes.html

Πιο συγκεκριμένα, οι νευρώνες διατάσσονται σε τρεις διαστάσεις: πλάτος, ύψος, βάθος. Είναι κατάλληλα για χωρικά δεδομένα (spatial data), αναγνώριση αντικειμένων (object recognition) και ανάλυση εικόνας (image analysis) με τη χρήση πολυδιάστατων δομών νευρώνων. Ο κύριος λόγος της δημοφιλίας της ΒΔ τα τελευταία χρόνια οφείλεται στην ανάπτυξη των CNNs. Κάποιες από τις κύριες εφαρμογές τους είναι τα αυτόνομα οχήματα (self-driving cars), τα drones, η όραση μέσω υπολογιστή (computer vision) και η ανάλυση κειμένου (text analytics).

Τα RNNs είναι επίσης εμπρόσθιας τροφοδότησης, αλλά με ανατροφοδοτούμενους βρόγχους μνήμης που δέχονται είσοδο από τα προηγούμενα ή και τα ίδια επίπεδα, όπως φαίνεται ενδεικτικά στην Εικόνα 4-5. Οι συνδέσεις τους σχηματίζουν έναν κατευθυνόμενο γράφο, όπου επιτρέπονται κύκλοι. Αυτό τους δίνει μια μοναδική δυνατότητα μοντελοποιήσης κατά μήκος της χρονικής διάστασης και της αυθαίρετης ακολουθίας συμβάντων και εισόδων.

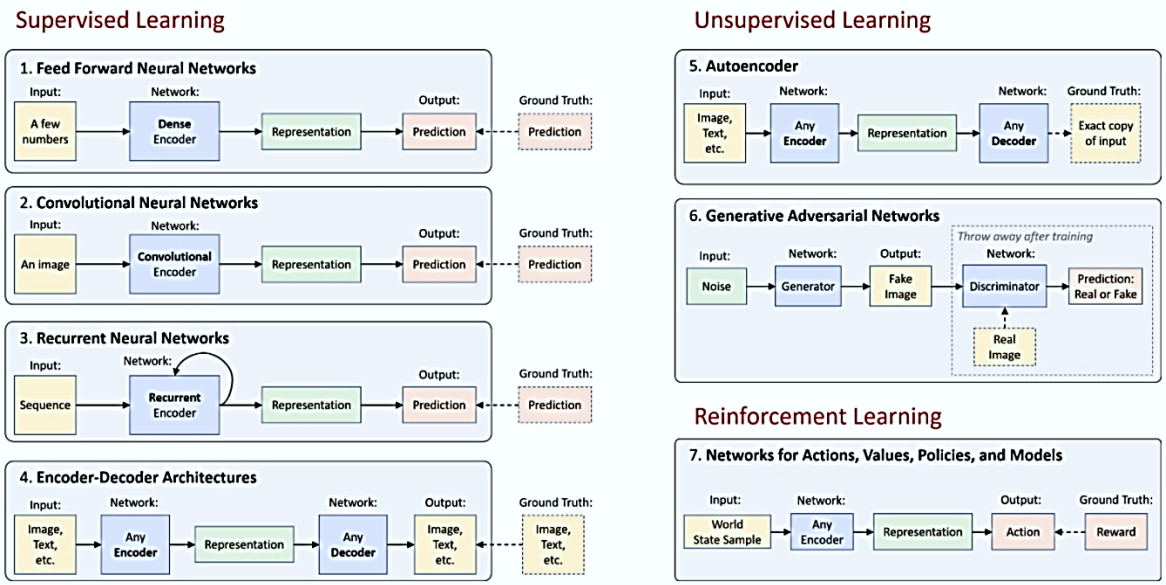


Εικόνα 4-5. Ανατροφοδοτούμενο νευρωνικό δίκτυο

Πηγή: https://mc.ai/deep-learning-series-chapter-1-introduction-to-deep-learning/

Με πιο απλά λόγια, για κάθε στιγμιότυπο, το δίκτυο διατηρεί μια μνήμη και μπορεί να προβλέψει την επόμενη ενέργεια. Ο πιο συχνοί τύποι των RNN μοντέλων είναι τα Long Short Term Memory (LSTM) δίκτυα. Τα RNNs, μεταξύ των άλλων, βρίσκουν εφαρμογή για πρόβλεψη ακολουθίας λέξεων και μάθηση γραμματικής.

Επίσης, σημειώνεται το γεγονός ότι, στις μέρες μας υπάρχουν διαφόρων ειδών ΤΝΔ για ΒΜ για κάθε κατηγορία μάθησης και τύπους δεδομένων, όπως φαίνεται συνοπτικά στην Εικόνα 4-6.



Εικόνα 4-6. Τα είδη ΤΝΔ για ΒΜ ανάλογα με τον τύπο μάθησης

Πηγή: L. Fridman, "MIT Deep Learning Basics: Introduction and Overview with TensorFlow." [Online]. Available: https://medium.com/tensorflow/mit-deep-learning-basics-introduction-and-overview-with-tensorflow-355bcd26baf0 [Accessed: 31-Aug-2020]

Τα MPLs, αποτελούν το κύριο αντικείμενο της εργασίας, κατά συνέπεια παρουσιάζονται αναλυτικά στην επόμενη παράγραφο.

4.3 Τα Πολυεπίπεδα Perceptrons (Multilayer Perceptrons – MLPs)

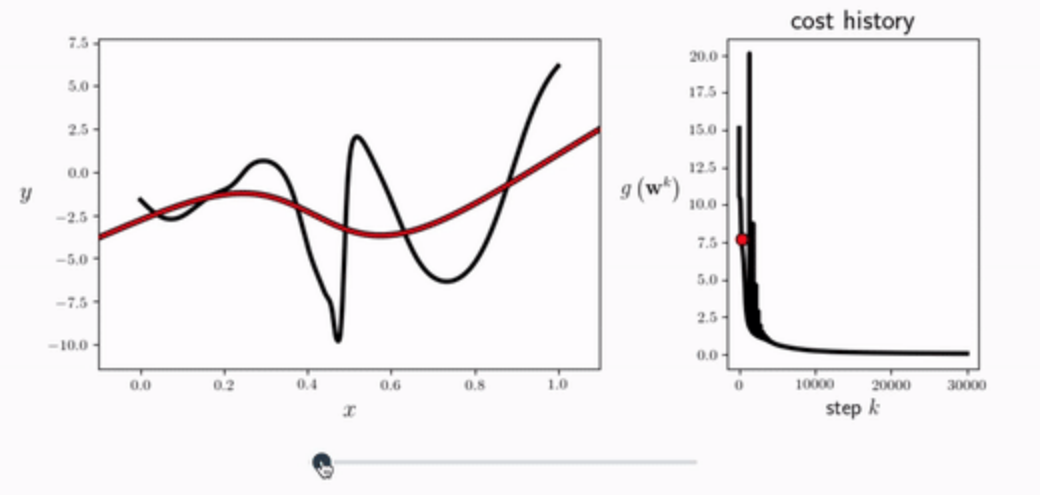
Όπως ήδη αναφέρθηκε παραπάνω, τα MLPs είναι σύνολα τεχνητών νευρώνων οργανωμένων σε επίπεδα με βασική δομική μονάδα το perceptron, τα οποία συνδέονται μεταξύ τους σχηματίζοντας έναν άκυκλο κατευθυνόμενο γράφο όπου οι ακμές είναι τα βάρη και οι πολώσεις. Οι έξοδοι κάποιων νευρώνων είναι είσοδοι για κάποιους άλλους και οργανώνονται σε διακριτά επίπεδα πλήρως διασυνδεμένων νευρώνων που ορίζουν την αρχιτεκτονική του δικτύου, όπως φαίνεται στην Εικόνα 4-3, και θα εξετάσουμε τις σχετικές έννοιες των MLPs πιο αναλυτικά. Πιο συγκεκριμένα:

- Η βασική δομική μονάδα είναι το perceptron. Οι συναρτήσεις ενεργοποίησης είναι μη-γραμμικές. Το δίκτυο παραμετροποιείται από τα βάρη και τις πολώσεις των μονάδων.
- Το τελευταίο επίπεδο είναι το **επίπεδο εξόδου (output layer)** και η συνάρτηση ενεργοποίησης εξαρτάται από το είδος του προβλήματος.
- Όλα τα επίπεδα πριν το επίπεδο εξόδου είναι τα **κρυφά επίπεδα (hidden layers)**.
- Ο αριθμός των μονάδων ενός επιπέδου ονομάζεται **πλάτος (width)** του επιπέδου. Το πλάτος κάθε επιπέδου δεν είναι κατ' ανάγκη ίδιο σε κάθε επίπεδο.
- Ο αριθμός των επιπέδων αναφέρεται ως **βάθος (depth)** του δικτύου και από αυτή την ονομασία προέρχεται ο όρος βαθιά (deep) στην ΒΜ. Το επίπεδο εισόδου δεν προσμετράται ως επίπεδο, γιατί δεν υπάρχουν παράμετροι εκμάθησης. Δηλαδή, το βαθύ νευρωνικό δίκτυο της Εικόνας 12, είναι ένα δίκτυο πέντε επιπέδων με τέσσερα κρυφά επίπεδα.
- Κάθε επίπεδο δέχεται ως είσοδο την έξοδο που παράγεται από το προηγούμενο επίπεδο, εκτός από το πρώτο επίπεδο, το **επίπεδο εισόδου (input layer)**, το οποίο δέχεται μόνο εισόδους.
- Η έξοδος του τελευταίου επιπέδου είναι η έξοδος του δικτύου και είναι η πρόγνωση που δημιουργείται ανάλογα με την είσοδο.

Τα επίπεδα αναπαρίστανται ως απλά διανύσματα τιμών και οι παράμετροι ως πίνακες. Για παράδειγμα, στο απλό δίκτυο με ένα κρυφό επίπεδο της Εικόνας 12, το επίπεδο εισόδου είναι ένα διάνυσμα τιμών $\boldsymbol{x}$ διαστάσεων [5x1]. Τα βάρη του κρυφού επιπέδου είναι ένας πίνακας τιμών $\boldsymbol{w}$ διαστάσεων [7x5]. Η έξοδος των μονάδων του επιπέδου είναι ένα διάνυσμα τιμών και προκύπτει από την εφαρμογή της συνάρτησης ενεργοποίησης στο σταθμισμένο άθροισμα $\boldsymbol{x^T w}$**.** Η έξοδος που υπολογίζεται από το κρυφό επίπεδο παράγει μια βαθμωτή τιμή που μπορεί να πάρει 4 διακριτές τιμές, εφόσον το επίπεδο εξόδου έχει 4 μονάδες.

Ο στόχος του MLP είναι η προσέγγιση μιας συνάρτησης $y= f^*(\boldsymbol{x})$, όπου αντιστοιχίζεται μια είσοδος $\boldsymbol{x}$ σε μια διακριτή κλάση y (για εργασία ταξινόμησης) ή τιμή y (για εργασία παλινδρόμησης). Το MLP ορίζει μια αντιστοίχιση $y = f(\boldsymbol{x},\boldsymbol{\theta})$ και μαθαίνει την τιμή των παραμέτρων $\boldsymbol{\theta}$ (βάρη και πολώσεις) που έχει ως αποτέλεσμα την καλύτερη προσέγγιση της συνάρτησης [1]. Οπότε, εξετάζουμε πως οι παραπάνω έννοιες εκφράζονται μαθηματικά και συνδέονται με το μοντέλο του τεχνητού νευρώνα.

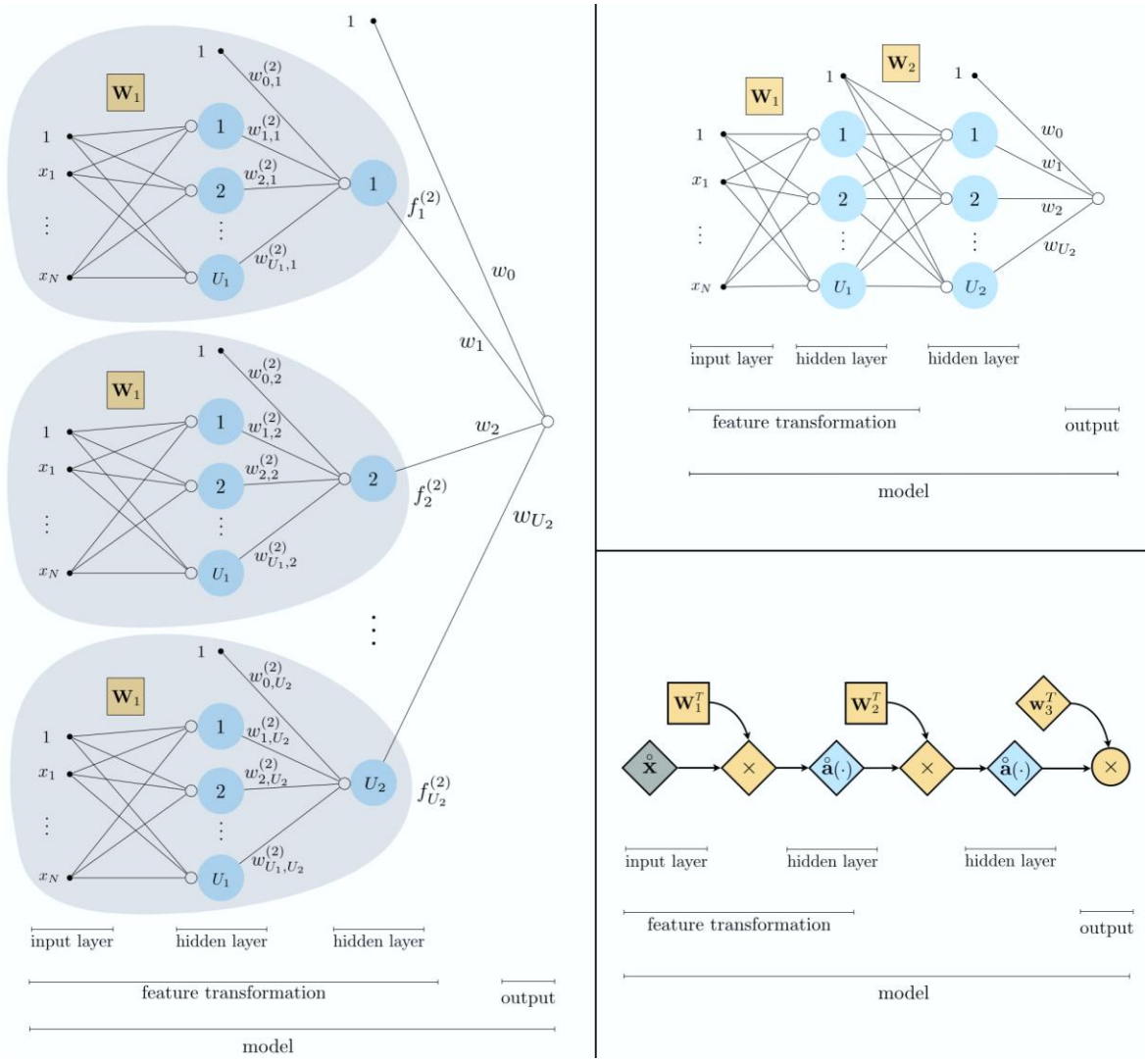
Το μοντέλο συνδέεται με τον κατευθυνόμενο άκυκλο γράφο που περιγράφει τη σύνθεση της συνάρτησης. Για παράδειγμα, μπορεί να έχουμε τρεις συναρτήσεις f^1, f^2, f^3 που συνδέονται αλυσιδωτά για τα σχηματίσουν μια μη γραμμική συνάρτηση $f(\boldsymbol{x}) = f^1\left(f^2\left(f^3(x)\right)\right)$ (εδώ, είναι χρήσιμο να ανατρέξουμε στην παράγραφο 3.3.1 για τους γραμμικούς και μη γραμμικούς μετασχηματισμούς). Σε αυτή την περίπτωση η f^1 είναι το πρώτο επίπεδο μη-γραμμικού μετασχηματισμού, η f^2 είναι το δεύτερο επίπεδο και ούτω καθεξής. Το μήκος αυτής της αλυσίδας, είναι το βάθος του δικτύου. Κατά τη διάρκεια της εκπαίδευσης καθοδηγούμε την $f(\boldsymbol{x})$ να προσεγγίσει την $f^*(\boldsymbol{x})$. Τα δεδομένα της εκπαίδευσης παρέχουν προσεγγιστικά παραδείγματα της $f^*(\boldsymbol{x})$ όπως αποτιμώνται για τα διάφορα παραδείγματα του συνόλου εκπαίδευσης. Τα παραδείγματα της εκπαίδευσης ορίζουν απευθείας τι πρέπει να κάνει το επίπεδο εξόδου για κάθε $\boldsymbol{x}$: πρέπει να παραχθεί μια τιμή που θα είναι κοντά στο y στο επίπεδο εξόδου. Η συμπεριφορά των ενδιάμεσων επιπέδων δεν ορίζεται απευθείας από τα δεδομένα εκπαίδευσης. Ο αλγόριθμος μάθησης θα πρέπει να αποφασίσει πως θα χρησιμοποιήσει αυτά τα επίπεδα για να παράγει την επιθυμητή έξοδο για την καλύτερη υλοποίηση της προσέγγισης της f^*. Επειδή τα δεδομένα εκπαίδευσης δεν υποδεικνύουν την επιθυμητή έξοδο για κάθε ένα από αυτά τα επίπεδα, γι' αυτό και ονομάζονται κρυφά επίπεδα [1]. Η Εικόνα 4-7 είναι μια ενδεικτική οπτικοποίηση της προσέγγισης μιας μη γραμμικής συνάρτησης από το δίκτυο κατά τη διάρκεια των κύκλων της εκπαίδευσης [31].



Εικόνα 4-7. Οπτικοποίηση προσέγγισης μιας μη γραμμικής συνάρτησης από νευρωνικό δίκτυο

Κάθε επίπεδο του δικτύου τυπικά περιέχει τιμές με μορφή διανύσματος. Η διαστασιμότητα (dimensionality) του κάθε επιπέδου ορίζει το πλάτος του επιπέδου. Κάθε στοιχείο του διανύσματος είναι και ένας τεχνητός νευρώνας, η μονάδα. Οι μονάδες σε κάθε επίπεδο ενεργούν παράλληλα και κάθε μία αναπαριστά μια συνάρτηση διανύσματος προς βαθμωτό μέγεθος. Κάθε μονάδα μοιάζει με τον βιολογικό νευρώνα με την έννοια ότι λαμβάνει είσοδο από πολλές άλλες μονάδες και υπολογίζει τη δική της συνάρτηση ενεργοποίησης. Η ιδέα της χρήσης πολλών επιπέδων με αναπαράσταση διανυσμάτων με τιμές προκύπτει από την νευρό επιστήμη. Η επιλογή των συναρτήσεων $f^i(\boldsymbol{x})$ που χρησιμοποιούνται για τον υπολογισμό αυτών των αναπαραστάσεων, οδηγείται επίσης από παρατηρήσεις της νευροεπιστήμης για το πώς υπολογίζουν συναρτήσεις οι βιολογικοί νευρώνες [1].

Μια εξαιρετική απεικόνιση ενός MLP με δύο κρυφά επίπεδα η οποία συμπεριλαμβάνει τις έννοιες που παρουσιάστηκαν παραπάνω δίνουν οι συγγραφείς στην [31] με την Εικόνα 4-8. Αριστερά στην εικόνα δίνεται η λεπτομερής γραφική απεικόνιση για όλο το σύνολο εκπαίδευσης. Δεξιά επάνω στην εικόνα δίνεται η απλούστερη απεικόνιση του άκυκλου γράφου για ένα παράδειγμα εκπαίδευσης. Δεξιά κάτω στην εικόνα δίνεται η υψηλού επιπέδου (high-level) απεικόνιση του υπολογιστικού γραφήματος του δικτύου με διανύσματα.



Εικόνα 4-8. Απεικόνιση των εννοιών ενός MLP τριών επιπέδων

Πηγή: J. Watt, R. Borhani, and A. K. Katsaggelos, "Machine Learning Refined." [Online]. Available: https://github.com/jermwatt/machine_learning_refined. [Accessed: 30-Sep-2020]

Ένας καλός τρόπος για την κατανόηση της λειτουργίας των MLP είναι η κατανόηση των περιορισμών των γραμμικών μοντέλων και πως ξεπερνούνται αυτοί οι περιορισμοί με την μετατροπή τους σε μη γραμμικά. Τα γραμμικά μοντέλα τείνουν να εξαφανιστούν γιατί αφενός δεν μπορούν να προσαρμοστούν αποτελεσματικά και αξιόπιστα και αφετέρου η χωρητικότητά τους περιορίζεται σε γραμμικές συναρτήσεις.

Η σχεδίαση ενός μοντέλου MLP αφορά, μεταξύ των άλλων, τον ορισμό της δομής του, συμπεριλαμβανομένων του αριθμού των επιπέδων και το πλάτος αυτών των επιπέδων που είναι **υπερπαράμετροι** του δικτύου και ορίζουν την **χωρητικότητα** του δικτύου. Σαφής απάντηση για

το πόσα θα είναι τα κρυφά επίπεδα και το πλάτος του επιπέδου δεν υπάρχει. Στο σημείο αυτό, ανατρέχοντας στην παράγραφο υπενθυμίζεται το under fitting και το over fitting. Σίγουρα, αυξάνοντας τη χωρητικότητα αυξάνεται και η ικανότητα του δικτύου να αναπαριστά πιο σύνθετες συναρτήσεις, αλλά υπάρχει ο κίνδυνος του over fitting όπως αναλύεται πιο λεπτομερώς από τους συγγραφείς στις [1], [22], [23], [26]. Γενικά, για σύνολα δεδομένων όπου τα παραδείγματα δεν έχουν σύνθετη μορφή, δύο κρυφά επίπεδα στα MLPs αρκούν για την αποφυγή του over fitting. Σύμφωνα με τις ίδιες πηγές εκτός από την μεταβολή της χωρητικότητας, μεγάλη σπουδαιότητα στα MLPs έχει η εξομάλυνση.

Γενικά, η ροή εργασιών για την εκπαίδευση, επικύρωση και αποτίμηση ενός MLP μοντέλου, είναι αυτή που ισχύει για κάθε μοντέλο ΜΜ, όπως παρουσιάστηκε στην παράγραφο 3.2.5. Οι ιδιαιτερότητες αφορούν την παραμετροποίηση και τους αλγόριθμους του μοντέλου για την προσέγγιση της κατάλληλης συνάρτησης, καθώς και τις υπερπαραμέτρους για την βελτιστοποίηση και εξομάλυνση κατά τη φάση της εκπαίδευσης, οπότε στη συνέχεια θα επικεντρωθούμε σε αυτά.

4.4 Η Εκπαίδευση του MLP

Εφόσον αποφασιστεί αρχικά η σχεδίαση του δικτύου, ακολουθεί η εκπαίδευση. Στόχος της εκπαίδευσης είναι από τα παραδείγματα είναι να βρεθούν οι κατάλληλες τιμές για τις παραμέτρους του δικτύου, οι οποίες στις περίπτωση των MLPs είναι τα βάρη και οι πολώσεις, ώστε το δίκτυο με βάση τα δεδομένα εισόδου να μπορεί να προβλέψει με όσο το δυνατόν μεγαλύτερη ακρίβεια μια έξοδο με βάση τις υπερπαραμέτρους που έχουν οριστεί.

Η έξοδος του δικτύου, εφόσον η πληροφορία ρέει από την είσοδο προς την έξοδο, υπολογίζεται με βάση έναν αλγόριθμο **εμπροσθοδιάδοσης (forward propagation)**. Η απόκλιση της τιμής εξόδου που υπολογίζεται από το δίκτυο με βάση τα δεδομένα εκπαίδευσης σε σχέση με τη μεταβλητή στόχο, υπολογίζεται από τη συνάρτηση κόστους, οποία αντικατοπτρίζει το πόσο καλά λειτουργεί το δίκτυο. Η βελτιστοποίηση του δικτύου επιτυγχάνεται με την ελαχιστοποίηση της τιμής που υπολογίζεται από τη συνάρτηση κόστους. Η διαδικασία για την βελτιστοποίηση της συνάρτησης κόστους βασίζεται σε έναν αλγόριθμο βελτιστοποίησης **καθόδου με βάση την κλίση (gradient descent)**. Ο βασικός αλγόριθμος για την εκπαίδευση των MPLs είναι αυτός που διατυπώθηκε το 1986 από τους Rumelhart, Hinton και Williams είναι ο **αλγόριθμος**

οπισθοδιάδοσης (back-propagation algorithm), ο οποίος χρησιμοποιείται ακόμη και σήμερα. Η γενική ιδέα της οπισθοδιάδοσης είναι να τροποποιούνται οι παράμετροι του δικτύου επαναληπτικά με τέτοιο τρόπο ώστε να ελαχιστοποιείται η τιμή που υπολογίζεται από τη συνάρτηση κόστους [1][22]. Όλοι οι υπολογισμοί αφορούν πράξεις γραμμικής άλγεβρας, όπως θα δούμε αναλυτικά στις επόμενες παραγράφους.

Η διαδικασία της εκπαίδευσης ενός MLP περιγράφεται απλά για ένα παράδειγμα ***x*** με στόχο *y* από το σύνολο της εκπαίδευσης ως εξής:

1. Ορισμός της αρχιτεκτονικής του δικτύου L επιπέδων, επιλογή της συνάρτησης ενεργοποίησης στα κρυφά επίπεδα και στο επίπεδο εξόδου, επιλογή της συνάρτησης κόστους, επιλογή εξομάλυνσης και επιλογή βελτιστοποιητή (επιλογή υπερπαραμέτρων).

*2. Τυχαία αρχικοποίηση των παραμέτρων εκμάθησης **w**, **b** για κάθε επίπεδο l.*

*3. Είσοδος στο σύστημα ενός παραδείγματος εκπαίδευσης **x** με τιμή στόχο y.*

4. Υπολογισμός της πρόβλεψης με τον αλγόριθμο εμπροσθοδιάδοσης:

Η είσοδος x αντιστοιχεί σε μια τιμή στόχο y

Για k= 1, …., L-1, L υπολογισμός:

Τιμής ενεργοποίησης

Εφαρμογή της συνάρτησης ενεργοποίησης

Τέλος επανάληψης

Επιστροφή της τιμής εξόδου y_{pred}

5. Υπολογισμός του λάθους εξόδου με εφαρμογή της συνάρτησης κόστους σε y_{pred}, y

6. Εφαρμογή του αλγόριθμου οπισθοδιάδοσης για την ενημέρωση των βαρών και των πολώσεων έτσι ώστε να μειωθεί η απώλεια στην πρόβλεψη:

Μετά την εμπροσθοδιάδοση υπολογισμός σφάλματος εξόδου (κλίση εξόδου)

Για k= L, L-1,…., 1 :

Μετατροπή της κλίσης επιπέδου εξόδου σε κλίση προ-γραμμικής ενεργοποίησης.

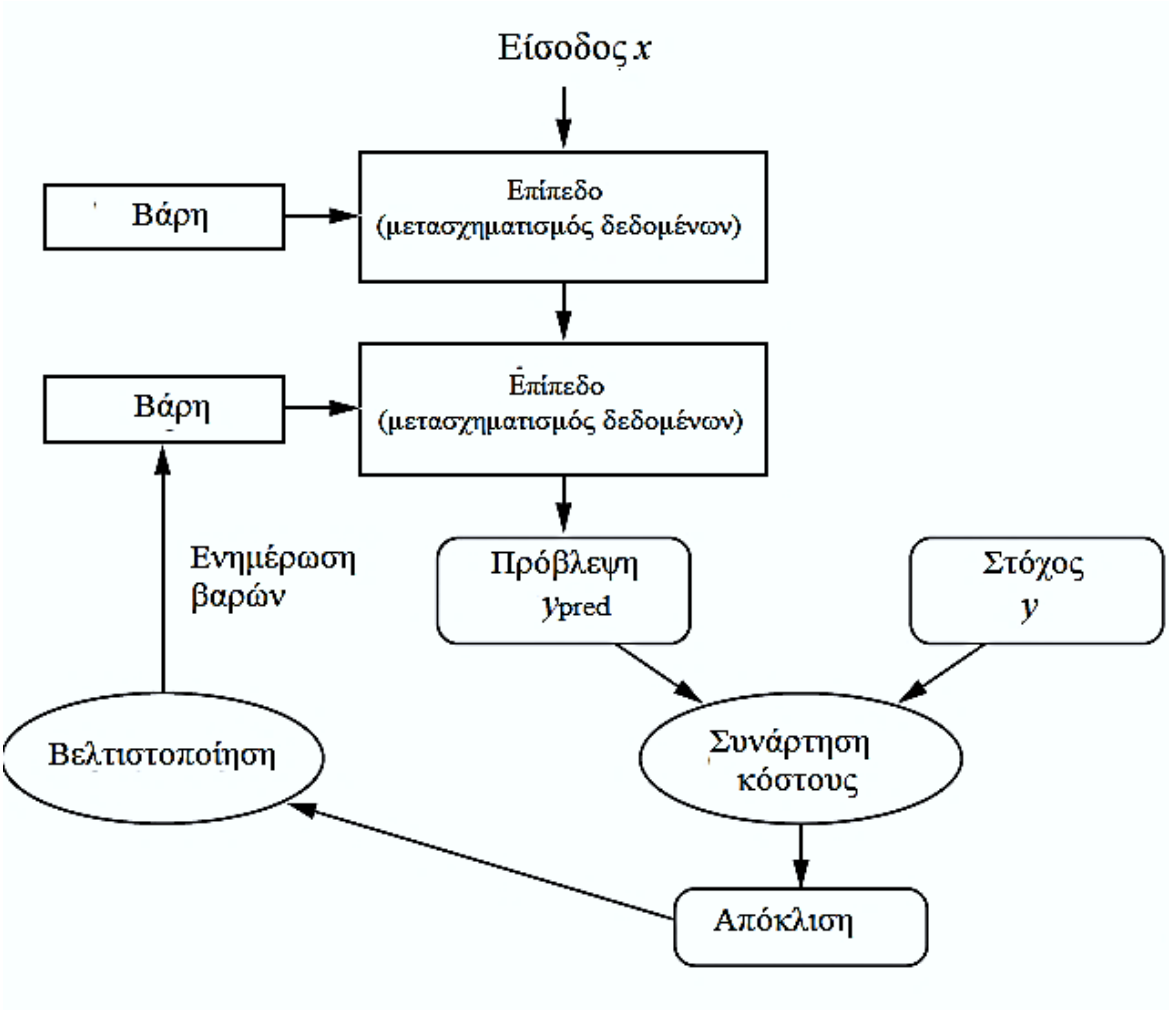
Υπολογισμός των κλίσεων σε βάρη και πολώσεις

Διάδοση των κλίσεων προς τα πίσω στις ενεργοποιήσεις του επόμενου χαμηλότερου επιπέδου.

Τέλος επανάληψης

8. Επανάληψη της διαδικασίας μέχρι να ικανοποιηθεί το κριτήριο σύγκλισης

Η παραπάνω διαδικασία από το βήμα 3 και μετά, φαίνεται διαγραμματικά στην Εικόνα 4-9.



Εικόνα 4-9. Διαγραμματική απεικόνιση της εκπαίδευσης του MLP

Ο επανάληψη των υπολογισμών για όλα τα παραδείγματα που περιλαμβάνονται στο σύνολο της εκπαίδευσης, ονομάζεται **εποχή (epoch)**. Εάν το σύνολο εκπαίδευσης περιέχει πολλά στοιχεία, τότε μπορεί ο υπολογισμός να γίνει με διαίρεση του κατά **δέσμες (batches)**. Για παράδειγμα, εάν το σύνολο περιέχει 1000 στοιχεία και το διαιρέσουμε σε δέσμες των 200 στοιχείων (batch size), τότε μια εποχή θα περιλαμβάνει 1000/200= 5 επαναλήψεις.

Πριν προχωρήσουμε στους υπολογισμούς και την εφαρμογή του αλγόριθμου οπισθοδιάδοσης, θα παρουσιάσουμε πρώτα στις επόμενες παραγράφους διαδοχικά τις συναρτήσεις ενεργοποίησης που

εφαρμόζονται στα MLPs, τις συναρτήσεις κόστους και τους κυριότερους αλγόριθμους βελτιστοποίησης που αποτελούν υπερπαραμέτρους για την εκπαίδευση.

4.5 Συναρτήσεις Ενεργοποίησης

Κάθε συνάρτηση ενεργοποίησης (ή μη γραμμικότητα) δέχεται ως όρισμα έναν απλό αριθμό ο οποίος είναι το σταθμισμένο άθροισμα της εισόδου και της πόλωσης και εκτελεί μια συγκεκριμένη μαθηματική πράξη με αυτόν. Η επιλογή της συνάρτησης ενεργοποίησης εξαρτάται από την εφαρμογή που βρίσκει το νευρωνικό δίκτυο. Πλήρως παραγωγίσιμες συναρτήσεις ενεργοποίησης είναι αυτές για τις οποίες υπάρχει η πρώτη παράγωγος για όλες τις τιμές στο πεδίο ορισμού της και αυτές προτιμώνται για τα νευρωνικά δίκτυα, λόγω του ότι ο αλγόριθμος εκπαίδευσης βασίζεται σε υπολογισμό παραγώγων. Η επιλογή της κατάλληλης συνάρτησης ενεργοποίησης εξαρτάται από το είδος του προβλήματος και από τη σχεδίαση του δικτύου. Στην πράξη, υπάρχουν διάφορες συναρτήσεις ενεργοποίησης, οι οποίες εφαρμόζονται στα νευρωνικά δίκτυα και κάθε μία έχει τα πλεονεκτήματά της και τα μειονεκτήματά της. Οι κυριότερες από αυτές περιγράφονται συνοπτικά στις επόμενες ενότητες. Για περισσότερες συναρτήσεις ενεργοποίησης, παραπέμπουμε στην εργασία των Ding, Qian και Zhou [32], καθώς και στο συγγράμματα [1], [22] όπου παρουσιάζονται τα πλεονεκτήματα και τα μειονεκτήματα τους, ανάλογα με την εφαρμογή .

4.5.1 Σιγμοειδής Συνάρτηση (Sigmoid)

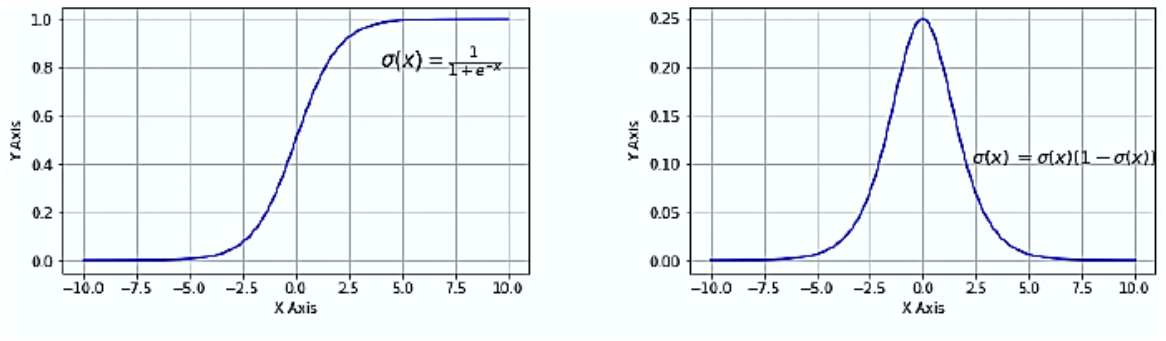
Η σιγμοειδής συνάρτηση υπολογίζεται από τον τύπο

$$\sigma(x) = \frac{1}{1+e^{-x}} \qquad (4.5.1)$$

Η πρώτη παράγωγός της είναι:

$$\sigma'(x) = \sigma(x)(1-\sigma(x)) \qquad (4.5.2)$$

Η γραφική της παράστασή της, καθώς και αυτή της παραγώγου της, φαίνεται στην Εικόνα 4-10.
Η σιγμοειδής συνάρτηση δέχεται ως είσοδο έναν πραγματικό αριθμό και περιορίζει την τιμή του στο διάστημα [0,1]. Πιο συγκεκριμένα, οι μεγάλοι αρνητικοί αριθμοί παίρνουν την τιμή 0 και οι μεγάλοι θετικοί αριθμοί την τιμή 1.
Είναι η δημοφιλέστερη ως συνάρτηση εξόδου για προβλήματα δυαδικής ταξινόμησης.



Εικόνα 4-10. Γραφική παράσταση της σιγμοειδούς συνάρτησης και της παραγώγου της

4.5.2 Συνάρτηση Υπερβολικής Εφαπτομένης (tanh)

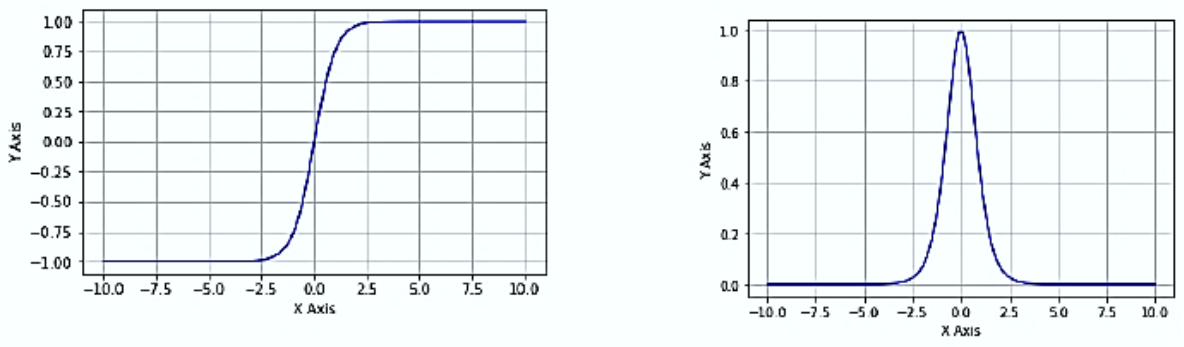
Η συνάρτηση υπερβολικής εφαπτομένης, δέχεται το ίδιο όρισμα με τη σιγμοειδή συνάρτηση, αλλά περιορίζει την τιμή εξόδου στο διάστημα [-1, 1] και υπολογίζεται από τον τύπο:

$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-z}} \qquad (4.5.3)$$

Η πρώτη παράγωγός της είναι:

$$f(x) = 1 - f(x)^2 \qquad (4.5.4)$$

Η γραφική παράσταση της υπερβολικής εφαπτομένης, καθώς και της παραγώγου της, φαίνεται στην Εικόνα 4-11.



Εικόνα 4-11. Γραφική παράσταση της συνάρτησης υπερβολικής εφαπτομένης και της παραγώγου της

Ο tanh νευρώνας είναι απλά ένας κλιμακωτός σιγμοειδής νευρώνας. Η συνάρτηση υπερβολικής εφαπτομένης και η σιγμοειδής συνάρτηση συνδέονται με τον τύπο:

$$tanh(x) = 2\sigma(2x) - 1 \qquad (4.5.5)$$

4.5.3 Ανορθωμένη Γραμμική Συνάρτηση (Rectified Linear Unit – ReLU).

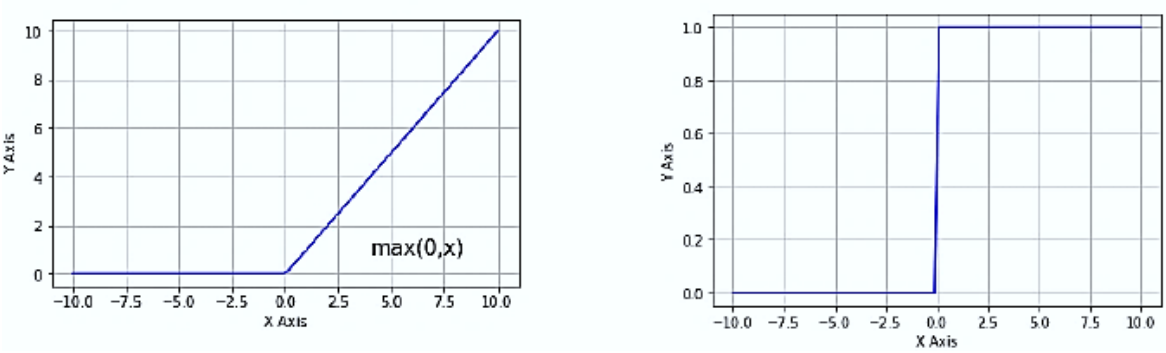
Η συνάρτηση ενεργοποίησης ReLU έγινε δημοφιλής τα τελευταία χρόνια ως συνάρτηση ενεργοποίησης νευρώνων στα κρυφά επίπεδα. Υπολογίζει τη συνάρτηση:

$$f(x) = max(0, x) = \begin{cases} 0 & \text{εάν } x < 0 \\ x & \text{εάν } x \geq 0 \end{cases} \qquad (4.5.6)$$

Δηλαδή, η ενεργοποίηση έχει απλά ως κατώφλι το μηδέν.
Η πρώτη παράγωγός της είναι:

$$f'(x) = \begin{cases} 1 & \text{εάν } x > 0 \\ 0 & \text{σε κάθε άλλη περίπτωση} \end{cases} \qquad (4.5.7)$$

Η γραφική παράσταση της ReLU, καθώς και της παραγώγου της, φαίνεται στην Εικόνα 4-12.



Εικόνα 4-12. Γραφική παράσταση της ReLU συνάρτησης και της παραγώγου της

Πλέον, με την ανάπτυξη των ΤΝΔ υπάρχουν διάφορες παραλλαγές της ReLU, όπως η Leaky rectified Linear Unit (LreLU), η Parametric rectified Linear Unit (PreLU), και η Randomized rectified Linear Unit (RreLU) με τις οποίες αντιμετωπίζονται τα μειονεκτήματα της ReLU [22], [32].

4.6 Συναρτήσεις κόστους

Στην απόκλιση της τιμής εξόδου που υπολογίζεται από το δίκτυο με βάση τα δεδομένα εκπαίδευσης σε σχέση με τη μεταβλητή στόχο, εφαρμόζεται η συνάρτηση κόστους, η οποία είναι

το μέτρο της απόδοσης του συστήματος. Εφόσον οι παράμετροι μάθησης του MLP είναι τα βάρη $\boldsymbol{w}$ και οι πολώσεις $\boldsymbol{b}$, το κόστος είναι μια συνάρτηση αυτών $C(\boldsymbol{w}, \boldsymbol{b})$ ή πιο απλά μιας παραμέτρου $\boldsymbol{\theta}$ που συμπεριλαμβάνει τα $\boldsymbol{w}$ και $\boldsymbol{b}$, δηλαδή $C(\boldsymbol{\theta})$. Η συνάρτηση κόστους συνήθως αποσυντίθεται ως άθροισμα των παραμέτρων της εκπαίδευσης [1], [26].

Συνήθως, η συνάρτηση κόστους επιλέγεται ανάλογα με το είδος του προβλήματος και αποτελεί υπερπαράμετρο του δικτύου. Έτσι, σε εφαρμογές MLP για εργασίες παλινδρόμησης συνήθως επιλέγεται η μέση τετραγωνική συνάρτηση κόστους (quadratic cost) και σε εφαρμογές ταξινόμησης η συνάρτηση εγκάρσιας εντροπίας (cross entropy), χωρίς αυτό βέβαια να είναι απόλυτο.

4.6.1 Συνάρτηση Μέσου Τετραγωνικού Λάθους (Mean Squared Error)

Μια απλή και συνηθισμένη συνάρτηση κόστους είναι η **μέση τετραγωνική συνάρτηση κόστους (Mean Squared Error Cost Function - MSE)**, η οποία υπολογίζεται από το μέσο τετραγωνικό σφάλμα των τιμών στόχων και των εξόδων που υπολογίζονται για όλα τα παραδείγματα του συνόλου εκπαίδευσης. Ως σφάλμα ορίζεται η διαφορά της τιμής πρόβλεψης y_{pred} σε σχέση με την τιμή στόχο y [1], όπου y είναι οι τιμές y_{true}.

Εάν $\boldsymbol{\theta}$ οι παράμετροι μάθησης του δικτύου, τότε η συνάρτηση κόστους είναι ο μέσος όρος του αθροίσματος των τετραγωνικών σφαλμάτων και δίνεται από την εξίσωση:

$$C(\boldsymbol{\theta}) = \frac{1}{m} \sum_i (y_{pred} - y)_i^2 \qquad (4.6.1)$$

Εάν η συνάρτηση ενεργοποίησης είναι η σιγμοειδής, τότε $y_{pred} = \sigma(z)$, όπου z το σταθμισμένο άθροισμα. Διαισθητικά, από την παραπάνω εξίσωση γίνεται αντιληπτό ότι το σφάλμα μειώνεται στο 0, όταν $y_{pred} = y$.

4.6.2 Συνάρτηση Δυαδικής Εγκάρσιας Εντροπίας (Binary Cost Entropy)

Στη περίπτωση της δυαδικής ταξινόμησης ο στόχος $y \in \{0,1\}$, οπότε μπορούμε να θεωρήσουμε πως, από το μοντέλο εκτιμάται η πιθανότητα[11] της περίπτωσης ενός στιγμιότυπου να ανήκει στην κλάση 0 ή 1 δοθέντος του x. Εάν η εκτίμηση της πιθανότητας είναι μεγαλύτερη του 50%, τότε το μοντέλο προβλέπει ότι το στιγμιότυπο ανήκει στην κλάση 1 (ή αλλιώς στη θετική κλάση), αλλιώς

[11]Το πρόβλημα ανάγεται στην εύρεση της μέγιστης πιθανότητας (maximum likelihood)

προβλέπει ότι ανήκει στην κλάση 0 (αρνητική κλάση). Σε μοντέλα ταξινόμησης, υπολογίζεται το σταθμισμένο άθροισμα *z* των χαρακτηριστικών της εισόδου, αλλά ως έξοδος δεν δίνεται η τιμή υπολογισμού, αλλά εφαρμόζεται μια συνάρτηση, όπως η σιγμοειδής, η οποία επιστρέφει τιμές στο εύρος [0,1], οπότε η πιθανότητα $p = \sigma(z)$, όπου z το σταθμισμένο άθροισμα [1], [22]. Συνεπώς, για ένα στιγμιότυπο του συνόλου εκπαίδευσης *x* με συνάρτηση ενεργοποίησης την σιγμοειδή η πρόβλεψη της εξόδου είναι:

$$y_{pred} = \begin{cases} 0 \text{ εάν } \sigma(z) < 0.5 \\ 1 \text{ εάν } \sigma(z) \geq 0.5 \end{cases} \quad (4.6.2)$$

Η συνάρτηση κόστους για ένα απλό στιγμιότυπο *x* με βάση την πιθανότητα p, εκφρασμένη στο λογαριθμικό διάστημα, δίνεται από τη σχέση:

$$C(\theta) = -y \log(p) - (1-y) \log(1-p), \text{ όπου } p = \sigma(z) \quad (4.6.3)$$

Πρακτικά, αυτό σημαίνει ότι:

$$C(\theta) = \begin{cases} -\log(p), & y = 1 \\ \log(1-p), & y = 0 \end{cases} \quad (4.6.4)$$

Αυτή η συνάρτηση κόστους έχει νόημα γιατί το – log(p) μεγαλώνει πολύ γρήγορα όταν το p τείνει στο 0, συνεπώς το κόστος θα είναι μεγάλο όταν το μοντέλο εκτιμά μια πιθανότητα κοντά στο 0 για ένα θετικό στιγμιότυπο, και το ίδιο συμβαίνει όταν το μοντέλο εκτιμά πιθανότητα κοντά στο 1 για ένα αρνητικό στιγμιότυπο. Από την άλλη, το – log(p) είναι κοντά στο 0 όταν το p είναι κοντά στο 1, συνεπώς το κόστος θα είναι κοντά στο 0 όταν η εκτιμώμενη πιθανότητα είναι κοντά στο 0 για ένα αρνητικό στιγμιότυπο ή κοντά στο 1 για ένα θετικό στιγμιότυπο, το οποίο είναι το επιθυμητό [22].

Η συνάρτηση κόστους για όλα τα στιγμιότυπα *m* του συνόλου εκπαίδευσης είναι ο μέσος όρος του κόστους που προκύπτει και δίνεται από τη σχέση:

$$C(\boldsymbol{\theta}) = -\frac{1}{m}\left[\sum_{i=0}^{m} y^i \log\left(p^i\right) + \left(1 - y^i\right) \log\left(1 - p^i\right)\right] \quad (4.6.5)$$

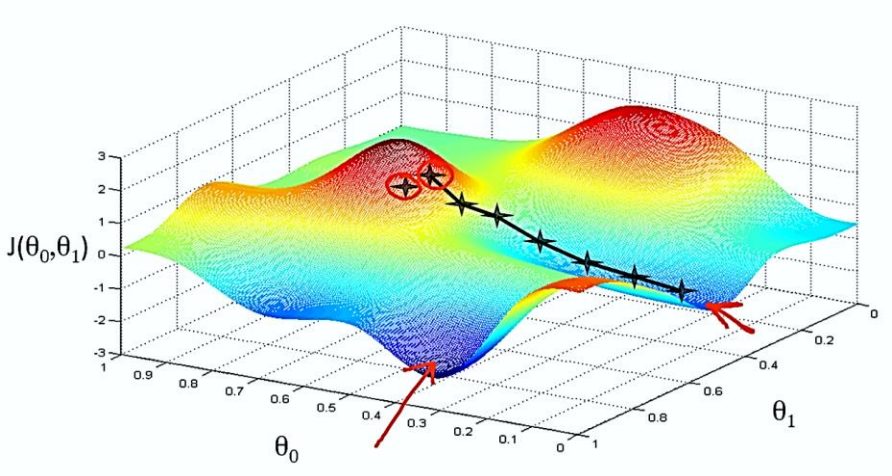
η οποία ονομάζεται συνάρτηση **δυαδικής εγκάρσιας εντροπίας (binary cost entropy).** Ο όρος εγκάρσια εντροπία προέρχεται από τη θεωρία της πληροφορίας, όπου η εγκάρσια εντροπία μεταξύ

δύο κατανομών πιθανοτήτων *p* και *q* για το ίδιο υποκείμενο σύνολο συμβάντων μετρά τον μέσο όρο των bit που απαιτούνται για τον προσδιορισμό ενός συμβάντος που αντλείται από το σύνολο βελτιστοποιείται για την εκτιμώμενη κατανομή πιθανότητας *q* αντί για την πραγματική κατανομή *p*[12].

4.7 Αλγόριθμοι Βελτιστοποίησης

Γενικά μιλώντας, ένα πρόβλημα **βελτιστοποίησης (optimization)** αφορά την εύρεση των παραμέτρων που ελαχιστοποιούν μια μαθηματική συνάρτηση και στην ΜΜ αυτή η συνάρτηση είναι η συνάρτηση κόστους. Ο αλγόριθμος για αυτή τη λειτουργία ονομάζεται **βελτιστοποιητής (optimizer)** και αποτελεί υπερπαράμετρο του δικτύου. Ο πιο συνηθισμένος αλγόριθμος για την βελτιστοποίηση συναρτήσεων κόστους είναι ο αλγόριθμος **καθόδου με βάση την κλίση (gradient descent)** και οι παραλλαγές του.

Διαισθητικά, ο αλγόριθμος μπορεί να παρομοιαστεί με την κατάβαση από την κορυφή ενός λόφου προς το χαμηλότερο σημείο μιας πεδιάδας όταν επικρατεί ομίχλη και δεν βλέπουμε αυτό το σημείο. Μπορούμε να νιώσουμε την κλίση του εδάφους, οπότε καλή στρατηγική είναι η κάθοδος προς την κατεύθυνση της πιο απότομης κλίσης. Η απεικόνιση μιας συνάρτησης στον χώρο με δύο παραμέτρους θυμίζει ένα σύνθετο τοπίο λόφων και πεδιάδων και η διαδρομή που ακολουθείται σύμφωνα με αυτή τη στρατηγική φαίνεται απλά στην Εικόνα 4-13.



Εικόνα 4-13. Η διαδρομή της καθόδου με βάση την κλίση

Πηγή: https://hackernoon.com/gradient-descent-aynk-7cbe95a778da

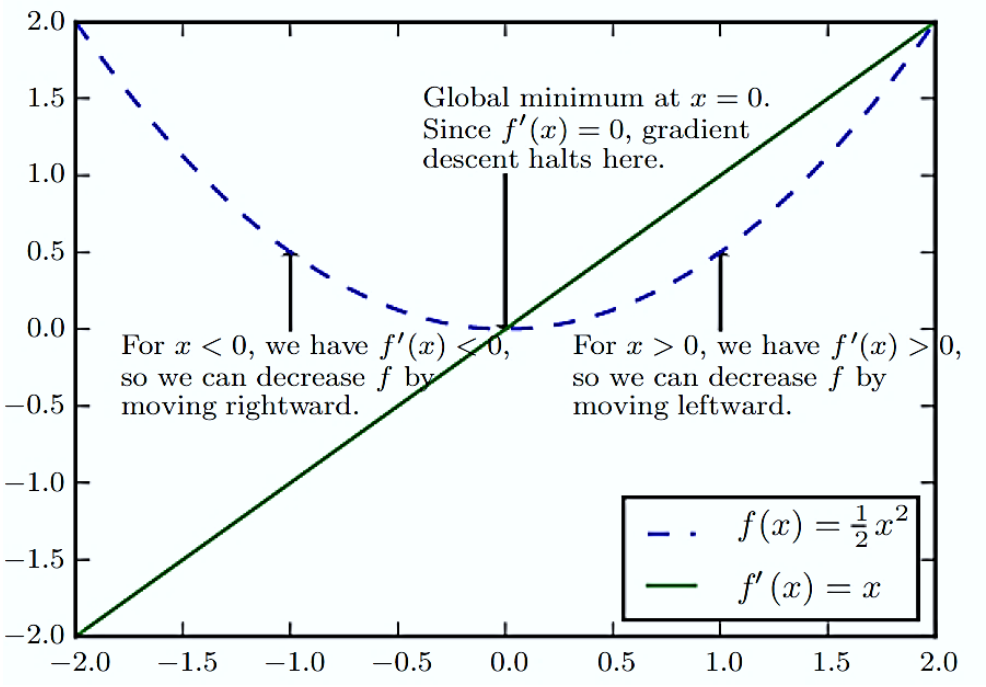
[12] https://en.wikipedia.org/wiki/Cross_entropy

Στην πράξη εφαρμόζονται διάφορες παραλλαγές του αλγόριθμου gradient descent και παρουσιάζονται στις επόμενες παραγράφους.

4.7.1 Gradient Descent και Batch Gradient Descent

Η βελτιστοποίηση γενικά, είναι μια εργασία κατά την οποία είτε ελαχιστοποιείται είτε μεγιστοποιείται μια συνάρτηση $f(x)$ μεταβάλλοντας το *x*. Συνήθως ο όρος βελτιστοποίηση χρησιμοποιείται για την ελαχιστοποίηση. Η συνάρτηση που πρέπει να ελαχιστοποιηθεί είναι η συνάρτηση κόστους [1]. Έστω *y*= $f(x)$, όπου *x*, *y* πραγματικοί αριθμοί. Η παράγωγος $f'(x)$ ή $\frac{dy}{dx}$ γραφικά απεικονίζεται ως η κλίση (slope) της $f(x)$ στο σημείο *x*. Με άλλα λόγια, ορίζει πως κλιμακώνεται μια μικρή αλλαγή στην είσοδο για να ληφθεί η αντίστοιχη αλλαγή στην έξοδο, δηλαδή η παράγωγος δείχνει πως θα αλλάξει η είσοδος ***x***, ώστε να βελτιωθεί η έξοδος *y*: $f(x + \varepsilon) \approx \varepsilon f(x)$. Επομένως, η παράγωγος είναι χρήσιμη στην ελαχιστοποίηση μιας συνάρτησης, γιατί δίνει πως θα πρέπει να αλλάξει το x προκειμένου να υπάρχει μια μικρή βελτίωση στο *y*.

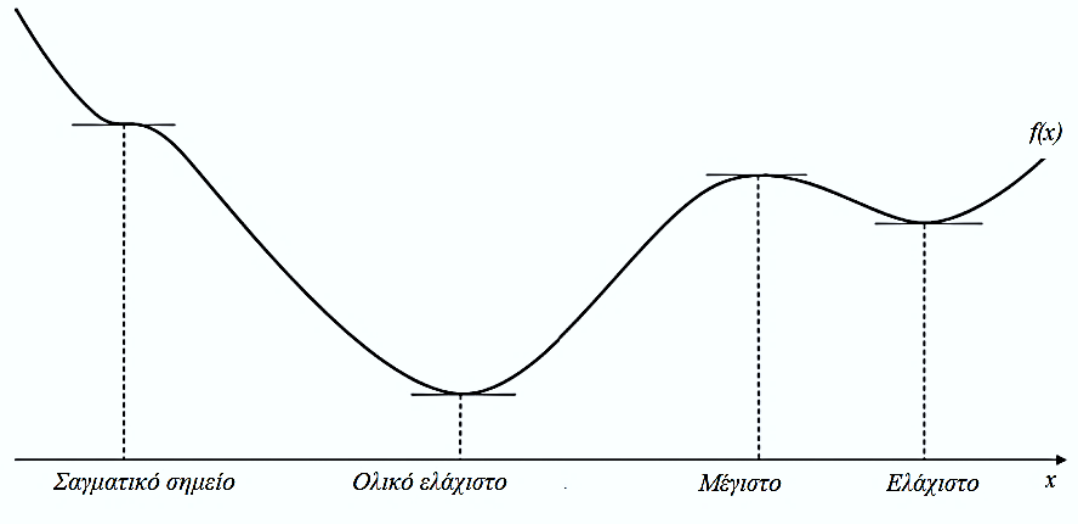
Από τα μαθηματικά, είναι γνωστό ότι $f(x - \varepsilon sign(f'(x))) < f(x)$, για πολύ μικρό ε. Ετσι, η $f(x)$ μπορεί να μειωθεί μετακινώντας το *x* κατά μικρά βήματα ε με το αντίθετο πρόσημο της παραμέτρου υπολογίζοντας την παράγωγο σε κάθε βήμα έως ότου $f'(x) = 0$. Το μέγεθος του βήματος στην ΜΜ ονομάζεται **ρυθμός μάθησης (learning rate)** και είναι υπερπαράμετρος του συστήματος. Αυτή η τεχνική για την βελτιστοποίηση ονομάζεται **κάθοδος με βάση την κλίση (gradient descent)**, όπως φαίνεται στο παράδειγμα της Εικόνας 4-14 για την παραβολική συνάρτηση $f = \frac{1}{2}x^2$, η οποία είναι μια κυρτή συνάρτηση.



Εικόνα 4-14. Παράδειγμα gradient descent
Πηγή: I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016

Γενικά για μια συνεχή συνάρτηση $f(x)$, όταν $f'(x) = 0$, η παράγωγος δεν δίνει πληροφορία για την κατεύθυνση της κίνησης. Τα σημεία αυτά ονομάζονται κρίσιμα σημεία (critical points). Το τοπικό ελάχιστο (local minimum) είναι το σημείο όπου η $f(x)$ παίρνει τιμές μικρότερες από τα αυτές των γειτονικών σημείων, οπότε δεν μπορεί η τιμή της να μειωθεί περισσότερο κάνοντας μικρά βήματα. Το τοπικό μέγιστο (local maximum) είναι το σημείο όπου η $f(x)$ παίρνει τιμές μεγαλύτερες από τα αυτές των γειτονικών σημείων, οπότε δεν μπορεί η τιμή της να αυξηθεί περισσότερο κάνοντας μικρά βήματα. Τα σημεία που δεν παρουσιάζουν ούτε ελάχιστα, ούτε μέγιστα λέγονται σαγματικά σημεία (saddle points).

Συνήθως, έχουμε να βελτιστοποιήσουμε πιο σύνθετες συναρτήσεις που η μορφή τους δεν είναι τόσο απλή όσο η παραβολική και παρουσιάζουν τοπικά και ολικά ελάχιστα και μέγιστα, και σαγματικά σημεία, όπως φαίνεται στην Εικόνα 4-15 , όπου $f'(x) = 0$, οπότε η βελτιστοποίηση είναι δύσκολη, ειδικά όταν η είσοδος είναι πολυδιάστατη. Ετσι, συνήθως συμβιβαζόμαστε στην εύρεση μιας τιμής της f η οποία είναι πολύ χαμηλή αλλά όχι απαραίτητα η ελάχιστη δυνατή [1].



Εικόνα 4-15. Κρίσιμα σημεία συνάρτησης

Στην πράξη, στόχος είναι η ελαχιστοποίηση συναρτήσεων $f(\boldsymbol{x})$, πολλών μεταβλητών $\boldsymbol{x}$ =$x_1,..x_n$, δηλαδή μιας σύνθετης συνάρτησης. Για να έχει νόημα η βελτιστοποίηση, θα πρέπει να υπάρχει μία μόνο βαθμωτή έξοδος. Για συναρτήσεις με πολλές μεταβλητές, όπως είναι οι συναρτήσεις κόστους, χρησιμοποιείται η έννοια της μερικής παραγώγου $\frac{\partial}{\partial x_i} f(\boldsymbol{x})$ όπου υπολογίζεται πως η f αλλάζει όταν αλλάζει μόνο η μεταβλητή x_i στο $\boldsymbol{x}$. Η όρος κλίση-gradient[13] γενικεύει την έννοια της παραγώγου σε σχέση με ένα διάνυσμα $\boldsymbol{x}$**:** Η gradient της f είναι ένα διάνυσμα που περιέχει όλες τις μερικές παραγώγους των στοιχείων του $\boldsymbol{x}$ και συμβολίζεται με $\nabla_x f(\boldsymbol{x})$. Το στοιχείο *i* της gradient είναι η μερική παράγωγος της f σε σχέση με το x_i [1].

Η κατευθυντική παράγωγος (directional derivative) της f σε μια διανυσματική διεύθυνση $\boldsymbol{u}$ είναι η κλίση της συνάρτησης στην κατεύθυνση $\boldsymbol{u}$. Χωρίς να μπούμε σε ιδιαίτερες λεπτομέρειες όσον αφορά τα μαθηματικά της κατευθυντικής παραγώγου, εάν θέλουμε να ελαχιστοποιήσουμε την συνάρτηση f, πρέπει να βρούμε τη διεύθυνση προς την οποία η f μειώνεται και αυτό γίνεται με τη βοήθεια της κατευθυντικής παραγώγου. Η f ελαχιστοποιείται όταν η $\boldsymbol{u}$ δείχνει σε αντίθετη κατεύθυνση της gradient. Συνεπώς, μειώνουμε την f με κίνηση προς την αρνητική gradient και αυτή η μέθοδος είναι η gradient descent. Με τη μέθοδο αυτή παίρνουμε ένα νέο σημείο:

$$\boldsymbol{x}' = \boldsymbol{x} - \varepsilon \nabla_{\boldsymbol{x}} f(\boldsymbol{x}) \qquad (4.7.1)$$

[13] Χρησιμοποιούμε τον αγγλικό όρο gradient αντί για τον όρο κλίση για τον διαχωρισμό από την κλίση slope

όπου ε είναι ο ρυθμός μάθησης, το θετικό βαθμωτό μέγεθος που ορίζει το μέγεθος του βήματος και αποτελεί υπερπαράμετρο. Η gradient descent συγκλίνει όταν κάθε στοιχείο της gradient είναι μηδέν ή κοντά στο μηδέν.

Όταν ο αλγόριθμος εφαρμόζεται σε όλο το σύνολο των παραδειγμάτων της εκπαίδευσης συγχρόνως, τότε ονομάζεται **batch gradient descent** [33].

4.7.1.1 Παράδειγμα Εφαρμογής Batch Gradient Descent

Για παράδειγμα, ας δούμε την εφαρμογή της batch gradient descent για τη βελτιστοποίηση της συνάρτησης κόστους δυαδικής εγκάρσιας εντροπίας για παραδείγματα με *n* χαρακτηριστικά και παραμέτρους $\boldsymbol{\theta}$ (βάρη και πολώσεις) που για ένα σύνολο εκπαίδευσης *m* παραδειγμάτων δίνεται από τη σχέση:

$$\mathrm{C}(\boldsymbol{\theta}) = -\frac{1}{\mathrm{m}}\left[\sum_{\mathrm{i}=0}^{\mathrm{m}} y^{\mathrm{i}} \log\left(p^{\mathrm{i}}\right) + \left(1 - y^{\mathrm{i}}\right) \log\left(1 - p^{\mathrm{i}}\right)\right] \qquad (4.7.2)$$

Αυτή η συνάρτηση είναι κυρτή, οπότε η gradient descent εγγυάται ότι θα βρεθεί ένα τοπικό ελάχιστο.

Η μερική παράγωγος, δηλαδή η κλίση της συνάρτησης κόστους *C* σε σχέση με κάθε θ_j δίνεται από την εξίσωση:

$$\frac{\partial}{\partial \theta_{\mathrm{j}}} \mathrm{C}(\boldsymbol{\theta}) = \frac{1}{\mathrm{m}} \sum_{\mathrm{i}=1}^{\mathrm{m}} (p^{\mathrm{i}} - y^{\mathrm{i}}) \cdot x_{\mathrm{j}}{}^{\mathrm{i}} \ (\text{για j} = 0,1,2,..,n) \qquad (4.7.3)$$

Εφόσον οριστεί η *C* σε σχέση με κάθε θ_j, ενημερώνεται η τιμή θ_j σύμφωνα με τη εξίσωση

$$\theta_j^{(next\ step)} = \theta_j - \varepsilon \frac{\partial C}{\partial \theta_j} \qquad (4.7.4)$$

όπου ο ε ο ρυθμός μάθησης.

Σε διανυσματική μορφή, η εξίσωση είναι:

$$\boldsymbol{\theta}^{(next\ step)} = \boldsymbol{\theta} - \varepsilon \nabla_{\boldsymbol{\theta}} C(\boldsymbol{\theta}) \qquad (4.7.5)$$

Γενικά, ο αλγόριθμος μπορεί να περιγραφεί με τα παρακάτω βήματα:

1. *Τυχαία αρχικοποίηση του πίνακα των παραμέτρων*
2. *Επανάληψη μέχρι τη σύγκλιση:*

3. Υπολογισμός καθόδου με βάση την κλίση $\frac{\partial}{\partial \theta_j} C(\theta)$ ή $\nabla_{\boldsymbol{\theta}} C(\boldsymbol{\theta})$

4. Ενημέρωση πίνακα παραμέτρων $\theta_j^{(next\ step)} = \theta_j - \varepsilon \frac{\partial C}{\partial \theta_j}$ ή

$$\boldsymbol{\theta}^{(next\ step)} = \boldsymbol{\theta} - \varepsilon \nabla_{\boldsymbol{\theta}} C(\boldsymbol{\theta})$$

5. Επιστροφή των τιμών των παραμέτρων

4.7.2 Stochastic Gradient Descent και Mini-batch Gradient Descent

Τα τελευταία χρόνια, τα μεγέθη των δεδομένων μεγαλώνουν πιο γρήγορα από ότι η ταχύτητα των επεξεργαστών. Σε αυτό το πλαίσιο, οι δυνατότητες των μεθόδων περιορίζονται από τον χρόνο υπολογισμού και όχι από το μέγεθος του δείγματος [34].

Το κύριο πρόβλημα της batch gradient descent είναι ότι ο υπολογισμός των κλίσεων για ένα πολύ μεγάλο σύνολο δεδομένων εκπαίδευσης είναι πολύ αργός έστω και εάν δίνει μεγαλύτερη ακρίβεια.

Η **στοχαστική κάθοδος με βάση την κλίση (Stochastic Gradient Descent -SGD)** απλά διαλέγει τυχαία (στοχαστικά) ένα τυχαίο στιγμιότυπο έστω *i* του συνόλου εκπαίδευσης και υπολογίζει τις κλίσεις για μια δέσμη με βάση μόνο αυτό το στιγμιότυπο [33]:

$$\boldsymbol{\theta}^{(next\ step)} = \boldsymbol{\theta} - \varepsilon \nabla_{\theta} C(\boldsymbol{\theta}; x^{(i)}; y^{(i)}) \qquad (4.7.6)$$

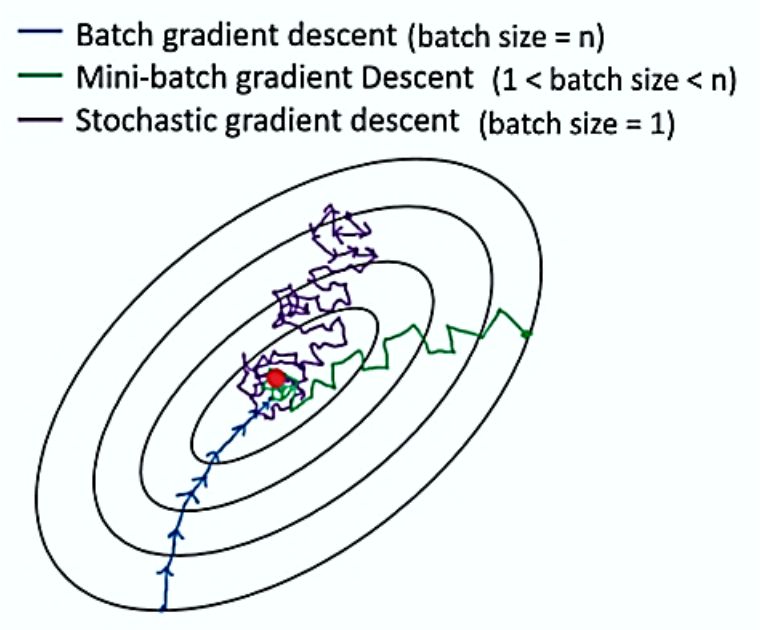
Προφανώς, αυτό καθιστά τον αλγόριθμο πιο γρήγορο καθώς έχει να χειριστεί λιγότερα δεδομένα σε κάθε επανάληψη. Επίσης, καθιστά δυνατή την εκπαίδευση τεράστιων συνόλων δεδομένων, καθώς μόνο ένα στιγμιότυπο χρειάζεται να είναι στη μνήμη για κάθε επανάληψη.

Μια παραλλαγή της batch gradient descent είναι η **mini-batch gradient descent**, όπου για τον υπολογισμό λαμβάνονται τυχαία μικρές ομάδες- δέσμες (mini-batches) μεγέθους *n* του συνόλου εκπαίδευσης [33]:

$$\boldsymbol{\theta}^{(next\ step)} = \boldsymbol{\theta} - \varepsilon \cdot \nabla_{\theta} C(\boldsymbol{\theta}; x^{(i:i+n)}; y^{(i:i+n)}) \qquad (4.7.8)$$

Θα μπορούσαμε να πούμε ότι, είναι μια ενδιάμεση κατάσταση όσον αφορά τον χρόνο υπολογισμού και την ακρίβεια.

Η συγκριτική απεικόνιση των τριών παραλλαγών της gradient descent όσον αφορά τη διαδρομή για τη σύγκλιση φαίνεται στην Εικόνα 4-16.



Εικόνα 4-16. Σύγκριση Batch, Stochastic και Mini-batch Gradient Descent

Πηγή: https://miro.medium.com/max/946/1*OwX5ky1lqycOIH2LiwSCyQ.png

4.7.3 Σύγχρονοι Αλγόριθμοι Βελτιστοποίησης

Ο αλγόριθμος gradient descent και οι παραλλαγές του, εμφάνισαν διάφορες προκλήσεις που έπρεπε να αντιμετωπιστούν στις εφαρμογές των σύγχρονων ΤΝΔ, όπως η επιλογή του κατάλληλου ρυθμού μάθησης, η εφαρμογή σε μη-κυρτές συναρτήσεις κόστους, κλπ. [33], με αποτέλεσμα να προταθούν τα τελευταία χρόνια πολλοί νέοι αλγόριθμοι βελτιστοποίησης [33], [35]. Ενδεικτικά αναφέρονται οι:

- Momentum
- Nesterov accelerated gradient
- Adagrad
- Adadelta
- RMSprop
- Adam
- AdaMax
- Nadam
- AMSGrad

και οι νεότεροι [36]:

- AdamW
- QHAdam
- YellowFin
- AggMo
- QHM
- Demon

Για περισσότερες πληροφορίες ο αναγνώστης παραπέμπεται στις πηγές.

Η απάντηση στο ερώτημα «ποιος αλγόριθμος πρέπει να επιλεγεί για μια συγκεκριμένη εργασία» δεν υπάρχει. Είναι μια υπερπαράμετρος και πρέπει κάθε φορά να γίνονται δοκιμές, ανάλογα βέβαια με το μέγεθος του συνόλου δεδομένων και με τους διαθέσιμους πόρους του χρήστη σε υλικό και λογισμικό.

4.8 Οι Αλγόριθμοι Εμπροσθοδιάδοσης και Οπισθοδιάδοσης

Ο κύριος λόγος που τα ΤΝΔ οργανώνονται σε επίπεδα είναι γιατί η δομή αυτή καθιστά πιο απλούς και αποτελεσματικούς τους υπολογισμούς με τη χρήση πράξεων μεταξύ διανυσμάτων και πινάκων. Για παράδειγμα, η είσοδος του δικτύου για ένα στιγμιότυπο με n χαρακτηριστικά μπορεί να αναπαρασταθεί ως ένα διάνυσμα $\boldsymbol{x}$ διαστάσεων [n x1], τα συναπτικά βάρη προς το πρώτο κρυφό επίπεδο m νευρώνων ως ένας πίνακας (matrix) $\boldsymbol{w}$ διαστάσεων [m x n] και οι πολώσεις ως ένα διάνυσμα $\boldsymbol{b}$ [m x 1].

Πριν προχωρήσουμε, θα αναφερθούμε σε έναν συχνά χρησιμοποιούμενο όρο, τον όρο του **τανυστή (tensor)**. Ο τανυστής έχει την έννοια του πίνακα (array) σε ένα κατάλληλο αριθμό διαστάσεων και η διάσταση καλείται άξονας (axis)[14]. Για παράδειγμα, το μητρώο (matrix) που είναι ένας δισδιάστατος πίνακας, είναι ένας δισδιάστατος τανυστής. Γενικότερα για τους τανυστές ισχύει ότι [23]:

- Ο τανυστής που περιέχει μόνο έναν αριθμό ονομάζεται βαθμωτός (scalar) ή 0-διάστατος (dimensional) ή 0D τανυστής.
- Ο μονοδιάστατος πίνακας αριθμών είναι το διάνυσμα (vector) ή 1D τανυστής και έχει έναν άξονα.

[14] Οι όροι τανυστής και πίνακας (array) χρησιμοποιούνται εναλλακτικά

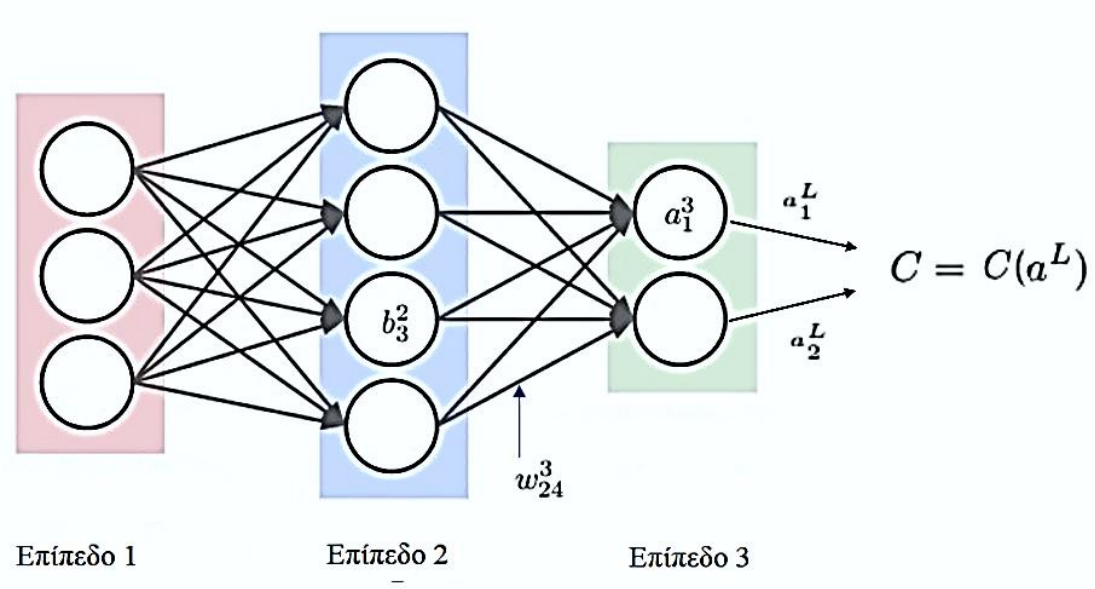
- Ο δισδιάστατος πίνακας είναι το μητρώο (matrix) ή 2D τανυστής. Το μητρώο έχει δύο άξονες, τις γραμμές και τις στήλες, είναι δηλαδή ένα ορθογώνιο πλέγμα αριθμών.
- Η συλλογή των μητρώων σε ένα νέο πίνακα δίνει έναν τρισδιάστατο πίνακα, δηλαδή 3D τανυστή, η συλλογή των 3D τανυστών δίνει έναν 4D τανυστή και ούτω καθεξής[15]

Έτσι, όλοι οι υπολογισμοί σε κάθε επίπεδο μπορούν να αναπαρασταθούν με πράξεις τανυστών, με κατάλληλες παραδοχές, όπως περιγράφεται στη συνέχεια.

Πριν την εξήγηση των αλγορίθμων και των υπολογισμών, παρατίθενται πρώτα οι απαραίτητοι συμβολισμοί, με παραδείγματα στο δίκτυο τριών επιπέδων της Εικόνας 4-17.

- L είναι ο αριθμός επιπέδων του δικτύου
- w_{jk}^{l} είναι το βάρος από τον κ νευρώνα στο $l-1$ επίπεδο προς τον j νευρώνα. Για παράδειγμα, στο δίκτυο τριών επιπέδων ($l=3$) της Εικόνας ο συμβολισμός του βάρους για τον 4ο νευρώνα στον 2ο νευρώνα του τρίτου επιπέδου είναι w_{24}^{3}.
- b_j^l είναι η πόλωση του j νευρώνα στο l επίπεδο. Για παράδειγμα, ο συμβολισμός για την πόλωση του 3ου νευρώνα στο 2ο επίπεδο είναι b_3^2.
- a_j^l είναι η ενεργοποίηση του j νευρώνα στο l επίπεδο. Για παράδειγμα, ο συμβολισμός για την ενεργοποίηση του 1ου νευρώνα του 3ο επίπεδου είναι α_1^3.
- η συνάρτηση ενεργοποίησης συμβολίζεται με σ.
- $\boldsymbol{a}^L = a^L(x)$ είναι το διάνυσμα των ενεργοποιήσεων εξόδου από το δίκτυο για ένα διάνυσμα εισόδου $\boldsymbol{x}$
- $\boldsymbol{\delta}^L = \boldsymbol{a}^L - y_{pred}$ είναι το σφάλμα στο επίπεδο εξόδου
- C είναι η συνάρτηση κόστους για ένα διάνυσμα εισόδου $\boldsymbol{x}$ με y ετικέτα-στόχο

[15] Στην ΒΜ διαχειριζόμαστε 0D έως 4D τανυστές και 5D για δεδομένα video



Εικόνα 4-17. Παράδειγμα συμβολισμών παραμέτρων και τιμών υπολογισμού για MLP τριών επιπέδων

4.8.1 Ο Αλγόριθμος Εμπροσθοδιάδοσης (Forward Propagation Algorithm)

Το πρώτο βήμα των υπολογισμών της εκπαίδευσης αφορά τον υπολογισμό της ενεργοποίησης ενός νευρώνα, με σκοπό τον υπολογισμό της τιμής ως πρόβλεψη y_{pred} προχωρώντας προς τα εμπρός διαμέσου κάθε επιπέδου l προς την έξοδο L. Η διαδικασία των υπολογισμών γίνεται με τον **αλγόριθμο εμπροσθοδιάδοσης**, που περιγράφεται παρακάτω [26], [37].

Η ενεργοποίηση a_j^l για τον j νευρώνα στο l επίπεδο σχετίζεται με τις ενεργοποιήσεις του $l-1$ επίπεδου σύμφωνα με την εξίσωση

$$a_j^l = \sigma(\sum_k w_{jk}^l a_k^{l-1} + b_j^l) \qquad (4.8.1)$$

όπου το άθροισμα αφορά όλους τους k νευρώνες του $l-1$ επίπεδου.

Για την μετατροπή της παραπάνω συνάρτησης σ σε μορφή πίνακα ορίζουμε το μητρώο των βαρών $\boldsymbol{w^l}$ για κάθε επίπεδο l. Οι τιμές του μητρώου $\boldsymbol{w^l}$ είναι είναι τα βάρη που συνδέονται με το l επίπεδο, δηλαδή η τιμή στη σειρά j και στη στήλη κ είναι w_{jk}^l. Παρόμοια, για κάθε επίπεδο ορίζουμε το διάνυσμα της πόλωσης $\boldsymbol{b^l}$, όπου τα στοιχεία του είναι οι πολώσεις b_j^l και το διάνυσμα της ενεργοποίησης του επιπέδου $\boldsymbol{a^l}$ όπου τα στοιχεία του είναι οι ενεργοποιήσεις a_j^l. Τέλος, για να μετατραπεί η εξίσωση σε μορφή διανυσμάτων, πρέπει να μετατραπεί γίνει η διαδικασία της διανυσματοποίησης (vectorization). Η διανυσματοποίηση είναι ένας γραμμικός μετασχηματισμός

όπου ένας πίνακας μετατρέπεται σε διάνυσμα με την εφαρμογή μιας συνάρτησης. Οπότε η εξίσωση με τη μορφή πινάκων είναι:

$$\boldsymbol{a^l} = \sigma(\boldsymbol{w^l a^{l-1}} + \boldsymbol{b^l}) \qquad (4.8.2)$$

Η εξίσωση 4.8.2 δίνει έναν γενικότερο τρόπο για το πώς σχετίζονται οι ενεργοποιήσεις σε ένα επίπεδο με τις ενεργοποιήσεις στο προηγούμενο επίπεδο: εφαρμόζεται ο πίνακας βαρών στις ενεργοποιήσεις, προστίθεται το διάνυσμα των πολώσεων και τέλος, εφαρμόζεται η συνάρτηση ενεργοποίησης σ. Ο πίνακας $\boldsymbol{z^l} = \boldsymbol{w^l a^{l-1}} + \boldsymbol{b^l}$ είναι η **σταθμισμένη είσοδος (weighted input)** στους νευρώνες του επιπέδου *l*. Ας σημειωθεί ότι ο πίνακας $\boldsymbol{z^l}$ περιλαμβάνει τα στοιχεία $z_j^l = \sum_k w_{jk}^l a_k^{l-1} + b_j^l$, πράγμα που σημαίνει ότι το z_j^l είναι η σταθμισμένη είσοδος στη συνάρτηση ενεργοποίησης για τον νευρώνα *j* στο επίπεδο *l*.

4.8.2 Ο Αλγόριθμος Οπισθοδιάδοσης (Backpropagation Algorithm)

Βασικός στόχος της εκπαίδευσης είναι η βελτιστοποίηση του δικτύου και η διαδικασία πραγματοποιείται με τον **αλγόριθμο οπισθοδιάδοσης** (backpropagation algorithm). Στόχος του αλγόριθμου είναι η επαναληπτική ενημέρωση των παραμέτρων του δικτύου, δηλαδή των πινάκων των βαρών και των πολώσεων προκειμένου να ελαχιστοποιηθεί η συνάρτηση κόστους. Άρα, πρέπει να υπολογιστεί το σφάλμα εξόδου δ^L, με βάση τη συνάρτηση κόστους *C* που έχει οριστεί, και να ελαχιστοποιηθεί η συνάρτηση κόστους με τον αλγόριθμο gradient descent [1], [23], [26], [37].

Πριν προχωρήσουμε στην περιγραφή της διαδικασίας, αναφέρουμε τον **κανόνα της αλυσίδας (chain rule)** του μαθηματικού λογισμού. Ο κανόνας αυτός χρησιμοποιείται για τον υπολογισμό των συναρτήσεων που συνθέτονται από άλλες συναρτήσεις που είναι γνωστές οι παράγωγοί τους. Ο αλγόριθμος οπισθοδιάδοσης είναι αλγόριθμος που υπολογίζει τον κανόνα της αλυσίδας, με συγκεκριμένη σειρά λειτουργιών, πολύ αποτελεσματικά.

Ποιος όμως είναι ο κανόνας της αλυσίδας. Ας υποθέσουμε ότι x πραγματικός αριθμός και f, g είναι δύο συναρτήσεις που αντιστοιχίζουν από πραγματικό αριθμό σε πραγματικό και ότι $y = g(x)$ και $z = f\big(g(x)\big) = f(y)$. Τότε, σύμφωνα με τον κανόνα της αλυσίδας ισχύει:

$$\frac{dz}{dx} = \frac{dz}{dy}\frac{dy}{dx} \qquad (4.8.3)$$

Γενικεύοντας πέρα από βαθμωτά μεγέθη, και θεωρώντας ότι $\boldsymbol{x} \in \mathbb{R}^m$, $\boldsymbol{y} \in \mathbb{R}^n$, η y αντιστοιχίζει από το $\mathbb{R}^m$ στο $\mathbb{R}^n$ και η f από το $\mathbb{R}^n$ στο $\mathbb{R}$. Εάν $\boldsymbol{y} = g(\boldsymbol{x})$ και $\boldsymbol{z} = f(\boldsymbol{y})$ τότε:

$$\frac{\partial z}{\partial x_i} = \sum_j \frac{\partial z}{\partial y_i}\frac{\partial y_i}{\partial x_i} \qquad (4.8.4)$$

Σε διανυσματική μορφή η παραπάνω σχέση γράφεται:

$$\nabla_x z = \left(\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}}\right)^T \nabla_y z \qquad (4.8.5)$$

όπου $\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}}$ είναι το n x m ιακωβιανό μητρώο (jacobian matrix) του $\boldsymbol{y}$.

Από τη παραπάνω εξίσωση προκύπτει ότι η gradient μιας μεταβλητής $\boldsymbol{x}$ εξάγεται πολλαπλασιάζοντας το ιακωβιανό μητρώο $\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}}$ με την gradient $\nabla_y z$. Ο αλγόριθμος οπισθοδιάδοσης συνίσταται από την εκτέλεση ενός γινομένου ιακωβιανού-gradient για κάθε υπολογισμό στον γράφο του δικτύου. Ο αλγόριθμος δεν εφαρμόζεται μόνο για διανύσματα, αλλά γενικότερα για τανυστές, οπότε με κατάλληλη διαδικασία γίνεται η διανυσματοποίηση των τανυστών, υπολογίζεται η gradient ως διάνυσμα τιμών και στη συνέχεια μετασχηματίζεται η gradient σε τανυστή.

Επιστρέφοντας στη διαδικασία, η οπισθοδιάδοση αρχικά, αφορά το πώς η αλλαγή των βαρών και των πολώσεων σε ένα δίκτυο αλλάζει τη συνάρτηση κόστους. Γενικά, οι πίνακες βαρών για ρυθμό εκμάθησης ε ενημερώνονται σύμφωνα με τη σχέση:

$$\boldsymbol{w} \quad \rightarrow \quad \boldsymbol{w} - \varepsilon \frac{\partial C}{\partial \boldsymbol{w}}$$

και οι πολώσεις σύμφωνα με τη σχέση:

$$\boldsymbol{b} \quad \rightarrow \quad \boldsymbol{b} - \varepsilon \frac{\partial C}{\partial \boldsymbol{b}}$$

Αυτό σημαίνει πως, πρέπει να υπολογιστούν, σύμφωνα με τους συμβολισμούς που δώσαμε για το δίκτυο, οι μερικές παράγωγοι $\partial C/\partial w_{jk}^{l}$ και $\partial C/\partial b_{j}^{l}$. Το ερώτημα που τίθεται είναι γιατί πρέπει να υπολογιστούν αυτές οι παράγωγοι και πρέπει να έχουμε μια διαίσθηση.

Ας υποθέσουμε ότι γίνεται μια μικρή αλλαγή Δw_{jk}^{l} σε κάποιο βάρος w_{jk}^{l}. Αυτή η αλλαγή θα επηρρεάσει την ενεργοποίηση εξόδου του σχετικού νευρώνα κατά Δa_{j}^{l} και θα προκαλέσει αλλαγή σε όλες τις ενεργοποιήσεις του επόμενου επιπέδου, που με τη σειρά τους θα προκαλέσουν αλλαγές στα επόμενα επίπεδα μέχρι το τελευταίο και συνεπώς στη συνάρτηση κόστους κατά ΔC η οποία είναι σχετική με την αλλαγή Δw_{jk}^{l} στο σχετικό βάρος και εκφράζεται από την εξίσωση

$$\Delta C \approx \frac{\partial C}{\partial w_{jk}^{l}} \Delta w_{jk}^{l} \qquad (4.8.6)$$

όπου ο υπολογισμός του $\frac{\partial C}{\partial w_{jk}^{l}}$ έχει την έννοια της ιχνηλάτησης κατά πόσο μια μικρή αλλαγή στο w_{jk}^{l} διαδίδεται για να προκαλέσει μια μικρή αλλαγή στην C. Έτσι, η αλλαγή Δa_{j}^{l} που θα προκαλέσει αλλαγές σε όλες τις ενεργοποιήσεις του επόμενου επίπεδου θα είναι:

$$\Delta a_{j}^{l} \approx \frac{\partial a_{j}^{l}}{\partial w_{jk}^{l}} \Delta w_{jk}^{l} \qquad (4.8.7)$$

στο επόμενο επίπεδο οι αλλαγές της ενεργοποίησης θα είναι:

$$\Delta a_{q}^{l+1} \approx \frac{\partial a_{q}^{l+1}}{\partial a_{j}^{l}} \Delta a_{j}^{l} \approx \frac{\partial a_{q}^{l+1}}{\partial a_{j}^{l}} \frac{\partial a_{j}^{l}}{\partial w_{jk}^{l}} \Delta w_{jk}^{l} \qquad (4.8.9)$$

που θα προκαλέσει αλλαγές στο επόμενο επίπεδο κ.λπ. Στην πραγματικότητα είναι ένα μονοπάτι από το βάρος w_{jk}^{l} στη συνάρτηση κόστους C με αλλαγές στις ενεργοποιήσεις. το μονοπάτι περνάει από τις ενεργοποιήσεις $a_{j}^{l}, a_{q}^{l+1}, \ldots, a_{n}^{L-1}, a_{m}^{L}$ και τελικά υπολογίζεται η συνολική αλλαγή της C προσθέτοντας όλα τα πιθανά μονοπάτια μεταξύ του βάρους και του τελικού κόστους:

$$\frac{\partial C}{\partial w_{jk}^{l}} = \sum_{mnp\ldots q} \frac{\partial C}{\partial a_{m}^{L}} \frac{\partial a_{m}^{L}}{\partial a_{n}^{L-1}} \frac{\partial a_{n}^{L-1}}{\partial a_{p}^{L-2}} \cdots \frac{\partial a_{q}^{l+1}}{\partial a_{j}^{l}} \frac{\partial a_{j}^{l}}{\partial w_{jk}^{l}} \qquad (4.8.10)$$

Για να υπολογιστούν οι $\partial C / \partial w_{jk}^{l}$ και $\partial C / \partial b_{j}^{l}$, θα πρέπει να εισαχθεί μια ενδιάμεση ποσότητα δ_{j}^{l} η οποία υπολογίζει το λάθος στον j νευρώνα του l επιπέδου και δίνεται από τη σχέση:

$$\delta_{j}^{l} = \frac{\partial C}{\partial z_{j}^{l}} \qquad (4.8.11)$$

Με βάση τα παραπάνω, η αλγεβρική σχέση για τον υπολογισμό του σφάλματος στο επίπεδο εξόδου δ^{L} είναι:

$$\delta_{j}^{L} = \frac{\partial C}{\partial a_{j}^{L}} \sigma'(z_{j}^{L}) \qquad (4.8.12)$$

Ο πρώτος όρος $\partial C / \partial a_{j}^{L}$ μετρά πόσο γρήγορα αλλάζει το κόστος C ως συνάρτηση της $j^{οστής}$ ενεργοποίησης εξόδου. Για παράδειγμα, εάν το C δεν εξαρτάται πολύ από έναν συγκεκριμένο νευρώνα j, τότε το δ_{j}^{L} θα είναι μικρό. Ο δεύτερος όρος $\sigma'(z_{j}^{L})$, μετρά το πόσο γρήγορα στη συνάρτηση ενεργοποίησης σ αλλάζει στο z_{j}^{L}.

Μια γνωστή πράξη της γραμμικής άλγεβρας είναι το γινόμενο πινάκων στοιχείο- προς- στοιχείο (element-wise product) ή αλλιώς γινόμενο Hadamard που συμβολίζεται με $\odot$.

Σε αυτή την πράξη λαμβάνονται δύο διανύσματα ίδιων διαστάσεων και παράγεται ένα διάνυσμα ίδιου μήκους του οποίου το κάθε στοιχείο είναι το γινόμενο των αντίστοιχων αρχικών διανυσμάτων. Για παράδειγμα, το γινόμενο Hadamard δύο απλών διανυσμάτων υπολογίζεται ως εξής:

$$\begin{bmatrix} 1 \\ 2 \end{bmatrix} \odot \begin{bmatrix} 3 \\ 4 \end{bmatrix} = \begin{bmatrix} 1 * 3 \\ 2 * 4 \end{bmatrix} = \begin{bmatrix} 3 \\ 8 \end{bmatrix}$$

Η εξίσωση 4.8.12 γράφεται υπό μορφή γινομένου Hadamard ως:

$$\boldsymbol{\delta}^{L} = \nabla_{a} \boldsymbol{C} \odot \sigma'(\boldsymbol{z}^{\boldsymbol{L}}) \qquad (4.8.13)$$

Ο όρος $\nabla_{a} \boldsymbol{C}$ είναι ένα διάνυσμα του οποίου οι συνιστώσες είναι οι μερικές παράγωγοι $\partial C / \partial a_{j}^{L}$ και εκφράζει τον ρυθμό (rate) της αλλαγής της συνάρτησης κόστους C σε σχέση με την ενεργοποίση της εξόδου.

Στη συνέχεια πρέπει να υπολογιστεί το σφάλμα δ^{l} σε σχέση με το σφάλμα στο επόμενο επίπεδο δ^{l+1} και δίνεται από τη σχέση:

$$\delta^l = ((\boldsymbol{w^{l+1}})^T \delta^{l+1}) \odot \sigma'(\boldsymbol{z}^l) \qquad (4.8.14)$$

όπου $(\boldsymbol{w}^{l+1})^T$ το ανάστροφο μητρώο του πίνακα βαρών $\boldsymbol{w^{l+1}}$ για το επίπεδο $(l+1)$. Η εξίσωση 4.8.14 έχει μια απλή εξήγηση. Έστω ότι είναι γνωστό το δ^{l+1}. Εφαρμόζοντας το ανάστροφο μητρώο, μετατοπίζεται το σφάλμα προς τα πίσω στο δίκτυο, δίνοντας έτσι ένα μέτρο για το σφάλμα εξόδου στο επίπεδο l. Στη συνέχεια παίρνουμε το γινόμενο Hadamard $\odot \sigma'(z^l)$, το οποίο μετακινεί το σφάλμα προς τα πίσω μέσω της συνάρτησης ενεργοποίησης στο επίπεδο l, δίνοντας το σφάλμα δ^l στα βάρη εισόδου στο επίπεδο l.

Συνδυάζοντας τις εξισώσεις 4.8.13 και 4.8.14 είναι δυνατό να υπολογιστεί το σφάλμα δ^l για κάθε επίπεδο του δικτύου. Στην αρχή υπολογίζεται το δ^L από την εξίσωση 4.8.13, στη συνέχεια το δ^{L-1} από την εξίσωση 4.8.14, το δ^{L-2} από την ίδια εξίσωση και ούτω καθεξής για όλη τη διαδρομή προς τα πίσω μέσω του δικτύου.

Επίσης, πρέπει να ληφθεί υπόψη ο ρυθμός της αλλαγής του κόστους σε σχέση με τις πολώσεις στο δίκτυο, η οποία δίνεται από τη σχέση:

$$\frac{\partial C}{\partial b_j^l} = \delta_j^l \qquad (4.8.15)$$

Αυτό σημαίνει ότι, το σφάλμα δ_j^l είναι ίσο με τον ρυθμό της αλλαγής $\partial C / \partial b_j^l$, το οποίο μπορεί να υπολογιστεί όπως τις εξισώσεις 4.8.13 και 4.8.14.

Τέλος, υπολογίζεται ο ρυθμός αλλαγής του κόστους σε σχέση με κάθε βάρος στο δίκτυο. Πιο συγκεκριμένα:

$$\frac{\partial C}{\partial w_{jk}^l} = a_k^{l-1} \delta_j^l \qquad (4.8.16)$$

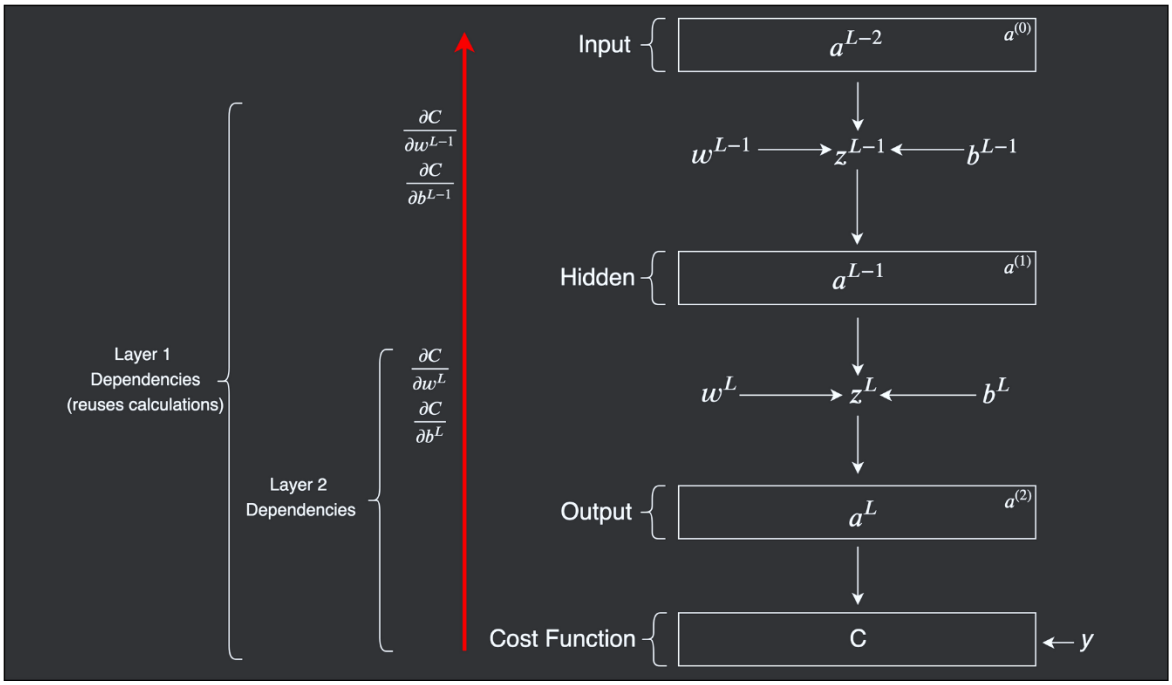
Η εξίσωση 4.8.16 δίνει τον τρόπο υπολογισμού των παραγώγων $\partial C / \partial w_{jk}^l$ με βάση τις ποσότητες $\delta^l και\, \alpha^{\,l-1}$ οι οποίες είναι γνωστό πως υπολογίζονται. Η εξίσωση 4.8.16 μπορεί να γραφεί με έναν πιο απλό τρόπο χωρίς δείκτες:

$$\frac{\partial C}{\partial w} = a_{\text{in}} \delta_{\text{out}} \qquad (4.8.17)$$

από όπου γίνεται κατανοητό ότι a_{in} είναι η ενεργοποίηση του νευρώνα εισόδου στο βάρος w και δ_{out} είναι το σφάλμα του νευρώνα εξόδου από το βάρος w.

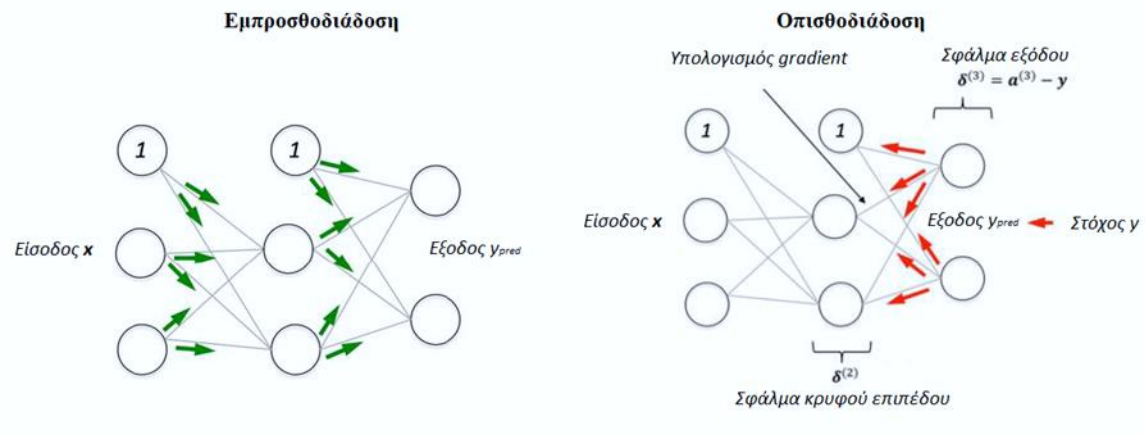
Αφού γίνει η ενημέρωση των βαρών και των πολώσεων, η διαδικασία επαναλαμβάνεται από την αρχή μέχρι το κριτήριο σύγκλισης.

Τέλος, παραθέτουμε δύο συνοπτικές και κατατοπιστικές εικόνες. Στην Εικόνα 4-18 φαίνεται με γραφικό τρόπο η εξάρτηση των μεταβλητών και των υπολογισμών της οπισθοδιάδοσης και στην Εικόνα 4-19 απεικονίζεται ενδεικτικά η κατεύθυνση της ροής υπολογισμών στον γράφο του ΤΝΔ.



Εικόνα 4-18. Γράφημα εξαρτήσεων μεταβλητών και υπολογισμών στην οπισθοδιάδοση

Πηγή: https://mlfromscratch.com/neural-networks-explained/



Εικόνα 4-19. Εμπροσθοδιάδοση – Σφάλμα εξόδου – Οπισθοδιάδοση.

4.9 Μέθοδοι Εξομάλυνσης

Όπως είδαμε στα βασικά της ΜΜ και πιο συγκεκριμένα στην παράγραφο 3.2.5, η εξομάλυνση (regularization) είναι κάθε τροποποίηση που γίνεται σε ένα μοντέλο μάθησης και που έχει ως σκοπό την μείωση του λάθους γενίκευσης και όχι του λάθους εκπαίδευσης και είναι μία από τις κύριες έννοιες στο πεδίο της ΜΜ που συναγωνίζεται την σπουδαιότητα της βελτιστοποίησης [1], [23] προκειμένου να καταπολεμηθεί το φαινόμενο του overfitting. Επίσης, αναφέρθηκε ότι, οι μέθοδοι εξομάλυνσης εξαρτώνται από τον αλγόριθμο της ΜΜ που έχει επιλεγεί για τη δημιουργία του μοντέλου.

Τα MLPs μπορεί να έχουν πολλαπλά μη-γραμμικά κρυφά επίπεδα και αυτό τα καθιστά ακριβά υπολογιστικά μοντέλα που μπορούν να μάθουν πολύ σύνθετες σχέσεις μεταξύ των εισόδων του και των εξόδων τους. Με περιορισμένα δεδομένα εκπαίδευσης πολλές από αυτές τις περίπλοκες σχέσεις έχουν ως αποτέλεσμα τον θόρυβο στη δειγματοληψία, οπότε θα υπάρχουν όχι μόνο στο σύνολο εκπαίδευσης, αλλά και στα νέα δεδομένα του συνόλου δοκιμής, ακόμα και εάν προέρχονται από την ίδια στατιστική κατανομή. Αυτό οδηγεί στο overfitting και έχουν αναπτυχθεί πολλές μέθοδοι για την αντιμετώπισή του του [38].

Αξίζει να σημειωθεί ότι, σύμφωνα με τους συγγραφείς στην [39], ως εξομάλυνση θεωρείται κάθε συμπληρωματική τεχνική που βοηθά στη γενίκευση του μοντέλου, δηλαδή στην παραγωγή καλύτερων αποτελεσμάτων από το σύνολο δοκιμής. Σε αυτή τη βάση, οι συγγραφείς εντάσσουν στις μεθόδους εξομάλυνσης τους σύγχρονους βελτιστοποιητές (όπως πχ όπως ο Adam), καθώς και την επιλογή κατάλληλων συναρτήσεων ενεργοποίησης στα κρυφά επίπεδα (όπως πχ οι σύγχρονες παραλλαγές της ReLU).

Γενικά όμως, για τα MLPs οι δημοφιλέστερες μέθοδοι εξομάλυνσης είναι [1], [22], [23]:

- Η μεταβολή της χωρητικότητας του δικτύου
- Η εξομάλυνση των βαρών
- Το πρόωρο σταμάτημα (early stopping)
- Το dropout
- Η κανονικοποίηση κατά δέσμες (batch normalization)

Στις επόμενες παραγράφους παρουσιάζονται οι κυριότερες μέθοδοι εξομάλυνσης συνοπτικά. Μια σύνοψη όλων των μεθόδων εξομάλυνσης δίνεται από τους συγγραφείς στην [39].

4.9.1 Εξομάλυνση με Μεταβολή της Χωρητικότητας

Η απλούστερη μέθοδος εξομάλυνσης είναι η μείωση της χωρητικότητας του μοντέλου. Πρακτικά αυτό σημαίνει μείωση των παραμέτρων εκμάθησης μειώνοντας τα κρυφά επίπεδα και τον αριθμό των νευρώνων των κρυφών επιπέδων.

Διαισθητικά, ένα μοντέλο με πολλές παραμέτρους εκμάθησης προσεγγίζει με μεγαλύτερη ακρίβεια την επιθυμητή συνάρτηση, αλλά η αύξηση της χωρητικότητας συνήθως είναι σε βάρος της γενίκευσης. Η συνιστώμενη πρακτική είναι να ξεκινάμε την σχεδίαση της αρχιτεκτονικής του μοντέλου με όσο το δυνατόν λιγότερα κρυφά επίπεδα και παραμέτρους και να αυξάνουμε την χωρητικότητα του δικτύου μέχρι να μειωθεί το λάθος επικύρωσης [1], [22], [23].

4.9.2 Εξομάλυνση των Βαρών

Από τον 14ο αιώνα, υπάρχει μια επιστημονική αρχή, η οποία αποτελεί τη βάση της μεθοδολογικής απαγωγής, η «λεπίδα του Occam (Occam's razor)[16] και εκφράζεται απλά ως «*Κανείς δεν θα πρέπει να προβαίνει σε περισσότερες εικασίες από όσες είναι απαραίτητες*». Δηλαδή, εάν έχουμε δύο εξηγήσεις για κάτι, η εξήγηση που φαίνεται να είναι πιο σωστή, αυτή είναι η πιο απλή – αυτή για την οποία γίνονται οι λιγότερες υποθέσεις. Αυτή η ιδέα εφαρμόζεται στα μοντέλα εκμάθησης των νευρωνικών δικτύων: με δοσμένα κάποια δεδομένα εκπαίδευσης και την αρχιτεκτονική του δικτύου, θα υπάρχουν πολλαπλά σύνολα με τιμές βαρών (πολλαπλά μοντέλα) που θα μπορούν να εξηγήσουν τα δεδομένα. Τα απλούστερα μοντέλα έχουν μικρότερη πιθανότητα για υπερπροσαρμογή από ότι τα πολύπλοκα μοντέλα [23].

Σε αυτό το πλαίσιο, ένα απλό μοντέλο είναι αυτό που η κατανομή των τιμών των παραμέτρων του έχει λιγότερη εντροπία, ή πιο απλά ένα μοντέλο με λιγότερες παραμέτρους. Έτσι, ένας συνηθισμένος τρόπος για την καταπολέμηση της υπερ προσαρμογής είναι να τεθούν περιορισμοί στην πολυπλοκότητα του δικτύου εξαναγκάζοντας τα βάρη του να πάρουν μόνο μικρές τιμές, κάτι το οποίο καθιστά την κατανομή των βαρών πιο ομαλή. Αυτή η μέθοδος ονομάζεται **εξομάλυνση των βαρών (weight regularization)** και επιτυγχάνεται με ενημέρωση της συνάρτησης κόστους που βελτιστοποιεί το δίκτυο κατά την εκπαίδευση λαμβάνοντας υπόψη το μέγεθος ή νόρμα των βαρών, η οποία αμέσως εξηγείται παρακάτω. Αυτό ονομάζεται και ποινή (penalty), καθώς όσο μεγαλώνουν τα βάρη, τόσο το δίκτυο «τιμωρείται» επειδή υπάρχουν μεγαλύτερες απώλειες και

[16] https://en.wikipedia.org/wiki/Occam%27s_razor

κατά συνέπεια περισσότερες ενημερώσεις των βαρών [1], [22]. Η εξομάλυνση αφορά μόνο τον πίνακα των βαρών και όχι των πολώσεων.

Ο υπολογισμός του μεγέθους ή του μήκους ενός διανύσματος χρειάζεται συχνά για πράξεις στην γραμμική άλγεβρα. Το μήκος του διανύσματος ονομάζεται **νόρμα του διανύσματος (vector norm)** ή **μέγεθος του διανύσματος (vector magnitude)**. Είναι ένας θετικός αριθμός και εκφράζει την επέκταση του διανύσματος στον χώρο.
Η L1 νόρμα υπολογίζει το άθροισμα των απόλυτων τιμών του διανύσματος και στην ουσία είναι ο υπολογισμός της απόστασης Manhattan από την πηγή του χώρου των διανυσμάτων. Για παράδειγμα, η πηγή του χώρου για ένα διάνυσμα με 3 στοιχεία είναι (0, 0, 0). Η L1 νόρμα ενός διανύσματος $\boldsymbol{x}$ συμβολίζεται ως $||x||_1$ και υπολογίζεται από τη σχέση:

$$||x||_1 = \sum_i |x_i| = |x_1| + |x_2| + \cdots + |x_i| \qquad (4.9.1)$$

Η L1 νόρμα υπολογίζει την απόσταση του διανύσματος από την πηγή και αυτή είναι η Ευκλείδεια απόσταση. Η L2 νόρμα ενός διανύσματος $\boldsymbol{x}$ συμβολίζεται ως $||x||_2$ και υπολογίζεται από τη σχέση:

$$||x||_2 = \sqrt{(\sum_i x_i^2)} = \sqrt{x_1^2 + x_2^2 + \cdots + x_i^2} \qquad (4.9.2)$$

Στην **L1 εξομάλυνση (L1 regularization)** το κόστος που προστίθεται είναι ανάλογο της L1 νόρμας των βαρών. Οπότε, η συνάρτηση κόστους, για παράδειγμα της δυαδικής εγκάρσιας εντροπίας που εφαρμόζεται στη δυαδική ταξινόμηση, θα είναι πλέον:

$$C(\boldsymbol{\theta}) = -\frac{1}{m}\left[\sum_{i=0}^{m} y^i \log\left(p^i\right) + \left(1 - y^i\right)\log\left(1 - p^i\right)\right] + \lambda||w||_1 \qquad (4.9.3)$$

όπου m ο αριθμός των παραδειγμάτων στο σύνολο εκπαίδευσης και ο παράγοντας λ, όπου $\lambda > 0$, είναι η παράμετρος της εξομάλυνσης η οποία επιλέγεται με κάποια κριτήρια: όταν επιθυμούμε να μειώσουμε την συνάρτηση κόστους, θέτουμε μικρή τιμή για το λ, ενώ εάν προτιμούμε μικρά βάρη θέτουμε μεγάλη τιμή για το λ.

Στην **L2 εξομάλυνση (L2 regularization)** το κόστος που προστίθεται είναι ανάλογο της L2 νόρμας των βαρών. Οπότε, η συνάρτηση κόστους, για παράδειγμα της δυαδικής εγκάρσιας εντροπίας που εφαρμόζεται στη δυαδική ταξινόμηση, θα είναι πλέον:

$$C(\boldsymbol{\theta}) = -\frac{1}{m}\left[\sum_{i=0}^{m} y^i \log\left(p^i\right) + \left(1 - y^i\right)\log\left(1 - p^i\right)\right] + \lambda||w||_2 \qquad (4.9.4)$$

Η L2 εξομάλυνση στα νευρωνικά δίκτυα επίσης ονομάζεται και δεκαετία των βαρών (weight decay).

Συμβολίζοντας ως $\Omega(\theta)$ γενικά την νόρμα των βαρών, ως C_0 την αρχική συνάρτηση κόστους και C τη νέα συνάρτηση κόστους, τότε η συνάρτηση κόστους εκφράζεται ως:

$$C = C_0 + \lambda\,\Omega(\theta) \qquad (4.9.5)$$

Κατά συνέπεια, όταν εφαρμόζουμε εξομάλυνση βαρών, κατά την εφαρμογή του αλγόριθμου οπισθοδιάδοσης, για τον υπολογισμό της gradient $\nabla_a C$ θα λαμβάνεται ως συνάρτηση κόστους C αυτή που δίνεται από την εξίσωση .

4.9.3 Εξομάλυνση με Πρόωρο Σταμάτημα (Early Stopping)

Κατά την εκπαίδευση συνήθως μεγάλων μοντέλων παρατηρείται συχνά ότι ενώ το λάθος εκπαίδευσης συνεχώς μειώνεται, το λάθος επικύρωσης από ένα σημείο και μετά αυξάνεται, όπως φαίνεται στην Εικόνα 3-11 και παίρνει τη μορφή U-καμπύλης. Αυτό σημαίνει ότι μπορούμε να πετύχουμε ένα μοντέλο με καλύτερο λάθος επικύρωσης επιστρέφοντας στις παραμέτρους που υπήρχαν στο χαμηλότερο σημείο της καμπύλης, δηλαδή στο σημείο του χαμηλότερου λάθους. Κάθε φορά που το λάθος επικύρωσης βελτιώνεται, αποθηκεύεται ένα αντίγραφο των παραμέτρων του μοντέλου. Όταν ο αλγόριθμος εκπαίδευσης τερματίζει, τότε επιστρέφει αυτές τις παραμέτρους αντί για τις τελευταίες. Ο αλγόριθμος τερματίζει όταν δεν υπάρχουν παράμετροι για βελτίωση στο τελευταίο καλύτερο αποθηκευμένο λάθος επικύρωσης μετά από κάποιες προκαθορισμένες επαναλήψεις. Αυτή η διαδικασία ονομάζεται **πρόωρο σταμάτημα (early stopping)** και περιγράφεται με τον παρακάτω ψευδό κώδικα [1]:

```
Έστω n ο αριθμός των βημάτων μεταξύ των εκτιμήσεων
```

```
Έστω p η «υπομονή», ο αριθμός των φορών που παρατηρείται η χειροτέρευση
στο σύνολο επικύρωσης πριν την παραίτηση
Έστω θ₀ οι αρχικές παράμετροι του δικτύου
ΑΡΧΗ
θ ← θ₀
i ← 0
j ← 0
v ← ∞
θ* ← θ
i* ← i
ΟΣΟ j < p ΕΠΑΝΕΛΑΒΕ
      Ενημέρωσε το θ τρέχοντας τον αλγόριθμο για n βήματα
      i ← i + n
      v΄ ← ValidationSetError(θ)
      ΕΑΝ v΄ < v ΤΟΤΕ
            j ← 0
            θ* ← θ
            i* ← i
            v ← v΄
      ΑΛΛΙΩΣ
            j ← j + 1
      ΤΕΛΟΣ ΕΑΝ
ΤΕΛΟΣ ΟΣΟ
ΤΕΛΟΣ
```

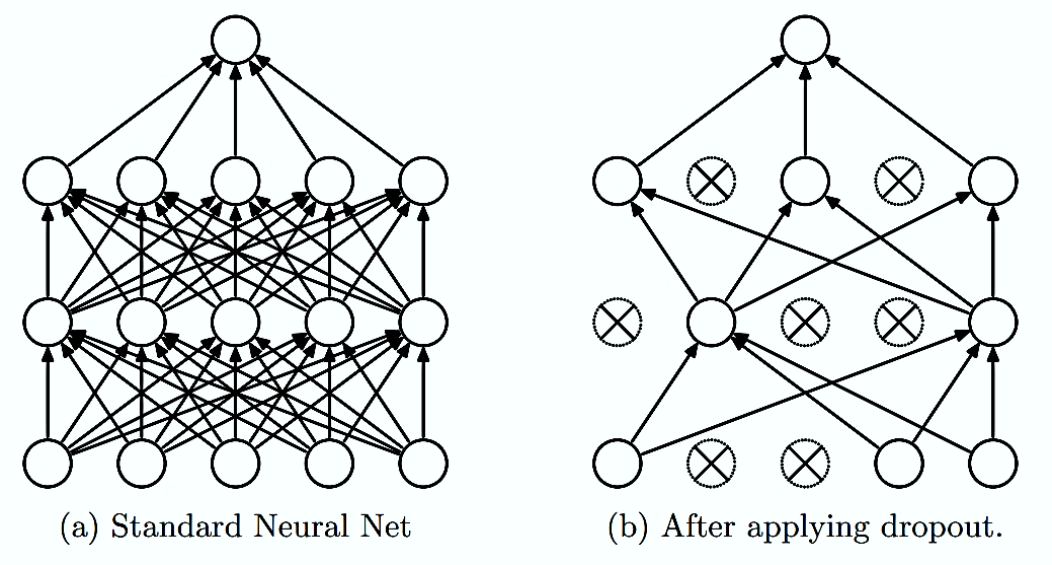
Οι καλύτερες παράμετροι είναι οι **θ***, και ο βέλτιστος αριθμός βημάτων εκπαίδευσης είναι το i^* . Το πρόωρο σταμάτημα είναι από τις πιο συχνά χρησιμοποιούμενες μεθόδους λόγω της αποτελεσματικότητάς του και της απλότητάς του. Επίσης, μπορεί να θεωρηθεί ότι, εισάγει ακόμα μία υπερπαράμετρο στο μοντέλο, των αριθμό των βημάτων εκπαίδευσης. Οι περισσότερες υπερπαράμετροι ελέγχου του μοντέλου, έχουν το σχήμα της U-καμπύλης της Εικόνας 3-11. Στην περίπτωση του πρόωρου σταματήματος, η αποτελεσματική χωρητικότητα του μοντέλου ελέγχεται ορίζοντας το πόσα βήματα χρειάζεται για την προσαρμογή στο σύνολο εκπαίδευσης.

4.9.4 Εξομάλυνση με Dropout

Το dropout είναι μια υπολογιστικά οικονομική τεχνική, αλλά συγχρόνως και μια ισχυρή μέθοδος εξομάλυνσης για μια ευρεία οικογένεια μοντέλων. Παρέχει έναν αποτελεσματικό τρόπο για να συνδυαστούν εκθετικά πολλές διαφορετικές αρχιτεκτονικές νευρωνικών δικτύων [1].

Ο όρος "**dropout**" αναφέρεται στην απόρριψη μονάδων (κρυφών και ορατών) στο δίκτυο. Η απόρριψη μιας μονάδας, σημαίνει ότι την αφαιρούμε προσωρινά από το δίκτυο μαζί με όλες τις εισερχόμενες και εξερχόμενες συνδέσεις της, όπως φαίνεται στην Εικόνα 4-20 [38]. Η επιλογή των μονάδων είναι τυχαία. Στην απλούστερη περίπτωση, κάθε μονάδα διατηρείται με μια σταθερή πιθανότητα, έστω p ανεξάρτητη από τις άλλες μονάδες, όπου η p μπορεί να επιλεγεί χρησιμοποιώντας ένα σύνολο επικύρωσης ή απλά να τεθεί μεταξύ 0.2 και 0.5, που φαίνεται να είναι κοντά στο βέλτιστο για ένα μεγάλο εύρος δικτύων και εργασιών [23]. Όμως, για τις μονάδες εισόδου η βέλτιστη πιθανότητα διατήρησης είναι συνήθως κοντά στο 1.

Η θεωρητική ανάλυση του dropout δίνεται αναλυτικά στην [38].



Εικόνα 4-20. Η τεχνική του dropout: (a) Αρχικό MPL, (b) MLP μετά το dropout

Πηγή: N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 56, pp. 1929–1958, 2014

4.9.5 Εξομάλυνση Batch Normalization

Η τεχνική της εξομάλυνσης με **κανονικοποίηση κατά δέσμες (batch normalization)** εφαρμόζεται κατά κανόνα σε βαθιά νευρωνικά δίκτυα, δηλαδή με πολλά κρυφά επίπεδα, όπου η εκπαίδευση είναι περισσότερο πολύπλοκη. Η τεχνική αφορά την κανονικοποίηση των συναρτήσεων ενεργοποίησης στην είσοδο ή στην έξοδο ενός κρυφού επιπέδου [1]. Με την batch normalization τα βαθιά νευρωνικά δίκτυα γίνονται πιο σταθερά γιατί περιορίζονται οι ακραίες τιμές των βαρών και παράλληλα επιτυγχάνονται υψηλότεροι ρυθμοί μάθησης και έτσι μπορεί να παραληφθεί το dropout . Η θεωρητική ανάλυση της batch normalization δίνεται αναλυτικά στην [40].

4.10 Σύνοψη της Διαδικασίας Εκπαίδευσης

Με βάση όσα αναφέρθηκαν στις παραγράφους 4.5 έως 4.9 η διαδικασία της εκπαίδευσης ενός MLP για ένα πρόβλημα δυαδικής ταξινόμησης, όπως έχει περιγραφεί στην παράγραφο 4.4, συνοψίζεται στον Πίνακα 4-2.

Έχοντας πλέον το θεωρητικό υπόβαθρο, μπορούμε να επιλέξουμε τις υπερπαραμέτρους που απαιτούνται για την υλοποίηση ενός MLP από διάφορες βιβλιοθήκες λογισμικού που διατίθενται για διάφορες γλώσσες προγραμματισμού, όπως πχ για Python και R, οι οποίες είναι και οι δημοφιλέστερες για την ΜΜ γενικότερα και την ΒΜ ειδικότερα.

Βήμα	**Περιγραφή**	**Παράδειγμα/Αλγόριθμος**
1α	*Ορισμός της αρχιτεκτονικής του δικτύου L επιπέδων*	Δίκτυο 3 επιπέδων με 10 νευρώνες στο πρώτο επίπεδο,10 νευρώνες στο δεύτερο και 2 νευρώνες στην έξοδο
1β	*Επιλογή της συνάρτησης ενεργοποίησης στο επίπεδο εξόδου και στα κρυφά επίπεδα*	Σιγμοειδής στην έξοδο και ReLU στα κρυφά επίπεδα
1γ	*Επιλογή συνάρτησης κόστους*	Συνάρτηση δυαδικής εγκάρσιας εντροπίας
1δ	*Επιλογή εξομάλυνσης*	Πρόωρο σταμάτημα ή/και dropout
1ε	*Επιλογή βελτιστοποιητή*	SGD

2	*Τυχαία αρχικοποίηση των παραμέτρων εκμάθησης* $\boldsymbol{w}$, $\boldsymbol{b}$ *για κάθε επίπεδο* l.	—
3	*Είσοδος στο σύστημα του συνόλου εκπαίδευσης και των τιμών στόχων*	—
4	*Αλγόριθμος εμπροσθοδιάδοσης*	Για κάθε επίπεδο $l = 2,3, \dots, L$ υπολογισμός $\boldsymbol{z^l = w^l a^{l-1} + b^l}$, $\boldsymbol{a^l} = \sigma(\boldsymbol{z^l})$
5	*Υπολογισμός του λάθους εξόδου με εφαρμογή της συνάρτησης κόστους σε* $y_{pred,}y$	Υπολογισμός του διανύσματος $\boldsymbol{\delta^L} = \nabla_a C \odot \sigma'(\boldsymbol{z^L})$
6	*Αλγόριθμος οπισθοδιάδοσης*	Για κάθε $l = L-1, L-2, \dots, 2$ υπολογισμός $\delta^l = ((\boldsymbol{w^{l+1}})^T \delta^{l+1}) \odot \sigma'(\boldsymbol{z}^l)$ Έξοδος σε κάθε επίπεδο από την gradient συνάρτησης κόστους $\frac{\partial C}{\partial w^l_{jk}} = a_k^{l-1}\delta_j^l$ και $\frac{\partial C}{\partial b_j^l} = \delta_j^l$
7	*Επιστροφή στο βήμα 4 έως ότου ικανοποιηθεί η σύγκλιση*	

Πίνακας 4-2.Η διαδικασία εκπαίδευσης του MLP

5 Μεθοδολογία

Στο κεφάλαιο αυτό αρχικά παρουσιάζεται το περιβάλλον υλοποίησης σε Python, όπου αναφέρονται οι κυριότερες βιβλιοθήκες, με έμφαση στην Keras και η μεθοδολογία για την ανάπτυξη του έργου. Στη συνέχεια, γίνεται μια παρουσίαση του συνόλου δεδομένων WBCD, προκειμένου να έχουμε μια εικόνα για τα δεδομένα και, τέλος, γίνεται μια ανασκόπηση της βιβλιογραφίας, όπου παρουσιάζονται τρεις μελέτες που αφορούν την υλοποίηση MLPs για το ίδιο σύνολο δεδομένων με Python και τις σχετικές βιβλιοθήκες.

5.1 Περιβάλλον Υλοποίησης

Για τη δημιουργία του έργου επιλέχθηκε η γλώσσα προγραμματισμού Python [41], η οποία στις μέρες μας είναι η δημοφιλέστερη για προβλήματα ΒΜ και ανάλυσης δεδομένων. Η Python είναι μια διερμηνευόμενη (interpreted), υψηλού επιπέδου και γενικού σκοπού γλώσσα προγραμματισμού, δημιουργήθηκε από τον Guido van Rossum και παρουσιάστηκε το 1991. Η φιλοσοφία σχεδιασμού της Python δίνει έμφαση στην αναγνωσιμότητα του κώδικα. Οι γλωσσικές δομές και η αντικειμενοστραφής προσέγγιση στοχεύουν να βοηθήσουν τους προγραμματιστές να γράψουν σαφή, λογικό κώδικα για μικρά και μεγάλα έργα. Η Python είναι ισχυρή και γρήγορη, τρέχει παντού, είναι φιλική και εύχρηστη και είναι δωρεάν και ανοιχτού κώδικα. Η έκδοση 2 της Python επίσημα σταμάτησε να υποστηρίζεται το 2020 και αυτή τη στιγμή υπάρχει υποστήριξη για τις εκδόσεις 3.6.x και τις μεταγενέστερες.

Οι διερμηνευτές της Python είναι διαθέσιμοι για πολλά λειτουργικά συστήματα. Μια παγκόσμια κοινότητα προγραμματιστών αναπτύσσει και διατηρεί τη Cpython που βασίζεται στη γλώσσα προγραμματισμού C, μια δωρεάν εφαρμογή ανοιχτού κώδικα. Ένας μη κερδοσκοπικός οργανισμός, ο Python Software Foundation, διαχειρίζεται και κατευθύνει πόρους για την ανάπτυξη Python και CPython.

Ως προγραμματιστικό περιβάλλον υλοποίησης επιλέχθηκε το Colaboratory [42] ή "Colab" για συντομία, το οποίο δίνει τη δυνατότητα να γράφουμε και να εκτελούμε την Python στο πρόγραμμα περιήγησης, με μηδενική ρύθμιση παραμέτρων και με δωρεάν πρόσβαση σε GPU. Τα Colab notebooks[17] είναι Jupyter notebooks[18] που λειτουργούν στο νέφος και είναι ενσωματωμένα στο Google Drive. Τα notebooks μπορούν να ανοίξουν είτε από το Google Drive είτε από τη διασύνδεση Colaboratory και είναι πολύ φιλικά στον χρήστη. Το περιβάλλον του Colab διαθέτει μια πληθώρα βιβλιοθηκών για την επιστήμη των δεδομένων και την ΜΜ, έτσι ώστε να μην απαιτείται η εγκατάσταση εικονικού περιβάλλοντος στον υπολογιστή μας (όπως για παράδειγμα

[17] Χρησιμοποιούμε τον αγγλικό όρο notebook που μπορεί να αποδοθεί και ως σημειωματάριο, γιατί μπορεί να συνδυάσει κώδικα και κείμενο.

[18] https://jupyter.org/

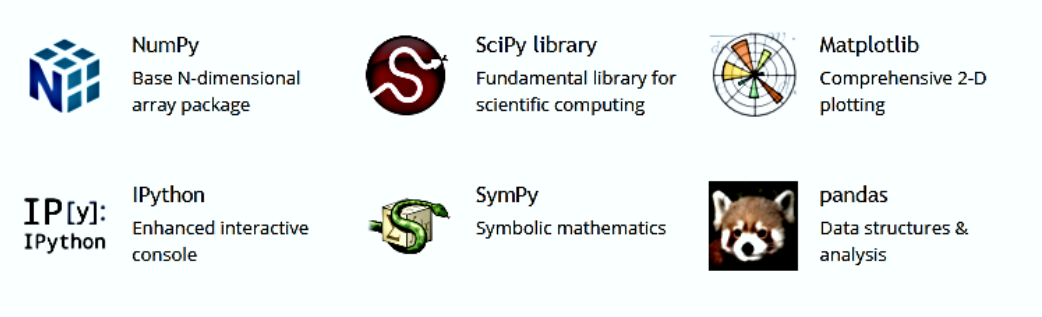
το Anaconda[19]). Στο Παράρτημα Α δίνονται οι εκδόσεις των βιβλιοθηκών της Python στο Colab κατά την δημιουργία του έργου.

5.1.1 Βιβλιοθήκες Python

Όλες οι βιβλιοθήκες για την γλώσσα προγραμματισμού Python που χρησιμοποιήθηκαν στο έργο είναι ελεύθερες και ανοιχτού κώδικα (free and open-source), υποστηρίζονται από μεγάλες κοινότητες προγραμματιστών και διαθέτουν πολύ καλή τεκμηρίωση (documentation). Οι βιβλιοθήκες που χρησιμοποιήθηκαν περιγράφονται συνοπτικά στη συνέχεια.

A. SciPy [43], [44]: Η βασική βιβλιοθήκη για επιστημονικούς και τεχνικούς υπολογισμούς.

Η SciPy περιέχει λειτουργικές μονάδες για βελτιστοποίηση, γραμμική άλγεβρα, ολοκλήρωση, παρεμβολή, ειδικές λειτουργίες και άλλες εργασίες κοινές στην επιστήμη και τη μηχανική. Βασίζεται στο αντικείμενο των πινάκων NumPy, αποτελεί μέρος της στοίβας NumPy και το οικοσύστημά της περιλαμβάνει ένα σύνολο επιστημονικών βιβλιοθηκών, όπως φαίνεται στην Εικόνα 5-1. Η στοίβα NumPy αναφέρεται επίσης μερικές φορές ως στοίβα SciPy.



Εικόνα 5-1. Το οικοσύστημα της SciPy

Πηγή: "SciPy." [Online]. Available: https://www.scipy.org/ [Accessed: 10-Oct-2020]

Πιο συγκεκριμένα, τα βασικά πακέτα- βιβλιοθήκες του οικοσυστήματος της SciPy είναι τα παρακάτω:

NumPy [45]**,** [46]**:** Βασική βιβλιοθήκη για N-διάστατους πίνακες.

Η NumPy προσθέτει υποστήριξη για μεγάλους, πολυδιάστατους πίνακες και μητρώα, μαζί με μια μεγάλη συλλογή μαθηματικών συναρτήσεων υψηλού επιπέδου για λειτουργίες γραμμικής άλγεβρας. Δημιουργήθηκε το 2005, με βάση τις πρώτες εργασίες των βιβλιοθηκών Numerical και Numarray

pandas [47]**:** Βιβλιοθήκη για δομές δεδομένων και ανάλυση.

[19] https://www.anaconda.com/

Η βιβλιοθήκη pandas είναι για χειρισμό και ανάλυση δεδομένων. Συγκεκριμένα, προσφέρει δομές δεδομένων και λειτουργίες για χειρισμό αριθμητικών πινάκων και χρονοσειρών. Το όνομα προέρχεται από τον όρο "panel data", έναν όρο οικονομετρίας για σύνολα δεδομένων που περιλαμβάνει παρατηρήσεις σε πολλαπλές χρονικές περιόδους για τα ίδια αντικείμενα.

Matplotlib [48]**,** [49]**:** Βιβλιοθήκη για ολοκληρωμένη 2-D σχεδίαση.

Η matplotlib είναι η βιβλιοθήκη σχεδίασης και η αριθμητική επέκταση μαθηματικών της NumPy. Παρέχει μια αντικειμενοστραφή διεπαφή προγραμματισμού (Application Programming Interface – API) για την ενσωμάτωση σχεδίων σε εφαρμογές με χρήση εργαλείων γενικής χρήσης γραφικών διεπαφών χρήστη (Graphical User Interface – GUI). Η matplotlib δεν υποστηρίζει την Python 2 μετά το 2020. Η Pyplot είναι μια ενότητα της matplotlib που παρέχει διεπαφή τύπου MATLAB[20]. Έχει σχεδιαστεί έτσι ώστε, να μπορεί να χρησιμοποιηθεί όπως το MATLAB, με τη δυνατότητα χρήσης της Python και το πλεονέκτημα του ότι είναι δωρεάν και ανοιχτού κώδικα. Η βιβλιοθήκη **Seaborn** [50] είναι για την οπτικοποίηση δεδομένων και βασίζεται στην matplotlib. Παρέχει διεπαφή υψηλού επιπέδου για τη σχεδίαση ελκυστικών και ενημερωτικών στατιστικών γραφικών.

IPython [51]**:** Η βελτιωμένη διαδραστική κονσόλα

Η IPython παρέχει μια πλούσια αρχιτεκτονική για διαδραστικούς υπολογισμούς με ισχυρό διαδραστικό κέλυφος (shell) και είναι πυρήνας (kernel) για το Jupyter και κατ' επέκταση για το Colab. Το Jupyter παρέχει τη λειτουργικότητα IPython στο πρόγραμμα περιήγησης στο Web, δίνοντας τη δυνατότητα τεκμηρίωσης των υπολογισμών. Επίσης, υποστηρίζει τη διαδραστική οπτικοποίηση δεδομένων και τη χρήση εργαλείων GUI και εύκολα στη χρήση εργαλεία υψηλής απόδοσης για παράλληλους υπολογισμούς**.**

B. scikit-learn [52], [53]: Βιβλιοθήκη για Μηχανική Μάθηση

Η βιβλιοθήκη scikit learn διαθέτει διάφορους αλγόριθμους ΜΜ για ταξινόμηση, παλινδρόμηση, συσταδοποίηση, MLP, προεπεξεργασία δεδομένων (εξαγωγή χαρακτηριστικών και εξομάλυνση) και άλλους. Είναι μια βιβλιοθήκη ανοιχτού κώδικα που υποστηρίζει την εποπτευόμενη και μη εποπτευόμενη μάθηση. Παρέχει εργαλεία για την αυτόματη εύρεση των καλύτερων παραμέτρων(μέσω εγκυρότητας). Έχει σχεδιαστεί για να λειτουργεί με τις αριθμητικές και επιστημονικές βιβλιοθήκες NumPy και SciPy.

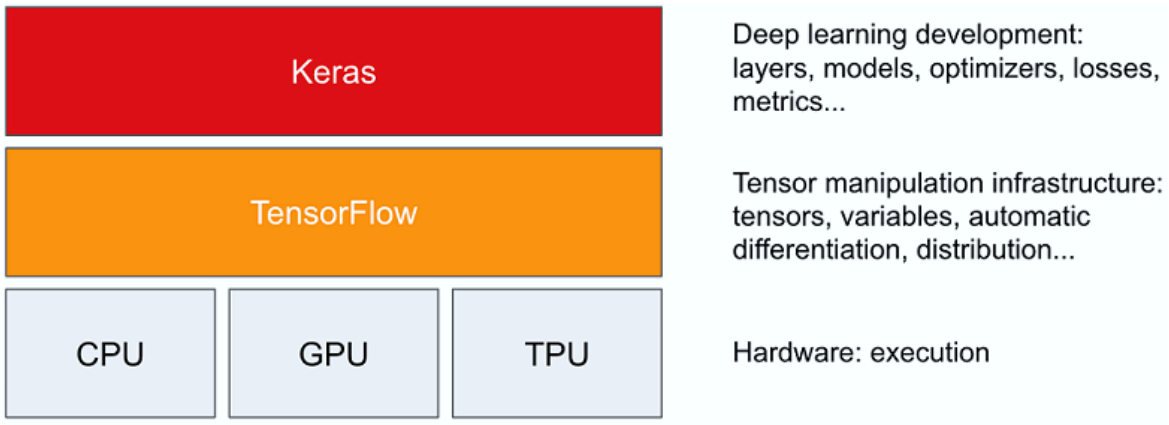
[20] https://www.mathworks.com/products/matlab.html

Γ. TensorFlow [54], [55]: Βιβλιοθήκη για Μηχανική Μάθηση

Η βιβλιοθήκη TensorFlow διαθέτει ένα ολοκληρωμένο, ευέλικτο οικοσύστημα εργαλείων, που δίνει τη δυνατότητα στους ερευνητές να προωθήσουν την τελευταία λέξη της τεχνολογίας στην ΜΜ και στους προγραμματιστές να κατασκευάζουν και να αναπτύσσουν εύκολα εφαρμογές που υποστηρίζονται από την ΜΜ. Η TensorFlow αναπτύχθηκε αρχικά από ερευνητές και μηχανικούς που εργάζονταν στην ομάδα της Google Brain στην Machine Intelligence της Google. Συνδυάζει τέσσερα βασικά χαρακτηριστικά:

- Αποτελεσματική εκτέλεση σε χαμηλού επιπέδου λειτουργίες τανυστών σε CPU, GPU, ή TPU.
- Υπολογισμό της gradient τυχαίων διαφοροποιήσιμων εκφράσεων
- Κλιμάκωση υπολογισμού σε πολλές συσκευές (π.χ. ο υπερυπολογιστής Summit στο Oak Ridge National Lab, που εκτείνεται σε 27.000 GPU).
- Εξαγωγή προγραμμάτων σε εξωτερικές εκτελέσεις, όπως διακομιστές, προγράμματα περιήγησης, κινητές συσκευές και ενσωματωμένες συσκευές.

Η TensorFlow παρέχει υψηλού επιπέδου Python API για κατασκευή μοντέλων ΜΜ. Ειδικά για την ΒΜ παρέχει την βιβλιοθήκη Keras.



Εικόνα 5-2. TensorFlow και Keras
Πηγή: F. Chollet, *Deep Learning with Python, Second Edition*. Manning Early Access Program (MEAP), 2020

Keras [56], [57]: Βιβλιοθήκη Βαθιάς Μάθησης

Η βιβλιοθήκη Keras είναι μία διεπαφή προγραμματισμού εφαρμογών (Application Programming Interfaces- API) βαθιάς μάθησης σε Python, που τρέχει πάνω από την πλατφόρμα της TensorFlow (Εικόνα 5.2). Είναι σχεδιασμένη για γρήγορους πειραματισμούς με βαθιά νευρωνικά δίκτυα, εστιάζει στο να είναι φιλική προς το χρήστη, αρθρωτή και επεκτάσιμη. Ο κύριος συγγραφέας και συντηρητής της είναι ο François Chollet, μηχανικός της Google. Ο κώδικάς της φιλοξενείται στο Github και διαθέτει μια μεγάλη κοινότητα που την υποστηρίζει.

Μέχρι την έκδοση 2.3 η Keras υποστήριζε διάφορα backends, συμπεριλαμβανομένων των TensorFlow, Microsoft Cognitive Toolkit, R, Theano, και PlaidML. Από την έκδοση2.4.0 και μετά, έχει ως backend την TensorFlow.

5.1.2 Βασικές Συνιστώσες Βιβλιοθήκης Keras

Η Keras αποτελείται από διάφορα APIs και τα περιεχόμενά τους δίνονται στην [58]. Τα πιο βασικά είναι τα παρακάτω:

- **Μοντέλα (Models)[21]:** Είναι η συνιστώσα μέσω της οποίας επιλέγουμε την αρχιτεκτονική των επιπέδων του μοντέλου. Για τα MLP η επιλογή του μοντέλου είναι συγκεκριμένη, από τη στιγμή που τα επίπεδα είναι πλήρως συνδεδεμένα και αναπτύσσονται ακολουθιακά (Sequential model), είναι δηλαδή είναι στοίβες επιπέδων με ένα μοναδικό επίπεδο εισόδου και ένα μοναδικό επίπεδο εξόδου.
- **Επίπεδα (Layers)**[22]: Είναι τα βασικά δομικά στοιχεία οποιουδήποτε ΤΝΔ. Οι τελεστές είναι συναρτήσεις διανυσμάτων τιμών που μετασχηματίζουν τα δεδομένα και αφορούν τα επίπεδα. Για παράδειγμα, βασικός τελεστής είναι οι συναρτήσεις ενεργοποίησης[23]. Επίσης, βασικός τελεστής είναι οι αρχικοποιητές (initializers) [24] οι οποίοι παρέχουν τις αρχικές τιμές για τις παραμέτρους του μοντέλου (βάρη και πολώσεις) κατά την έναρξη της εκπαίδευσης. Η αρχικοποίηση παίζει σημαντικό ρόλο στην εκπαίδευση των ΤΝΔ, καθώς η μη κατάλληλη αρχικοποίηση παραμέτρων μπορεί να οδηγήσει σε αργή ή καθόλου σύγκλιση του αλγόριθμου. Τέλος, ένας άλλος βασικός τελεστής είναι οι εξομαλυντές (regularizers), όπως πχ οι εξομαλυντές βαρών[25] και οι εξομαλυντές επιπέδων[26] όπως το

[21] https://keras.io/api/models/

[22] https://keras.io/api/layers/

[23] https://keras.io/api/layers/activations/

[24] https://keras.io/api/layers/initializers/

[25] https://keras.io/api/layers/regularizers/

[26] https://keras.io/api/layers/regularization_layers/

Dropout. Οι εξομαλυντές παρέχουν τους απαραίτητους μηχανισμούς ελέγχου για την αποφυγή του overfitting και την επίτευξη του στόχου της γενίκευσης του μοντέλου.

- **Βελτιστοποιητές (Optimizers)** [27]: Είναι ο σκελετός κάθε βιβλιοθήκης ΒΜ. Παρέχουν τα απαραίτητα συστατικά για την ενημέρωση των παραμέτρων μοντέλου χρησιμοποιώντας τις gradient σε σχέση με τον στόχο βελτιστοποίησης.
- **Συναρτήσεις Κόστους (Loss Functions)** [28]: Είναι κλειστής μορφής και διαφοροποιήσιμες μαθηματικές εκφράσεις που χρησιμοποιούνται για την επίτευξη του στόχου της βελτιστοποίησης του μοντέλου.
- **Callbacks[29]:** Ένα callback είναι ένα αντικείμενο που μπορεί να εκτελέσει ενέργειες σε διάφορα στάδια της εκπαίδευσης, για παράδειγμα στην έναρξη ή στο τέλος μιας εποχής (epoch), πριν ή μετά μια απλή δέσμη (batch) κλπ. Τα callbacks χρησιμοποιούνται όταν θέλουμε περιοδικά να σώζουμε το μοντέλο μας στον δίσκο, να εφαρμόσουμε ομαλοποίηση με πρόωρο σταμάτημα, να δούμε τις εσωτερικές καταστάσεις και τα στατιστικά ενός μοντέλου και πολλά άλλα.

Επίσης, διατίθεται μια βιβλιοθήκη, η Keras Tuner, την οποία θα χρησιμοποιήσουμε για την επιλογή των βέλτιστων υπερπαραμέτρων ενός μοντέλου, ή αλλιώς, όπως ονομάζεται, για τη **ρύθμιση υπερπαραμέτρων (hyperparameter tuning)** ή **υπερρύθμιση (hypertuning)**. Όπως είδαμε, οι υπερπαράμετροι είναι οι μεταβλητές που διέπουν τη διαδικασία εκπαίδευσης και την τοπολογία ενός μοντέλου, παραμένουν σταθερές κατά τη διάρκεια της εκπαίδευσης και επηρεάζουν άμεσα την απόδοση του προγράμματος. Μέσω αυτής της βιβλιοθήκης δίνεται η δυνατότητα επιλογής της χωρητικότητας ενός μοντέλου, καθώς και άλλων υπερπαραμέτρων που επηρεάζουν την ταχύτητα και την ποιότητα του αλγορίθμου, όπως για παράδειγμα ο ρυθμός μάθησης. Τα μοντέλα που δημιουργούνται με υπερρύθμιση, ονομάζονται **υπερμοντέλα (hypermodels).**

5.2 Βασική Προσέγγιση Υλοποίησης του Έργου (Project)

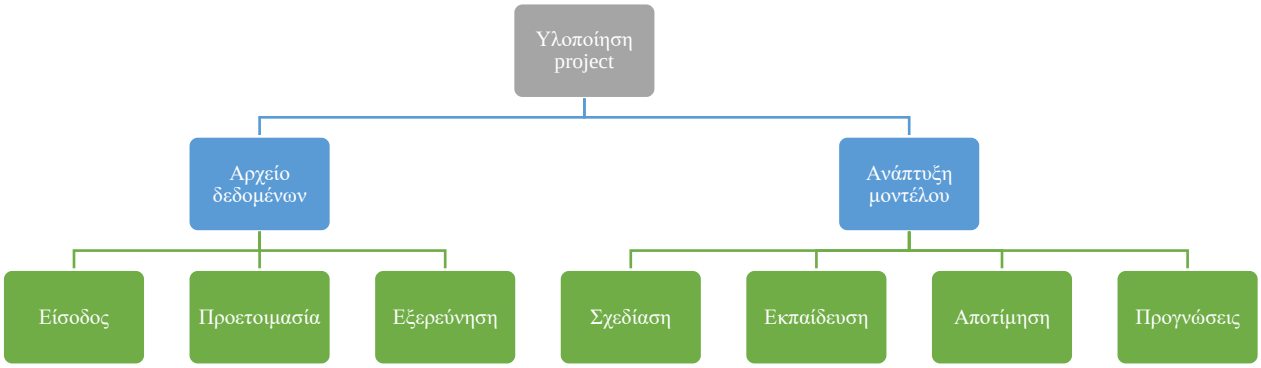
Το πρόβλημα της διάγνωσης του καρκίνου του στήθους που έχουμε να επιλύσουμε, είναι ένα καλώς ορισμένο πρόβλημα επιβλεπόμενης μάθησης και πιο συγκεκριμένα δυαδικής ταξινόμησης. Έχουμε επιλέξει τον αλγόριθμο ΜΜ που θα εφαρμόσουμε, και μάλιστα αλγόριθμο που ανήκει

[27] https://keras.io/api/optimizers/

[28] https://keras.io/api/losses/

[29] https://keras.io/api/callbacks/

στην BM και αυτός είναι ο MLP. Επίσης, έχουμε επιλέξει το σύνολο δεδομένων με βάση το οποίο θα δημιουργήσουμε τα μοντέλα μας και το περιβάλλον υλοποίησης και αυτό είναι το σύνολο δεδομένων για τη διάγνωση του καρκίνου του μαστού του Ουισκόνσιν (Wisconsin Breast Cancer Diagnostic -WBCD) [9], που διατίθεται στο αποθετήριο μηχανικής μάθησης UCI [10]. Αντλώντας την απαιτούμενη πληροφορία από τους [1], [22], [23], θα ακολουθήσουμε μια βασική προσέγγιση, όπως φαίνεται στην Εικόνα 5-3, όπου παρουσιάζονται οι βασικές συνιστώσες για την υλοποίηση του έργου.

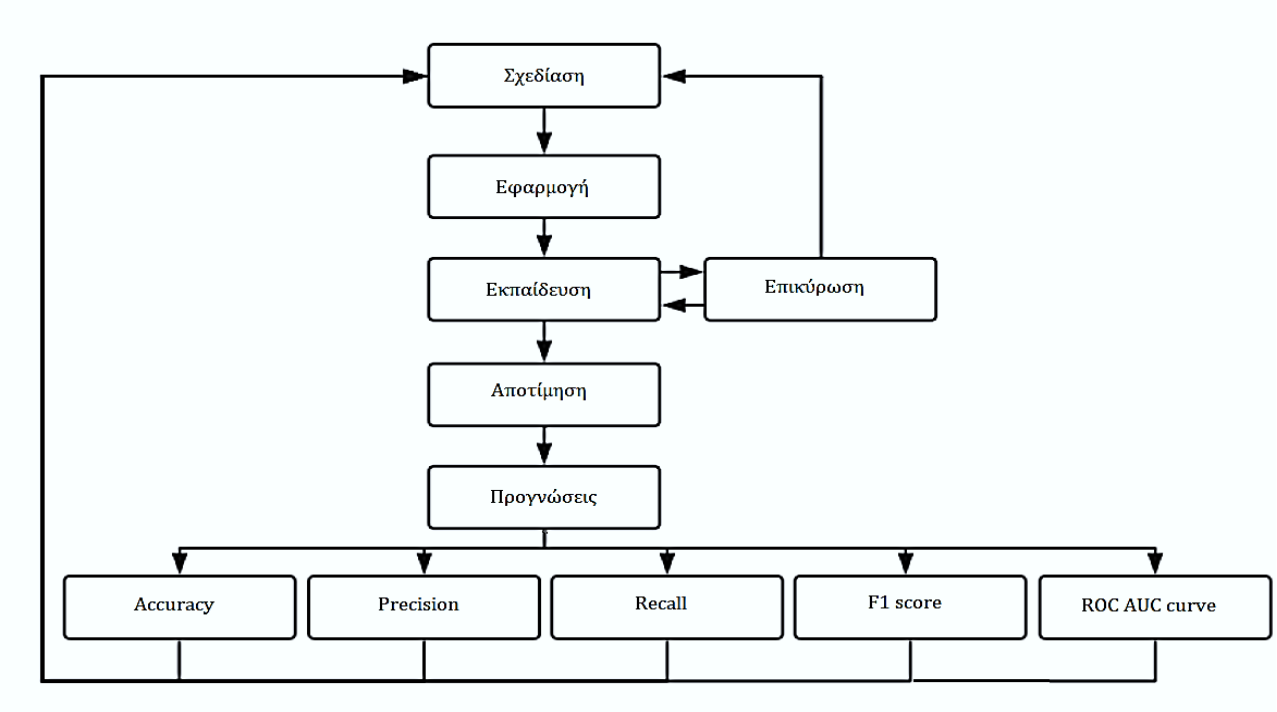


Εικόνα 5-3. Οι συνιστώσες υλοποίησης του έργου

Αφενός έχουμε ένα συγκεκριμένο σύνολο δεδομένων, το οποίο θα πρέπει να το εισάγουμε στο προγραμματιστικό μας περιβάλλον, να το προετοιμάσουμε και να το εξερευνήσουμε. Αυτή είναι μια διαδικασία που θα γίνει μία φορά και στη συνέχεια θα προχωρήσουμε στη διαδικασία της ανάπτυξης του μοντέλου, όπου κι εδώ ακολουθείται μια ροή εργασιών, η οποία όμως είναι επαναληπτική και απαιτεί δοκιμές και πειραματισμούς για να δημιουργήσουμε ένα προγνωστικό μοντέλο το οποίο θα είναι γρήγορο και αποτελεσματικό.

Στο σημείο αυτό θα δώσουμε μόνο τη ροή των εργασιών για την ανάπτυξη του μοντέλου που θα ακολουθήσουμε για την υλοποίηση των MLPs που περιλαμβάνεται στο επόμενο κεφάλαιο και στην επόμενη παράγραφο θα πραγματοποιήσουμε μια στοιχειώδη εξερευνητική ανάλυση των δεδομένων (Exploratory Data Analysis – EDA).

Η ροή εργασιών για την ανάπτυξη του μοντέλου δίνεται πιο αναλυτικά στην Εικόνα 5-4.



Εικόνα 5-4. Ροή Εργασιών Ανάπτυξης MLP

Η ανάπτυξη του μοντέλου είναι μια επαναληπτική διαδικασία, όπως φαίνεται από τα βέλη στο διάγραμμα, μέχρι να καταλήξουμε στο μοντέλο που μας ικανοποιεί. Πιο αναλυτικά η κάθε εργασία του διαγράμματος περιγράφεται παρακάτω.

Σχεδίαση

Κατά τη σχεδίαση του μοντέλου επιλέγουμε για το μοντέλο τα εξής:

- Αρχιτεκτονική: αριθμός επιπέδων και νευρώνων ανά επίπεδο
- Συνάρτηση ενεργοποίησης κρυφών επιπέδων
- Συνάρτηση ενεργοποίησης εξόδου
- Τυχόν μεθόδους εξομάλυνσης

Εφαρμογή

Η εφαρμογή αφορά την προετοιμασία μοντέλου για την εκπαίδευση με επιλογή της συνάρτησης κόστους, του βελτιστοποιητή, του ρυθμού μάθησης και του μέτρου απόδοσης.

Εκπαίδευση – Επικύρωση

Εκπαιδεύουμε και επικυρώνουμε το μοντέλο ορίζοντας τα σύνολα εκπαίδευσης και επικύρωσης, τον αριθμό των εποχών, το batch size, και εφόσον το επιθυμούμε εφαρμόζουμε το πρόωρο σταμάτημα για εξομάλυνση.

Αποτίμηση

Κατά την αποτίμηση πραγματοποιείται ο υπολογισμός των τιμών της συνάρτησης κόστους και του μέτρου αποτίμησης στο σύνολο δοκιμής.

Προγνώσεις

Χρησιμοποιούμε το μοντέλο για εξαγωγή προγνώσεων και το αποτιμούμε με μετρικές απόδοσης για να διαπιστώσουμε την ικανότητα γενίκευσης του μοντέλου.

Μετρικές απόδοσης

Οι μετρικές απόδοσης για ένα πρόβλημα δυαδικής ταξινόμησης παρουσιάστηκαν στην παράγραφο 3.3.2. Εδώ τις αναλύουμε με βάση το πρόβλημα μας, που αφορά τη διάγνωση για ύπαρξη ασθένειας ή όχι.

Η ορθότητα, δηλαδή το κλάσμα των ορθών προβλέψεων, τυπικά δεν αποτελεί ικανή πληροφορία για την αποτίμηση του μοντέλου, από τη στιγμή μάλιστα που το σύνολο δεδομένων δεν είναι πλήρως ισορροπημένο. Ενώ είναι ένα σημείο εκκίνησης, μπορεί να μας οδηγήσει σε λάθος αποφάσεις. Μοντέλα με μεγάλη ορθότητα, μπορεί να μην έχουν ικανοποιητικές τιμές για ακρίβεια και ανάκληση. Για τον λόγο αυτό, πρέπει να δούμε και άλλες μετρικές.

Η ακρίβεια είναι η ικανότητα του ταξινομητή να μη προβλέπει ως θετικό ένα στιγμιότυπο που είναι αρνητικό, αλλά και το αντίστροφο. Στην περίπτωσή μας η ακρίβεια είναι όταν το μοντέλο προβλέπει κακοήθεια και στην πραγματικότητα μετρά πόσο σίγουρα ένας όγκος είναι πραγματικά κακοήθης. Για παράδειγμα, όταν η ακρίβεια είναι 0.9 αυτό σημαίνει ότι εάν το μοντέλο προβλέψει 100 κακοήθεις όγκους, οι 90 από αυτούς θα είναι πραγματικά κακοήθεις και 10 θα είναι καλοήθεις (ψευδές).

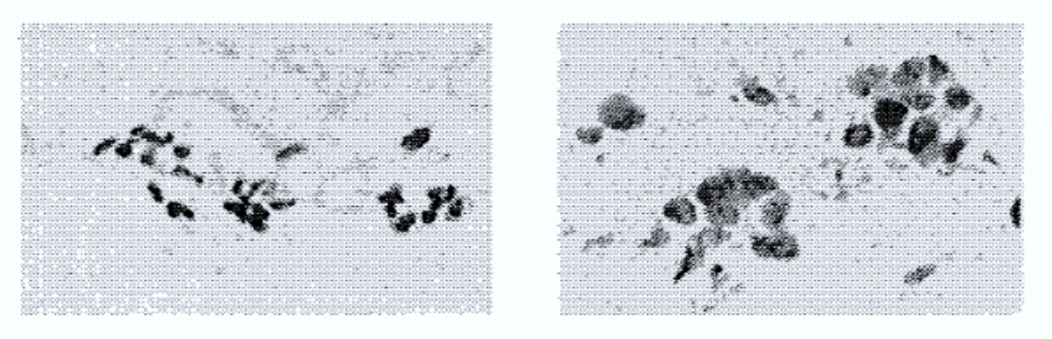
Η ανάκληση είναι η ικανότητα του ταξινομητή να βρίσκει όλα τα θετικά στιγμιότυπα. Στην περίπτωσή μας, η ανάκληση δείχνει πόσο καλά ο ταξινομητής μας μπορεί να βρει τους κακοήθεις όγκους. Για παράδειγμα, όταν η ανάκληση είναι 0.8 αυτό δείχνει ότι το μοντέλο μας βρίσκει μόνο το 80% των πραγματικά κακοήθων όγκων. Το υπόλοιπο 20% των πραγματικά κακοήθων δεν θα βρεθούν από τη διάγνωση που βασίζεται σε αυτό το μοντέλο, κάτι που δεν είναι αποδεκτό.

Η F1 τιμή είναι ο σταθμισμένος μέσος όρος της ακρίβειας και της ανάκλησης, προκειμένου να έχουν την ίδια συμβολή στην εκτίμηση του μοντέλου. Η σχετική συμβολή της ακρίβειας και της ανάκλησης είναι ίσες και είναι η πλέον συνιστώμενη μετρική, σε συνδυασμό με την ROC AUC καμπύλη.

Η ROC AUC καμπύλη είναι η γενική μετρική απόδοσης του μοντέλου. Ο τέλειος ταξινομητής έχει ROC AUC ίση με 1, ενώ ένας κακός ταξινομητής θα έχει 0.5, ίσως και λιγότερο.

5.3 Το Σύνολο Δεδομένων Wisconsin Breast Cancer Diagnostic

Το σύνολο δεδομένων που επιλέξαμε, είναι το σύνολο δεδομένων για τη διάγνωση του καρκίνου του μαστού του Ουισκόνσιν (Wisconsin Breast Cancer Diagnostic -WBCD) [9], ο οποίο διατίθεται υπάρχει στο αποθετήριο μηχανικής μάθησης UCI [10]. Τα δεδομένα συλλέχθηκαν από τους Wolberg, Street and Mangasarian από τα Νοσοκομεία του Πανεπιστημίου του Ουισκόνσιν [11] και περιλαμβάνουν αριθμητικά αποτελέσματα από εικόνες βιοψίας όγκων του μαστού (Εικόνα 5-5), με τη μέθοδο της αναρρόφησης με λεπτή βελόνα (FNA - Fine Needle Aspiration). Τα δεδομένα προήλθαν από την επεξεργασία των εικόνων και περιγράφουν τα χαρακτηριστικά των κυττάρων που βρέθηκαν σε κάθε εικόνα.



Εικόνα 5-5. Ψηφιακές εικόνες FNA: Καλοήθης (αριστερά) , Κακοήθης (δεξιά)

Πηγή: A. F. M. Agarap, "On breast cancer detection: An application of machine learning algorithms on the Wisconsin diagnostic dataset," *ACM Int. Conf. Proceeding Ser.*, no. 1, pp. 5–9, 2018

Τα χαρακτηριστικά που παρατηρήθηκαν για κάθε στιγμιότυπο είναι:

1. Η ακτίνα (radius)
2. Η υφή (texture)
3. Η περίμετρος (perimeter)
4. Η επιφάνεια (area)
5. Η ομαλότητα (smoothness)
6. Το συμπαγές (compactness= perimeter^2/area - 1)
7. Το κοίλωμα (concavity)

8. Τα σημεία του κοιλώματος (concave points)
9. Η συμμετρία (symmetry)
10. Η διάσταση του μορφοκλασματικού συνόλου[30] (fractal_dimension)

Για κάθε ένα από αυτά τα χαρακτηριστικά έχουν καταγραφεί ο μέσος όρος (mean), το πρότυπο λάθος (standard error -se) και το χειρότερο (worst), δηλαδή ο μέσος όρος των τριών μεγαλύτερων τιμών, καθώς και η αντίστοιχη διάγνωση ως καλοήθης (benign) ή κακοήθης (malignant). Έτσι, κάθε παράδειγμα έχει 30 χαρακτηριστικά.

Με τη βοήθεια των βιβλιοθηκών pandas, matplotlib και seaborn θα περιγράψουμε τα περιεχόμενα του συνόλου δεδομένων και θα οπτικοποιήσουμε τις μεταξύ τους συσχετίσεις στις επόμενες παραγράφους. Το σχετικό αρχείο Colab, είναι το Breast_Cancer_Preprocess.ipynb.

5.3.1 Περιγραφή του Συνόλου Δεδομένων

Στο αρχείο υπάρχουν 569 σειρές και 32 στήλες με επικεφαλίδες:

1	ID	9	symmetry_ mean	17	smoothness_ se	25	perimeter_ worst
2	diagnosis	10	concavity_mean	18	compactness_ se	26	area_worst
3	radius_mean	11	concave points_ mean	19	concavity_se	27	smoothness_ worst
4	texture_mean	12	fractal dimension_ mean	20	concave points_se	28	compactness_ worst
5	perimeter_mean	13	radius_se	21	symmetry_se	29	concavity_worst
6	area_mean	14	texture_se	22	fractal dimension_se	30	concave points_ worst
7	smoothness_ mean	15	perimeter_se	23	radius_worst	31	symmetry_worst

[30] Ο όρος fractal - μορφοκλασματικό σύνολο, χρησιμοποιείται για γεωμετρικά σχήματα που επαναλαμβάνονται αυτούσια σε άπειρο βαθμό μεγέθυνσης.

8	compactness_mean	16	area_se	24	texture_worst	32	fractal dimension_worst

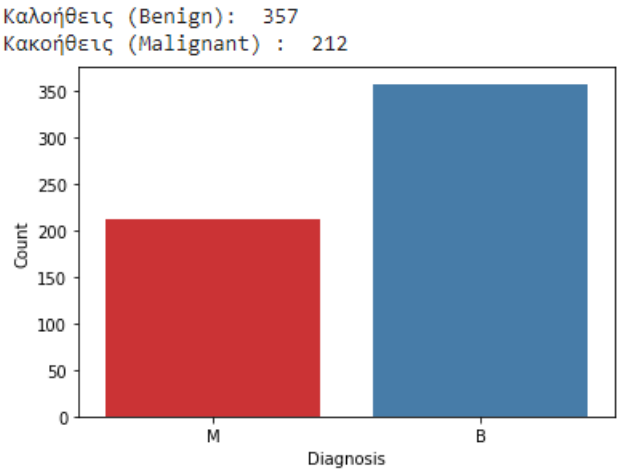
Η τιμή του πεδίου ID είναι ακέραιος αριθμός, η τιμή του πεδίου diagnosis είναι τύπου object που παίρνει τιμές B (Benign) ή M (Malignant) και οι υπόλοιπες τιμές είναι πραγματικοί αριθμοί.
Από το σύνολο δεδομένων δεν λείπει καμία τιμή και προφανώς δεν μας ενδιαφέρει η στήλη ID. Συνεπώς, για να πειραματιστούμε με το μοντέλο μας, έχουμε στη διάθεσή μας ένα σύνολο δεδομένων με 569 παραδείγματα. Κάθε παράδειγμα έχει 30 χαρακτηριστικά και μία ετικέτα για τη διάγνωση.

5.3.2 Εξερεύνηση Συνόλου Δεδομένων

Η εξερευνητική ανάλυση των δεδομένων (Exploratory Data Analysis – EDA) είναι εκτός σκοπού της παρούσας ΜΔΕ. Παρόλα αυτά, αξίζει να δούμε κάποιες από τις δυνατότητες που προσφέρουν οι βιβλιοθήκες οπτικοποίησης στατιστικών κατανομών και συσχετίσεων της Python και ιδιαίτερα η βιβλιοθήκη seaborn. Από τα γραφήματα, ακόμη και κάποιος που δεν είναι ειδικός, μπορεί να αποκτήσει μια αίσθηση για το πώς συνδέονται τα χαρακτηριστικά με τη διάγνωση.
Επίσης, αξίζει εδώ να σημειώσουμε τα εξής: στην ΜΜ για περιπτώσεις όπου έχουμε σύνολα δεδομένων με πολλά χαρακτηριστικά, συνηθίζεται να γίνεται με διάφορους αλγόριθμους μη επιβλεπόμενης μάθησης, όπως για παράδειγμα ο αλγόριθμος της ανάλυσης των κύριων συστατικών (Principal Component Analysis – PCA), εξαγωγή των χαρακτηριστικών (feature extraction) για μείωση των διαστάσεων των παραδειγμάτων και με σκοπό τη διατήρηση των χαρακτηριστικών που έχουν ιδιαίτερη σημασία για τον τελικό στόχο, και συνήθως για αυτό τον λόγο από έναν αναλυτή δεδομένων γίνεται η οπτικοποίηση και εξετάζονται τα στατιστικά. Όπως έχει αναφερθεί οι αλγόριθμοι της ΒΜ καθιστούν συνήθως περιττή αυτή τη διαδικασία, γιατί ο αλγόριθμοι είναι τέτοιοι που, οι ίδιοι πραγματοποιούν την εξαγωγή των χαρακτηριστικών [24], [25], ειδικά για μικρά σύνολα δεδομένων όπως αυτό που εξετάζουμε. Σε παρακάτω ενότητα θα παραθέσουμε πειραματική μελέτη για αυτές τις τεχνικές εξαγωγής χαρακτηριστικών του συνόλου δεδομένων μας.

5.3.2.1 Κατανομή των διαγνώσεων

Το σύνολο δεδομένων περιέχει 569 εγγραφές από ισάριθμες παρατηρήσεις οι οποίες αφορούν 357 περιπτώσεις όπου διαγνώστηκε καλοήθης όγκος (Benign – B) και 212 περιπτώσεις όπου διαγνώστηκε κακοήθης όγκος (Malignant- M), όπως φαίνεται στην Εικόνα 5-6.



Εικόνα 5-6. Αριθμητική κατανομή των περιπτώσεων των όγκων

Η απεικόνιση αυτή για ένα σύνολο δεδομένων είναι πάντα χρήσιμη για να δούμε εάν ένα σύνολο είναι σχετικά ισορροπημένο. Εάν για παράδειγμα υπήρχαν 10 παραδείγματα με ετικέτα M, προφανώς αυτό το σύνολο δεν θα ήταν κατάλληλο να χρησιμοποιηθεί για να δημιουργήσουμε ένα αξιόπιστο μοντέλο. Το συγκεκριμένο σύνολο είναι ένα από τα πλέον χρησιμοποιούμενα από το UCI για δοκιμές σε αλγόριθμους ΜΜ που αφορούν το πρόβλημα της ταξινόμησης.

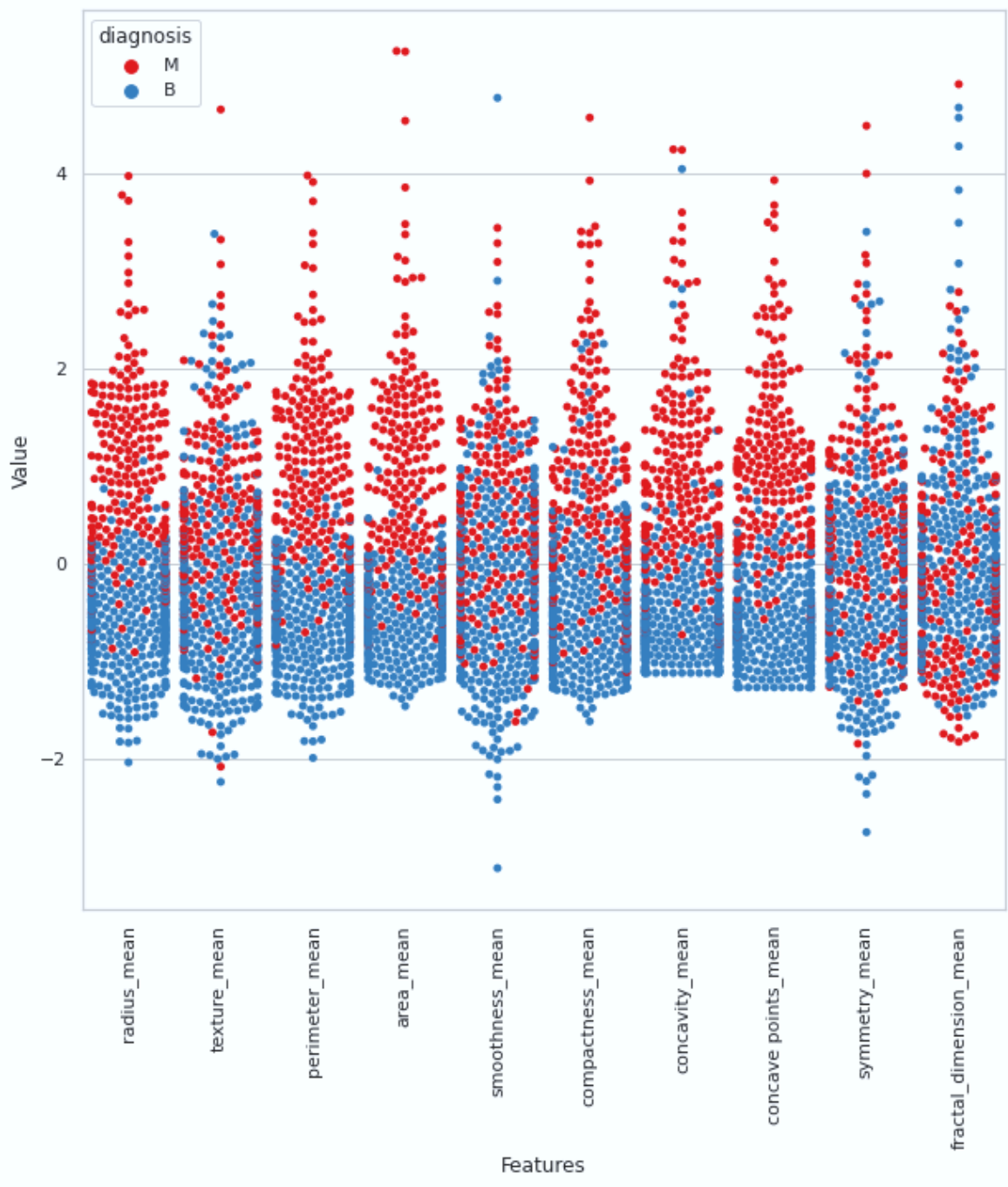
5.3.2.2 Εξερεύνηση των χαρακτηριστικών σε σχέση με την διάγνωση

Το γράφημα της seaborn swarm plot (γράφημα σμήνους) είναι ένα scatter plot (γράφημα διασκορπισμού) κατηγοριοποίησης με μη επικαλυπτόμενα σημεία. Τα σημεία προσαρμόζονται (μόνο κατά μήκος του άξονα της κατηγορίας) έτσι ώστε να μην αλληλεπικαλύπτονται. Αυτό δίνει μια καλύτερη αναπαράσταση της κατανομής των τιμών.

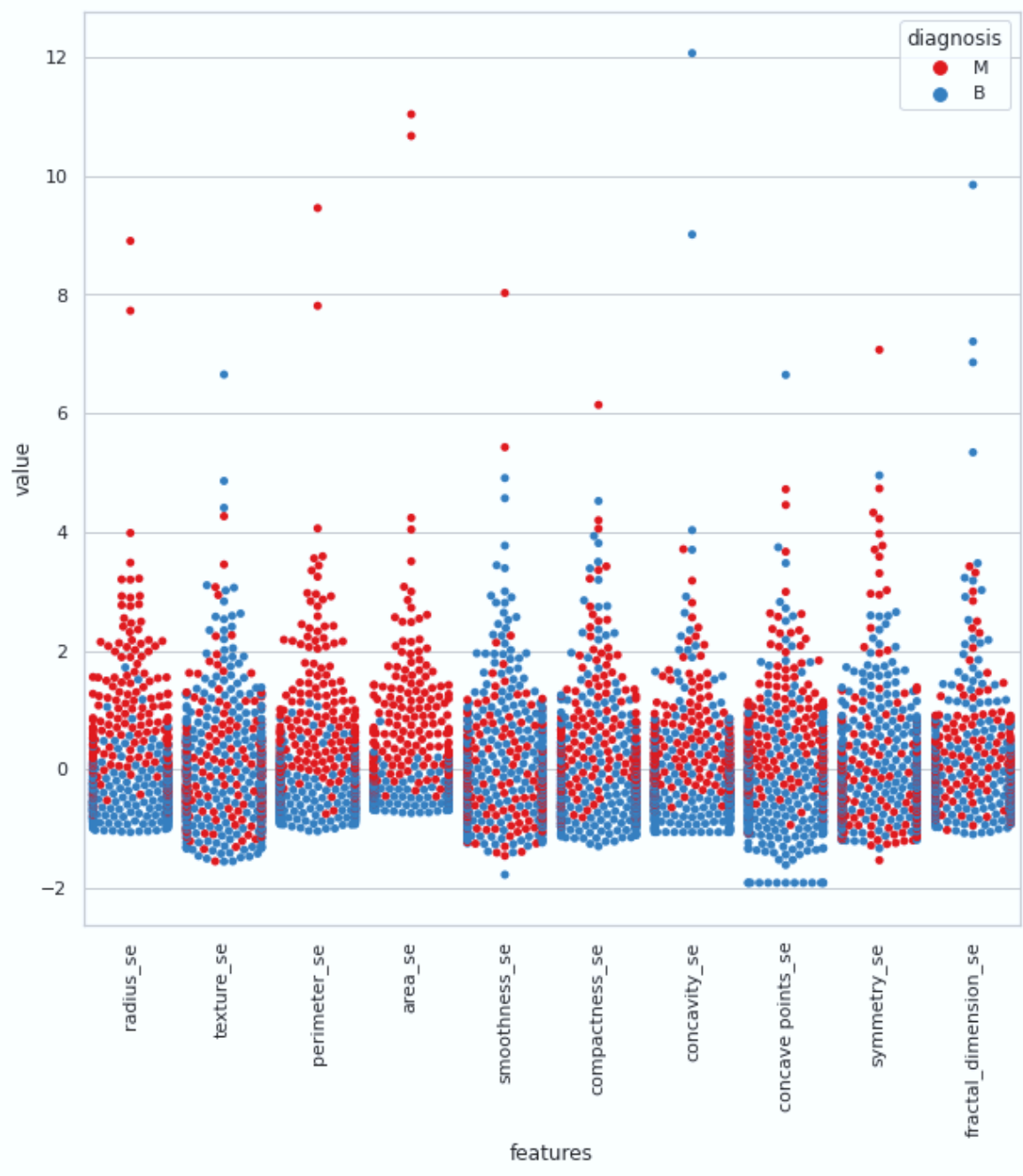
Να σημειώσουμε εδώ ότι, πριν τη δημιουργία των γραφημάτων θα πρέπει να γίνει μια κανονικοποίηση των τιμών, ώστε να βρίσκονται όλες σε κάποιο συγκεκριμένο εύρος. Περισσότερα για την κανονικοποίηση και γιατί πρέπει να γίνεται πάντα, θα δούμε στο επόμενο κεφάλαιο.

Στην Εικόνα 5-7 δίνονται τα swarm plots για τις κατανομές των mean τιμών των χαρακτηριστικών ανάλογα με τη διάγνωση. Στην Εικόνα 5-8 δίνονται τα swarm plots για τις κατανομές των standard

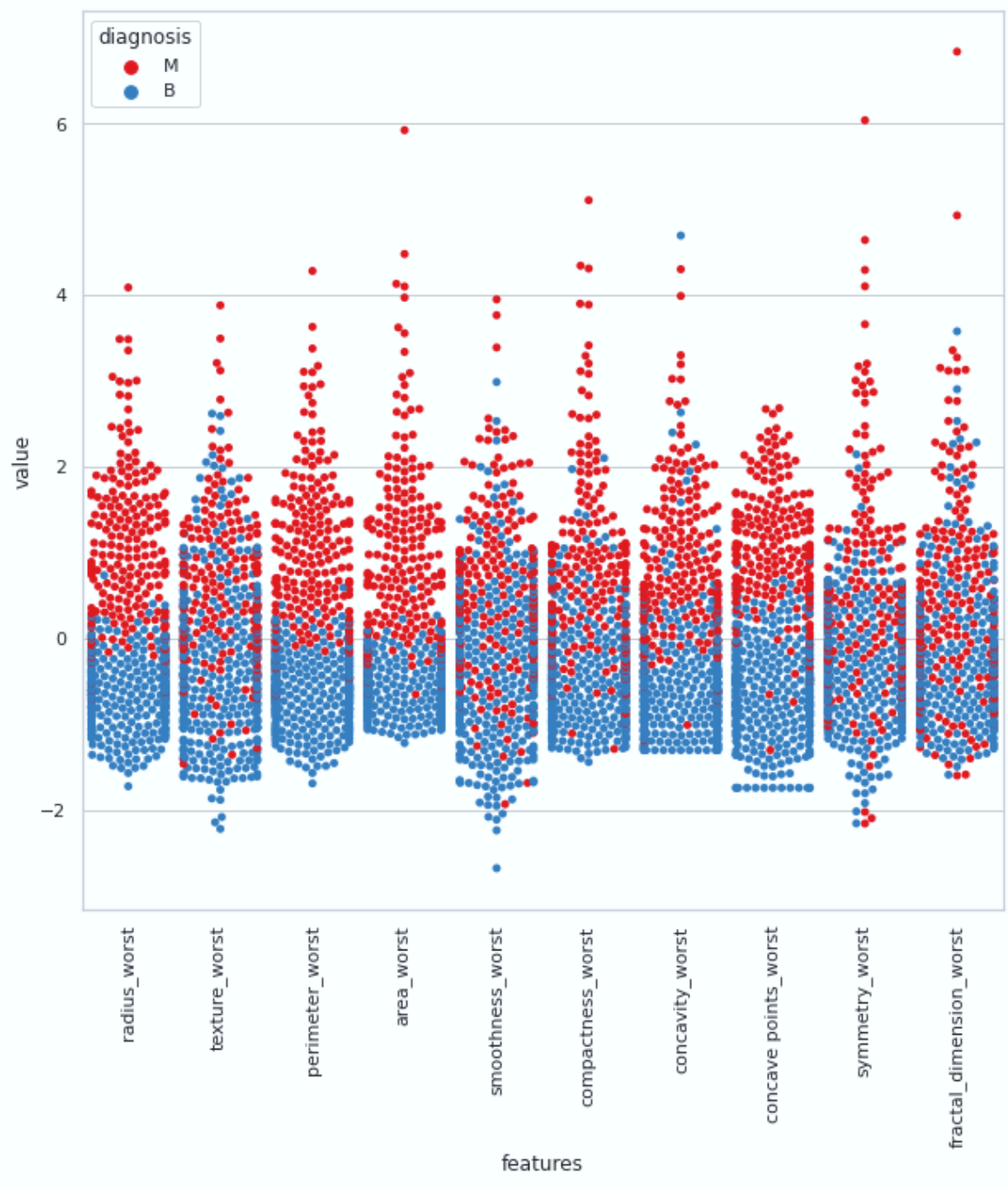
error τιμών των χαρακτηριστικών ανάλογα με τη διάγνωση. Στην Εικόνα 5-9 δίνονται τα swarm plots για τις κατανομές των worst τιμών των χαρακτηριστικών ανάλογα με τη διάγνωση.
Με μια πρώτη ματιά σε κάθε μία από αυτές τις εικόνες, μπορεί ο καθένας να αντιληφθεί άμεσα, για παράδειγμα, πόσο πιο ξεκάθαρη είναι η διασπορά για την ακτίνα, σε αντίθεση με το fractal και οι ειδικοί να βγάλουν πολύ περισσότερα συμπεράσματα για τις κατανομές.



Εικόνα 5-7. Κατανομή mean των χαρακτηριστικών με swarm plots



Εικόνα 5-8. Κατανομή standard error των χαρακτηριστικών με swarm plots

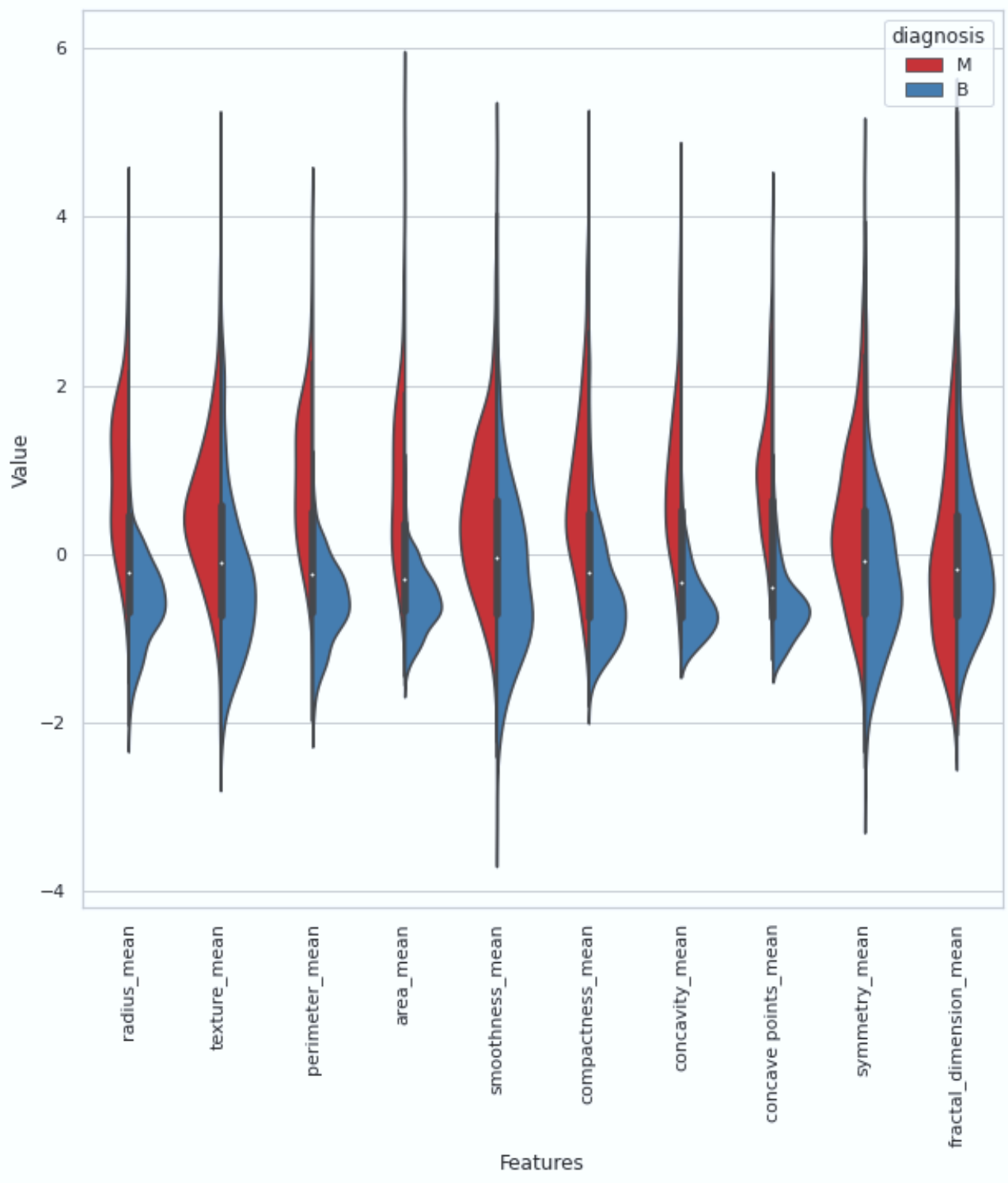


Εικόνα 5-9. Κατανομή worst των χαρακτηριστικών με swarm plots

Ένας άλλο είδος γραφήματος που χρησιμοποιείται συχνά είναι το τύπου violin (βιολί), που δείχνει τη στατιστική κατανομή (distribution) των ποσοτικών δεδομένων σε διάφορα επίπεδα μιας (ή περισσότερων) κατηγορηματικών μεταβλητών, έτσι ώστε αυτές οι κατανομές να μπορούν να

συγκριθούν. Είναι επίσης ένας αποτελεσματικός τρόπος για την εμφάνιση πολλαπλών διανομών δεδομένων ταυτόχρονα, ελκυστικός όμως για αυτούς που γνωρίζουν από στατιστική.

Δίνουμε ενδεικτικά τα violin plot για τα mean των χαρακτηριστικών στην Εικόνα 5-10.



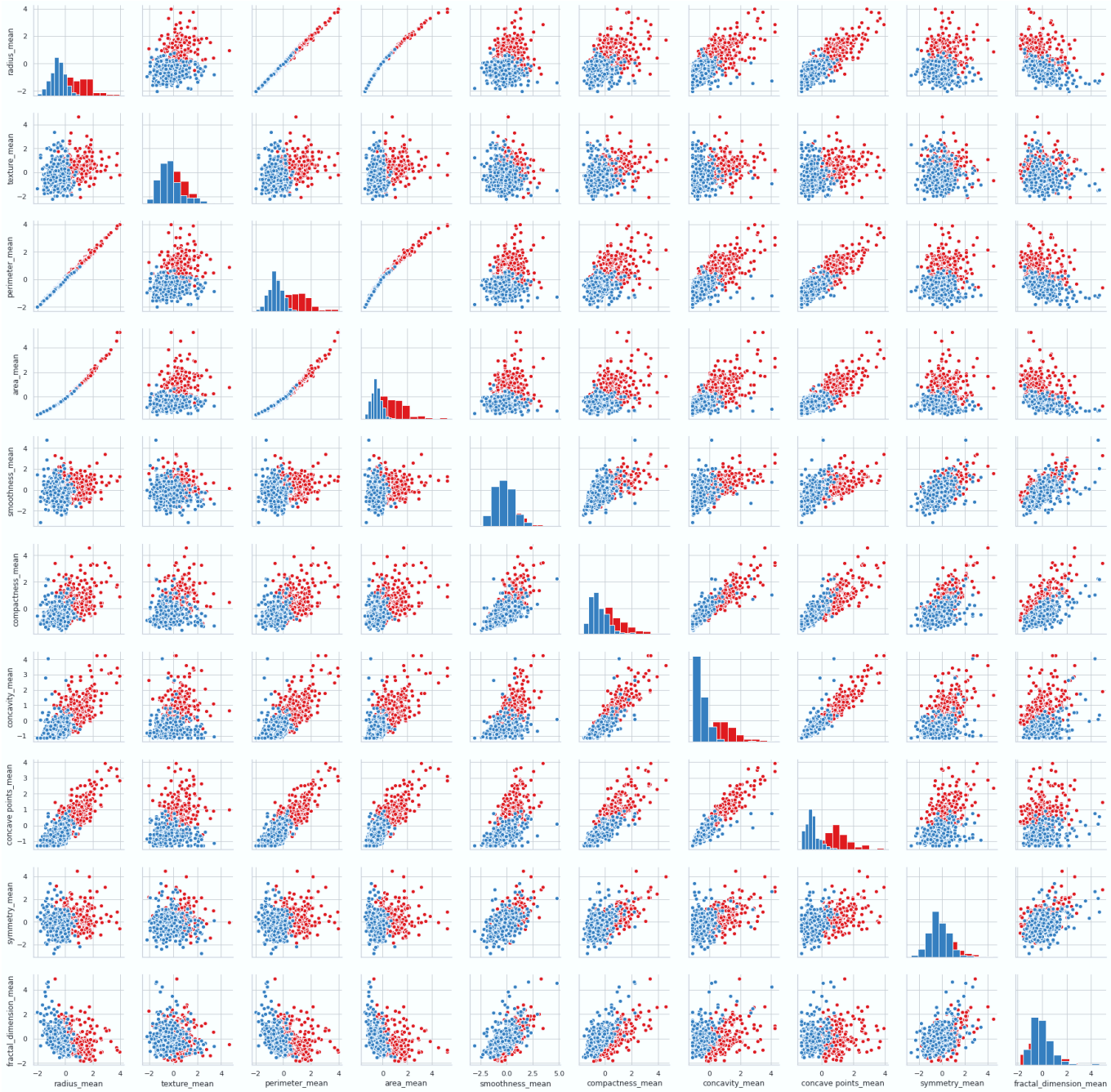
Εικόνα 5-10. Στατιστική κατανομή των mean των χαρακτηριστικών με violin plot

5.3.3 Εξερεύνηση της Συσχέτισης των Χαρακτηριστικών

Για τη συσχέτιση των χαρακτηριστικών εξετάζουμε δύο επιλογές: τη σύγκριση κατά ζεύγη με στατιστικές κατανομές και τη σύγκριση των αριθμητικών τιμών όλων των χαρακτηριστικών.

5.3.3.1 Συσχέτιση χαρακτηριστικών ανά ζεύγη

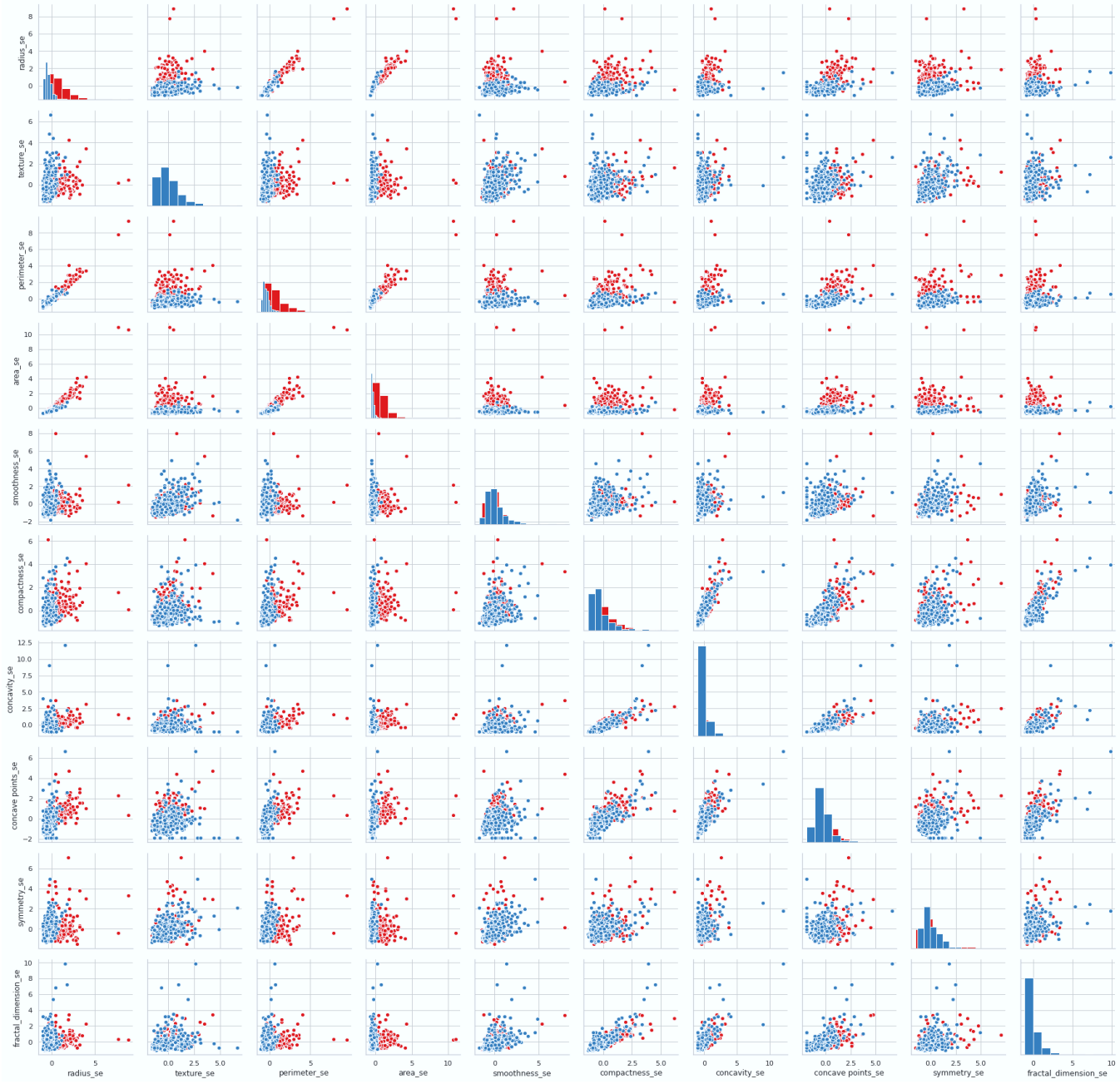
Για τη συσχέτιση των χαρακτηριστικών ανά ζεύγη η seaborn διαθέτει τα γραφήματα pair grid (πλέγμα ζευγών). Το pair grid γράφημα χαρτογραφεί κάθε μεταβλητή σε ένα σύνολο δεδομένων σε μια στήλη και μια σειρά σε ένα πλέγμα πολλαπλών αξόνων. Μπορούν να χρησιμοποιηθούν διαφορετικές λειτουργίες σχεδίασης επιπέδου αξόνων για τη σχεδίαση διμετατροπών σε άνω και κάτω τρίγωνα και η οριακή κατανομή κάθε μεταβλητής μπορεί να εμφανιστεί στη διαγώνιο, όπου επιλέξαμε να φαίνονται τα ιστογράμματα.



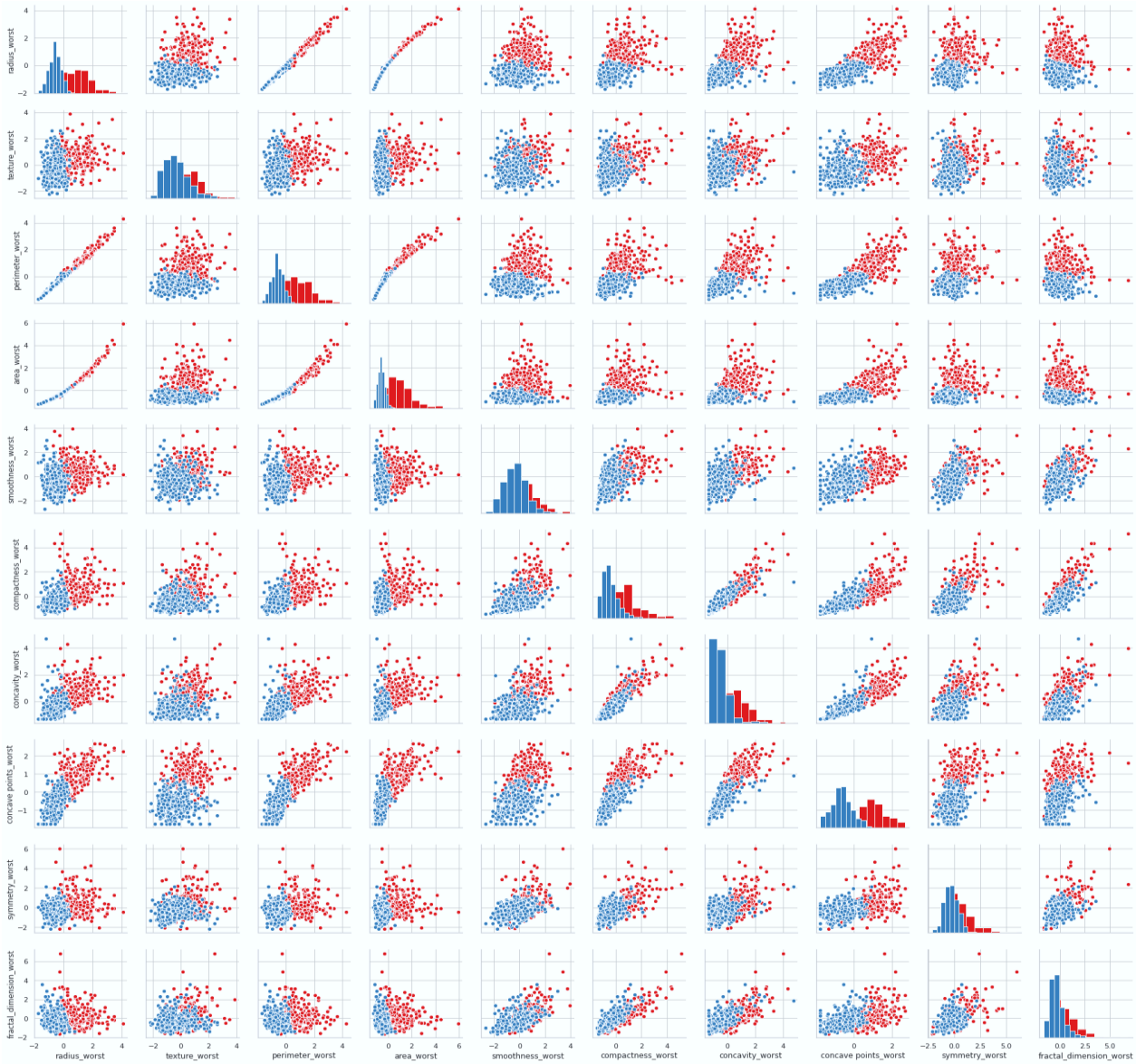
Εικόνα 5-11. Κατανομή mean των χαρακτηριστικών με pair grid plots

Στην Εικόνα 5-11 δίνονται τα pair grid plots για τις κατανομές των mean τιμών των χαρακτηριστικών ανάλογα με τη διάγνωση. Στην Εικόνα 5-12 δίνονται τα pair grid plots για τις κατανομές των standard error τιμών των χαρακτηριστικών ανάλογα με τη διάγνωση.

Στην Εικόνα 5-13 δίνονται τα pair grid plots για τις κατανομές των worst τιμών των χαρακτηριστικών ανάλογα με τη διάγνωση.



Εικόνα 5-12. Κατανομή standard error των χαρακτηριστικών με pair grid plots



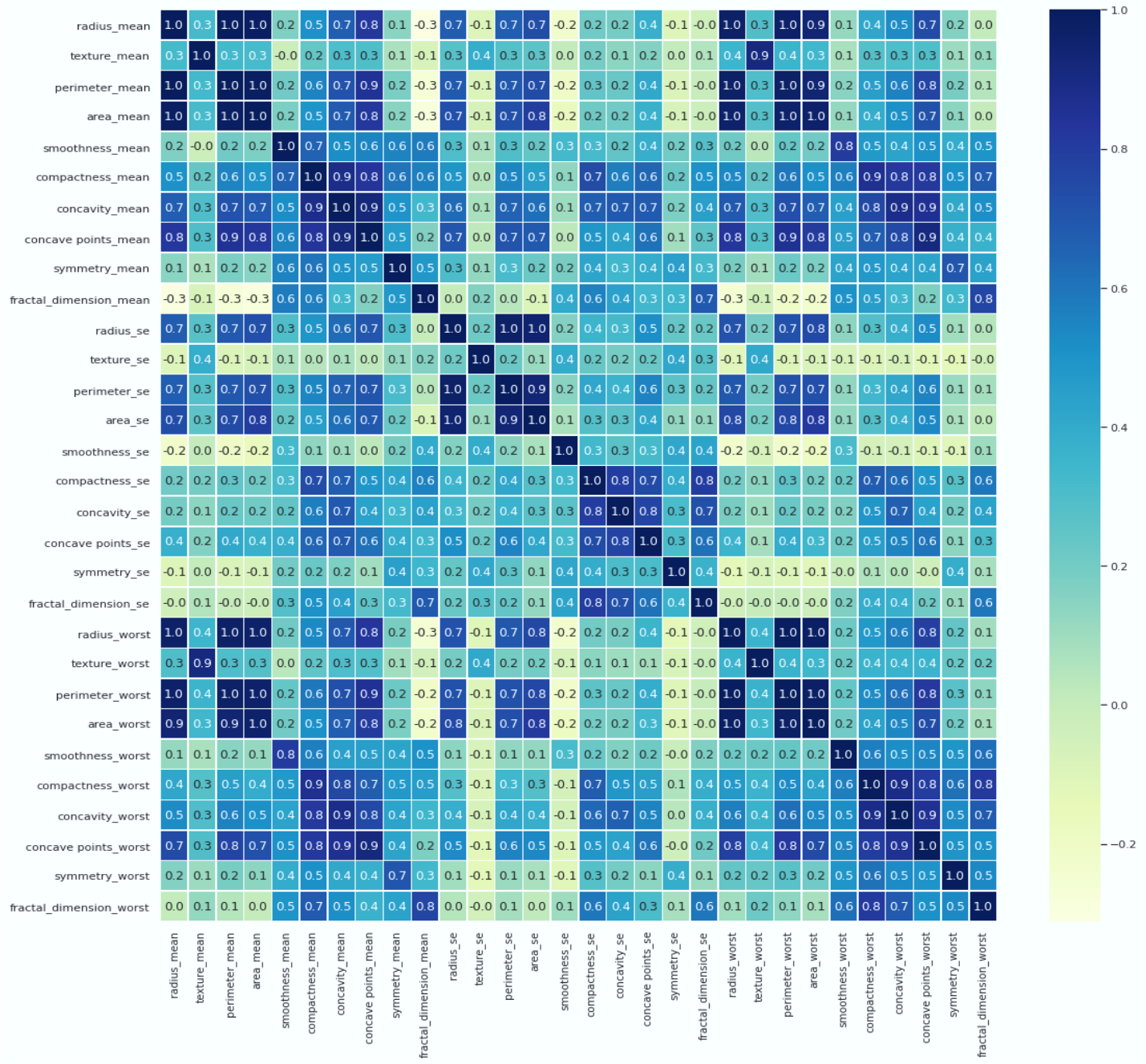
Εικόνα 5-13. Κατανομή worst των χαρακτηριστικών με pair grid plots

Από τα αυτά γραφήματα, είναι φανερό ότι, υπάρχει ισχυρή συσχέτιση χαρακτηριστικών ανά δύο μεταξύ πολλών χαρακτηριστικών, όπως, για παράδειγμα, μεταξύ της ακτίνας, της περιμέτρου και της επιφάνειας.

Στη συνέχεια, θα χρησιμοποιήσουμε έναν άλλο τύπο γραφήματος, το μητρώο θερμότητας (heatmap matrix) για να δείξουμε τις αριθμητικές συσχετίσεις μεταξύ όλων των χαρακτηριστικών.

5.3.3.2 Συσχέτιση μεταξύ όλων των χαρακτηριστικών

Ο heatmap matrix (μητρώο θερμότητας), γνωστός και ως πίνακας συσχέτισης (correlation matrix), δημιουργείται μεταξύ όλων των 30 χαρακτηριστικών όπως φαίνεται στην Εικόνα 5-14.



Εικόνα 5-14. Μητρώο συσχέτισης (heatmap matrix)

Οι τιμές της συσχέτισης έχουν πεδίο τιμών από -1 έως 1. Τιμή πιο κοντά στο 1 σημαίνει ότι, τα χαρακτηριστικά είναι απόλυτα συσχετισμένα και ως συμπέρασμα εξάγεται ότι, τα χαρακτηριστικά εξαρτώνται το ένα από το άλλο θετικά, ενώ για αρνητικές τιμές κοντά στο -1 συμπεραίνεται ότι, τα χαρακτηριστικά είναι εξαρτώμενα μεταξύ τους, ενώ για τιμές κοντά στο 0 είναι ανεξάρτητα μεταξύ τους. Πιο συγκεκριμένα, έχουμε τιμές οι οποίες είναι στο 0,9 που σημαίνει ότι τα χαρακτηριστικά έχουν πολύ ισχυρή συσχέτιση αναμεταξύ τους και 0,8 υψηλή συσχέτιση αναμεταξύ τους(σχετική αναφορά στο αρχείο Breast_Cancer_Preprocess.ipynb). Στη διαγώνιο του πίνακα οι τιμές είναι 1, πράγμα που σημαίνει ότι υπάρχει απόλυτη συσχέτιση, διότι η σύγκριση γίνεται μεταξύ του χαρακτηριστικού και του εαυτού του.

Έτσι, για παράδειγμα, μπορούμε να δούμε ότι, τα κακοήθη κύτταρα δείχνουν να έχουν μεγαλύτερες τιμές για την ακτίνα, την περίμετρο, την επιφάνεια, το πόσο συμπαγή είναι, την κοιλότητα και τα σημεία του κοιλότητας. Αυτό σημαίνει ότι, υπάρχει απόλυτη συσχέτιση μεταξύ της ακτίνας, της περιμέτρου και της επιφάνειας, όπως άλλωστε αναμενόταν από την μεταξύ τους σχέση και, επίσης, υπάρχει απόλυτη συσχέτιση μεταξύ των mean του πόσο συμπαγή είναι, της κοιλότητας και του σημείου της κοιλότητας.

5.4 Σχετική Βιβλιογραφία

Το σύνολο δεδομένων WBCD έχει χρησιμοποιηθεί από πάρα πολλούς ερευνητές για την υλοποίηση μοντέλων με βάση αλγόριθμους ΜΜ και τις βιβλιοθήκες της γλώσσας προγραμματισμού Python, όπως για παράδειγμα στις εργασίες [7], [8]. Οι περισσότερες εργασίες αφορούν υλοποιήσεις με τη βιβλιοθήκη scikit-learn και υπάρχουν σχετικά λίγες με τη βιβλιοθήκη TensorFlow και την Keras, οι οποίες είναι νεότερες της scikit-learn, και ακόμα λιγότερες με υλοποιήσεις που αφορούν μόνο ΤΝΔ, και πιο συγκεκριμένα MLPs. Αξίζει να σημειωθεί ότι, σε ελάχιστες εργασίες είδαμε μεθόδους εξομάλυνσης (πλην της k-fold validation) και ρύθμιση των υπερπαραμέτρων με τις μεθόδους που προσφέρουν οι scikit-learn και η TensorFlow.

Για να μπορέσουμε να συγκρίνουμε την εργασία μας με άλλες εργασίες, επιλέξαμε 3 εργασίες, δημοσιευμένες σε διεθνείς και αξιόπιστες επιστημονικές βιβλιοθήκες και όπου υπήρχε όσο το δυνατόν περισσότερη πληροφορία για τα μοντέλα που υλοποίησαν οι συγγραφείς και που υλοποιήθηκαν με την TensorFlow και backend τη βιβλιοθήκη Keras.

Η πρώτη εργασία που επιλέξαμε είναι αυτή του Agarap [60], με τίτλο *«Deep Learning using Rectified Linear Units (ReLU)»*

Στην εργασία αυτή ο συγγραφέας συγκρίνει την απόδοση διάφορων μοντέλων νευρωνικών δικτύων σε τρία σύνολα δεδομένων εφαρμόζοντας διάφορες συναρτήσεις ενεργοποίησης στα κρυφά επίπεδα, χρησιμοποιώντας ως βελτιστοποιητή τον Adam για τις παραμέτρους εκμάθησης. Ένα από αυτά τα σύνολα δεδομένων είναι και το WDBC, υλοποίησε ένα μοντέλο δύο κρυφών επιπέδων με 64 και 32 νευρώνες αντίστοιχα. Για μέθοδο εξομάλυνσης επιλέχθηκε η 10-fold validation και δεν εφάρμοσε την μείωση διαστάσεων των χαρακτηριστικών με PCA.

Τα αποτελέσματα δίνονται από τον συγγραφέα σε ένα πίνακα, όπως φαίνεται στην Εικόνα 5-17.

Metrics / Models	FFNN-Softmax	FFNN-ReLU
Training cross validation	≈ 91.21%	≈ 87.96%
Test accuracy	≈ 92.40%	≈ 90.64%
Precision	0.92	0.91
Recall	0.92	0.91
F1-score	0.92	0.90

Εικόνα 5-15. Αποτελέσματα εργασίας [60]

Να σημειωθεί ότι, πριν από αυτή την εργασία, ο συγγραφέας για το WDBC σύνολο δεδομένων παρουσίασε σε παλαιότερη εργασία [7], συγκριτικά αποτελέσματα για διάφορους αλγόριθμους μηχανικής μάθησης, μεταξύ αυτών και τον MLP, όπου είχε υλοποιήσει ένα μοντέλο με τρία κρυφά επίπεδα με 500 νευρώνες ανά επίπεδο με την TensorFlow. Ως βελτιστοποιητή επέλεξε τον Adam με ρυθμό μάθησης 0.01, η εκπαίδευση διήρκησε 3000 εποχές με batch_size= 128 και διαχωρισμό 70-30 με μέθοδο εξομάλυνσης k-fold validation. Ο συγγραφέας διεπίστωσε ότι ο αλγόριθμος MLP είχε τα καλύτερα αποτελέσματα σε σχέση με τους υπόλοιπους αλγόριθμους ΜΜ. Πληροφορικά, δίνουμε τα αποτελέσματα της εργασίας [7] στην Εικόνα 5-16 για να δείξουμε την υπεροχή του MLP , τα οποία όμως δεν θα συγκρίνουμε γιατί αφορούν παλαιότερη έκδοση του TensorFlow.

Parameter	GRU-SVM	Linear Regression	MLP	L1-NN	L2-NN	Softmax Regression	SVM
Accuracy	93.75%	96.09375%	99.038449585420729%	93.567252%	94.736844%	97.65625%	96.09375%
Data points	384000	384000	512896	171	171	384000	384000
Epochs	3000	3000	3000	1	1	3000	3000
FPR	16.666667%	10.204082%	1.267042%	6.25%	9.375%	5.769231%	6.382979%
FNR	0	0	0.786157%	6.542056%	2.803738%	0	2.469136%
TPR	100%	100%	99.213843%	93.457944%	97.196262%	100%	97.530864%
TNR	83.333333%	89.795918%	98.732958%	93.75%	90.625%	94.230769%	93.617021%

Εικόνα 5-16. Αποτελέσματα εργασίας [7]

Η δεύτερη εργασία που επιλέξαμε είναι αυτή των Hasan, Haque και Kabir [61], με τίτλο, *«Breast Cancer Diagnosis Models Using PCA and Different Neural Network Architectures»"*

Στην εργασία αυτή οι συγγραφείς χρησιμοποιούν δύο σύνολα δεδομένων: το WDBC και το SEER 2017 Breast Cancer Dataset[31] και υλοποιούν δύο τύπους ΤΝΔ: MLP και CNN, αφού προηγουμένως έχουν εφαρμόσει τον αλγόριθμο PCA προκειμένου να μειώσουν τα χαρακτηριστικά από 30 σε 8. Ο διαχωρισμός του συνόλου δεδομένων είναι με αναλογία 80-20. Οι πληροφορίες που δίνονται από τους συγγραφείς για την υλοποίηση του MLP μοντέλου, είναι ότι είναι με ένα κρυφό επίπεδο, η εκπαίδευση διήρκησε 50 εποχές και ως σύνολο επικύρωσης χρησιμοποιήθηκε το σύνολο δοκιμής.

Οι συγγραφείς δίνουν τα αποτελέσματά τους σε ένα πίνακα, όπως φαίνεται στην Εικόνα 5-17.

Evaluation Measure	Neural Network Architecture	
	MLP	CNN
Sensitivity	1.000	0.97
Specificity	0.975	0.950
Precision	0.986	0.972
Accuracy	0.991	0.964
F-score	0.993	0.973
AUC	1.000	0.993

Εικόνα 5-17. Αποτελέσματα εργασίας [61]

Η τρίτη εργασία που επιλέξαμε είναι αυτή των Prakash and Visakha [62], με τίτλο *«Breast Cancer Malignancy Prediction Using Deep Learning Neural Networks»*

Στην εργασία αυτή οι συγγραφείς υλοποιούν ένα MLP μοντέλο με τρία κρυφά επίπεδα και αριθμό νευρώνων 15, 7 και 3 αντίστοιχα. Ως μεθόδους εξομάλυνσης εφαρμόζουν το πρόωρο σταμάτημα και dropout 0.15 σε κάθε επίπεδο. Ο διαχωρισμός του συνόλου δεδομένων είναι σε αναλογία 75-25 και δεν αναφέρουν τη μέθοδο επικύρωσης.

Οι συγγραφείς δίνουν τα αποτελέσματά τους παραθέτοντας τον πίνακα ταξινόμησης και την αναφορά ταξινόμησης, όπως φαίνεται στην Εικόνα 5-17.

[31] https://ieee-dataport.org/open-access/seer-breast-cancer-data

	Predicted 0 : Benign	Predicted 1 : Malignant
Actual 0 : Benign	54	1
Actual 1 : Malignant	1	87

	precision	recall	f1-score	support
0	0.98	0.98	0.98	55
1	0.99	0.99	0.99	88
accuracy			0.99	143
macro avg	0.99	0.99	0.99	143
weighted avg	0.99	0.99	0.99	143

Εικόνα 5-18. Αποτελέσματα εργασίας [62]

6 Υλοποίηση MLPs με Python

Στο κεφάλαιο αυτό παρουσιάζονται οι υλοποιήσεις διάφορων μοντέλων MLPs με τη γλώσσα προγραμματισμού Python και τις συναφείς βιβλιοθήκες σύμφωνα με τη βασική προσέγγιση της δημιουργίας ενός έργου. Αρχικά παρουσιάζεται το στάδιο της εισόδου και της προετοιμασίας των δεδομένων σε μορφή κατάλληλη για τις επόμενες εργασίες που αφορούν την ανάπτυξη ενός MLP μοντέλου. Στη συνέχεια, παρουσιάζεται η γενική ροή των εργασιών για την ανάπτυξη, καθώς και οι αρχικές επιλογές για τις υπερπαραμέτρους της βελτιστοποίησης και της ομαλοποίησης. Σύμφωνα με τις αυτές τις επιλογές, παρουσιάζονται πρώτα για κάθε μοντέλο MLP τα αποτελέσματα της υλοποίησης με πειραματισμό όσον αφορά την επιλογή υπερπαραμέτρων των μοντέλων MLPs όπως αυτή περιγράφεται στη γενική ροή των εργασιών. Τέλος, παρουσιάζονται τα αποτελέσματα για διάφορα μοντέλα με αυτόματη ρύθμιση κάποιων υπερπαραμέτρων και επίσης παρατίθεται η σύγκριση τους.

Πρέπει να σημειωθεί πως, ειδικά στο κεφάλαιο αυτό, γίνεται εκτεταμένη χρήση των αγγλικών όρων, για να υπάρχει όσο το δυνατόν μεγαλύτερη συνέπεια με τις εντολές των βιβλιοθηκών. Επίσης, σημειώνεται ότι για την υλοποίηση αντλήσαμε τις πληροφορίες μόνο από τις επίσημες ιστοσελίδες των βιβλιοθηκών, καθώς και από τρία συγγράμματα που κατά την άποψή μας, αλλά και κατά γενική ομολογία θεωρούνται από τα πλέον αξιόπιστα και συνιστώμενα από μαθήματα κορυφαίων πανεπιστημίων, όπως του MIT [25]. Πιο συγκεκριμένα, βασιστήκαμε στο σύγγραμμα του δημιουργού της Keras, του François Chollet [23] και το σύγγραμμα του Aurélien Géron [22] που συνιστάται από το MIT για το οποίο διατίθεται ελεύθερα ο κώδικας στο GitHub. Πρόθεσή

μας δεν είναι να ανακαλύψουμε ξανά τον τροχό, αλλά να συνδυάσουμε την γνώση που παρέχεται από τις πηγές μας και να αντλήσουμε την απαιτούμενη πληροφορία.

Επίσης, τονίζεται ιδιαιτέρως το γεγονός ότι, γενικά οι αλγόριθμοι της ΜΜ, και κατά συνέπεια και των ΤΝΔ, βασίζονται σε στατιστικές μεθόδους. Συνεπώς, τρέχοντας μια υλοποίηση ενός μοντέλου, τα αποτελέσματα θα διαφέρουν από εκτέλεση σε εκτέλεση. Για τον λόγο αυτό, συνηθίζεται στις ακαδημαϊκές εργασίες να δίνονται οι υλοποιήσεις σε notebook, όπου φαίνονται και οι έξοδοι από μία συγκεκριμένη εκτέλεση μιας ή περισσότερων εντολών που περιέχονται στο notebook. Τέλος, σημειώνουμε ότι, δόθηκε ιδιαίτερη βαρύτητα στην υλοποίηση μοντέλων με αυτόματη ρύθμιση παραμέτρων.

6.1 Είσοδος και Επεξεργασία Συνόλου Δεδομένων

Η είσοδος και η επεξεργασία των δεδομένων είναι κοινή για κάθε MLP μοντέλο που υλοποιήθηκε και έχει πραγματοποιηθεί με την βοήθεια των βιβλιοθηκών scikit learn και pandas. Η διαδικασία, καθώς και οι αντίστοιχες εντολές συνοψίζονται στον Πίνακα 6-1.

Επιλέξαμε σε αυτή τη φάση να εισάγουμε το αρχείο δεδομένων από την scikit learn, εφόσον διατίθεται από εκεί σε μορφή που μας απαλλάσσει από κάποιες επιπλέον εργασίες, όσον αφορά τον διαχωρισμό των τανυστών για τα παραδείγματα και τις ετικέτες, καθώς και τη μετατροπή των ετικετών σε δυαδική μορφή.

Ειδικά για τα μοντέλα των MLP, και για παραδείγματα όπου συμπεριλαμβάνεται ένας μεγάλος αριθμός χαρακτηριστικών που παίρνουν πραγματικές τιμές πολύ μικρές ή πολύ μεγάλες, σύμφωνα με τους [1], [22], [23], είναι απαραίτητος ο περιορισμός των αριθμητικών τιμών των χαρακτηριστικών σε κάποιο εύρος αριθμητικών τιμών και συνεπώς απαιτείται κάποια προεργασία (preprocessing) για αυτή την κλιμάκωση.

Εάν ένα χαρακτηριστικό έχει διακύμανση που είναι τάξεις μεγέθους μεγαλύτερη από άλλες, μπορεί να κυριαρχήσει στην συνάρτηση κόστους με αποτέλεσμα το μοντέλο να μην μπορεί να μάθει από άλλα χαρακτηριστικά όπως αναμενόταν. Εάν δεν γίνει η κλιμάκωση, θα κυριαρχήσουν στο μοντέλο οι μεγάλες τιμές και όλες οι αποφάσεις θα είναι μεροληπτικές προς αυτές. Έτσι, με τη βοήθεια της scikit-learn επιλέγουμε να κανονικοποιήσουμε τα χαρακτηριστικά του συνόλου των δεδομένων των παραδειγμάτων με μια απλή και γρήγορη συνάρτηση, την scale. Τα

κλιμακωτά δεδομένα έχουν μέση τιμή (mean) ίση με 0 και τυπική απόκλιση (standard deviation) ίση με 1.

Όσον αφορά τον διαχωρισμό του συνόλου δεδομένων σε σύνολα εκπαίδευσης και δοκιμής, χρησιμοποιούμε τη συνάρτηση train_test_split της scikit-learn. Εφόσον το σύνολο μας θεωρείται μικρό, επιλέγουμε ως παράμετρο το ποσοστό του συνόλου δοκιμής στο 20% επί του συνόλου (test_size= 0.20). Επίσης, επιλέγουμε μια άλλη παράμετρο που ορίζει τον ‘’σπόρο’’ (seed) που χρησιμοποιείται για τη γεννήτρια των τυχαίων αριθμών, την random_state= 20, η οποία μπορεί να λάβει τιμές από 1-42. Εάν δεν οριστεί αυτή η παράμετρος, κάθε φορά που θα καλούμε τη συνάρτηση train_test_split θα παίρνουμε διαφορετικά σύνολα εκπαίδευσης και δοκιμής, κάτι το οποίο δεν είναι επιθυμητό για τη σύγκριση των μοντέλων.

Στον Πίνακα 6-1 συνοψίζονται οι επιλογές, καθώς και οι αντίστοιχες εντολές για την είσοδο και την προετοιμασία των δεδομένων.

Εργασία	**Είσοδος και προετοιμασία δεδομένων (scikit-learn)**

Είσοδος αρχείου δεδομένων από url
–
Ανάγνωση δεδομένων από το .csv αρχείο
	`import pandas as pd`
	`from urllib.request import urlopen`
	`from numpy import loadtxt`
	`from sklearn.datasets import load_breast_cancer`
	`cancer = pd.read_csv('http://www.jetkite.com/ai/data/breast_cancer_data.csv')`

Ορισμός τανυστών χαρακτηριστικών και στόχων	`df1 = pd.DataFrame(cancer.data)` `df2 = pd.DataFrame(cancer.target)`

Προετοιμασία αριθμητικών δεδομένων χαρακτηριστικών	`from sklearn import preprocessing` `preprocessing.scale(df1)`

Δημιουργία συνόλων εκπαίδευσης (train) και δοκιμής (test)	`from sklearn.model_selection`
	`import train_test_split`
	`X_train, X_test, y_train, y_test =`
	`train_test_split(df1, df2, test_size=0.20, random_state=20)`

Πίνακας 6-1. Είσοδος αρχείου δεδομένων και προετοιμασία δεδομένων.

Εφόσον έχουμε ολοκληρώσει τις φάσεις της εισόδου και της προετοιμασίας, προχωρούμε στις επόμενες φάσεις που αφορούν την επιλογή και την ανάπτυξη του μοντέλου.

6.2 Επιλογές Ανάπτυξης Μοντέλων

Γενικά, η ανάπτυξη μοντέλων ΒΜ, εκτός από τις παραμέτρους μάθησης (βάρη και πολώσεις) αφορά την επιλογή ενός πλήθους υπερπαραμέτρων, όπως την χωρητικότητά τους, τις συναρτήσεις ενεργοποίησης των κρυφών επιπέδων, τους βελτιστοποιητές, την εξομάλυνση κλπ., με στόχο την βελτιστοποίηση και την γενίκευση του μοντέλου.

Σύμφωνα με όσα αναφέραμε σε προηγούμενες παραγράφους, επιλέξαμε τις διάφορες υπερπαραμέτρους και οι επιλογές συνοψίζονται στον Πίνακα 6-2. Πιο συγκεκριμένα οι επιλογές έγιναν με τα παρακάτω κριτήρια:

- Σύμφωνα με την παράγραφο 4.3, συνήθως ένα ή δύο κρυφά επίπεδα είναι αρκετά για την επίλυση ενός προβλήματος με MLP. Επομένως, θα πειραματιστούμε με MLP ενός και δύο κρυφών επιπέδων. Η πρόκληση εδώ είναι ο βέλτιστος αριθμός νευρώνων ανά επίπεδο. Αποφασίσαμε να πειραματιστούμε πρώτα θέτοντας τον αριθμό των κρυφών επιπέδων και των νευρώνων «χειρωνακτικά» θέτοντας κάθε φορά διάφορες άλλες υπερπαραμέτρους και στη συνέχεια να αναθέσουμε στην Keras Tuner την εύρεσή του αριθμού των νευρώνων ανά κρυφό επίπεδο, δηλαδή να δημιουργήσουμε υπερμοντέλα.
- Όσον αφορά τις υπερπαραμέτρους της βελτιστοποίησης που αναφέρονται στις παραγράφους 4.5, 4.6, 4.7, οι επιλογές μας εξαρτώνται από το πρόβλημα (δυαδική

ταξινόμηση), αλλά και τις συστάσεις από τις πηγές μας. Εδώ, η κυριότερη πρόκληση, σύμφωνα με τις πηγές μας, είναι ο ρυθμός μάθησης του βελτιστοποιητή, όπου οι επιλογές είναι πάλι δύο: είτε με πειραματισμούς, είτε με τη βοήθεια της Keras Tuner.

- Σύμφωνα με την παράγραφο 4.9, η βελτιστοποίηση και η εξομάλυνση έχουν την ίδια βαρύτητα στην ανάπτυξη του μοντέλου. Επομένως, θα πειραματιστούμε σε μεθόδους εξομάλυνσης που διατίθενται από την Keras και προτείνονται από τις πηγές. Όμως, επιλέγουμε από την αρχή να εφαρμόσουμε σε όλες τις περιπτώσεις το πρόωρο σταμάτημα, οπότε η υπερπαράμετρος για τις εποχές δεν χρειάζεται να οριστεί εξ ’αρχής, παρά μόνο το batch size.

Φάση	**Υπερπαράμετρος ή Ενέργεια**	**Επιλογή**	**Παρατηρήσεις**
Σχεδίαση	Αρχιτεκτονική	MLP με ένα ή δύο κρυφά επίπεδα	1. Πειραματισμός με δύο κρυφά επίπεδα για απλά μοντέλα 2. Επιλογή αριθμού νευρώνων από την Keras για τα υπερμοντέλα 3. Αρχικοποίηση παραμέτρων εκπαίδευσης από την Keras 4. Εξορισμού από την Keras για την αρχικοποίηση των βαρών kernel_initializer='glorot_uniform' και των πολώσεων bias_initializer='zeros'
	Ενεργοποίηση κρυφών επιπέδων	ReLU	Συνιστώμενη από τις πηγές
	Ενεργοποίηση εξόδου	Σιγμοειδής	Ορίζεται από το πρόβλημα και είναι συνιστώμενη από τις πηγές
Εφαρμογή	Συνάρτηση κόστους	Δυαδική συνάρτηση εντροπίας	Ορίζεται από το πρόβλημα και είναι συνιστώμενη από τις πηγές
	Βελτιστοποιητής	Adam	1. Συνιστώμενη από τις πηγές

Φάση	Υπερπαράμετρος ή Ενέργεια	Επιλογή	Παρατηρήσεις
			2. Πειραματισμός τιμής learning rate για απλά μοντέλα 3. Επιλογή learning rate από την Keras για τα υπερμοντέλα
	Μετρική απόδοσης	Accuracy	Συνιστώμενη από τις πηγές
Εκπαίδευση - Επικύρωση	Εξομάλυνση	Πρόωρο σταμάτημα	1. Συνιστώμενη από τις πηγές 2. Πειραματισμός για απλά μοντέλα στο batch size 3. batch size εξορισμού από την Keras (=32) για τα υπερμοντέλα
	Επιπλέον εξομάλυνση για MLP με δύο κρυφά επίπεδα	Ναι	1. Dropout 0.1 σε κάθε επίπεδο για απλά μοντέλα και 0.2 για υπερμοντέλα 2. Batch Normalization σε κάθε επίπεδο 3. L2 Regularization 0.0001 για απλά μοντέλα σε κάθε επίπεδο 4. L2 Regularization εξορισμού από την Keras (0.01) στο 1ο επίπεδο
	Επικύρωση	Ναι	1. Συνιστώμενη από τις πηγές 2. Validation_split= 0.1 (το 10% του train set) με ίδιο batch_size 3. Εξορισμού από την Keras shuffle=True (ανακατάταξη των δεδομένων εκπαίδευσης πριν κάθε εποχή)
	Παρακολούθηση εκπαίδευσης	Ναι	1. Συνιστώμενη από τις πηγές 2. Θα εξάγεται γραφική παράσταση καμπυλών εκπαίδευσης, όπου θα

Φάση	Υπερπαράμετρος ή Ενέργεια	Επιλογή	Παρατηρήσεις
			φαίνονται οι συναρτήσεις κόστους (loss) και η ακρίβεια (accuracy) για train και validation
Αποτίμηση	Αποτέλεσμα συνάρτησης κόστους και ακρίβειας σταα test sets	Ναι	1. Ορίζεται από το πρόβλημα και είναι συνιστώμενη από τις πηγές. 2. Εμφάνιση τιμών loss και accuracy
Προγνώσεις	Μετρικές απόδοσης συνόλου δοκιμής	Πίνακας ταξινόμησης Precision Recall F1 score AUC	1. Ορίζεται από το πρόβλημα και είναι συνιστώμενη από τις πηγές. 2. Θα εξάγεται: • Σχεδίαση πίνακα ταξινόμησης • Αναφορά ταξινόμησης • Σχεδίαση καμπύλης ROC AUC

Πίνακας 6-2. Επιλογές ανάπτυξης μοντέλων

Τέλος, αποφασίσαμε για κάθε μοντέλο που ικανοποιεί τις προσδοκίες μας, να το αποθηκεύουμε μετά την φάση της εκπαίδευσης – επικύρωσης, ώστε να είναι διαθέσιμο για μελλοντικούς πειραματισμούς.

Αφού ολοκληρώσαμε τις απαραίτητες επιλογές μας, είμαστε πλέον έτοιμοι να προχωρήσουμε στην ανάπτυξη μοντέλων, αφού πρώτα παρουσιάσουμε το γενικό πλάνο της ανάπτυξης για κάθε μοντέλο.

6.2.1 Γενικό Πλάνο Ανάπτυξης Μοντέλου MLP με την Keras

Το API της Keras παρέχει όλες τις απαραίτητες συναρτήσεις για την ανάπτυξη του μοντέλου. Στον Πίνακα 6-3 παρουσιάζονται – με τη μορφή παραδείγματος- τα βασικά βήματα και οι αντίστοιχες εντολές την ανάπτυξη ενός μοντέλου, με δύο κρυφά επίπεδα με 16 και 8 νευρώνες αντίστοιχα,

σύμφωνα με τις αντίστοιχες επιλογές του Πίνακα 6-2 για τις υπερπαραμέτρους ενός μοντέλου χωρίς υπερρύθμιση από την Keras Tuner.

Εργασία	Ανάπτυξη μοντέλου (Keras)

Σχεδίαση	`# Σχεδίαση απλού μοντέλου χωρίς εξομάλυνση`
	`# Χρήση του αντικειμένου model`
	`# Ορισμός μοντέλου ως γραμμικός σωρός επιπέδων Sequential()`
	`model = Sequential()`
	`# Προσθήκες επιπέδων με τη συνάρτηση add`
	`# πρώτο κρυφό πλήρως συνδεμένο επίπεδο με συνάρτηση Dense`
	`# ορίσματα ο αριθμός των νευρώνων στο επίπεδο`
	`# οι μονάδες επιπέδου εισόδου (input_shape)`
	`# η συνάρτηση ενεργοποίησης (activation)`
	`model.add(Dense(16,input_shape=(30,),activation='relu'))`
	`# Ορισμός δεύτερου κρυφού επιπέδου - συνάρτησης ενεργοποίησης`
	`model.add(Dense(8, activation='relu'))`
	`# Ορισμός επιπέδου εξόδου και συνάρτησης ενεργοποίησης`
	`model.add(Dense(1, activation='sigmoid'))`
	`# Εμφάνιση αρχιτεκτονικής μοντέλου`
	`model.summary()`
	`# Σχεδίαση αρχιτεκτονικής μοντέλου (προαιρετικά - Εικόνα 6-2)`
	`from keras.utils.vis_utils import plot_model`
	`plot_model(model, to_file='model_plot.png', show_shapes=True, show_layer_names=True)`

	`# Σχεδίαση μοντέλου με εξομάλυνση Dropout 0.1`
	`model = Sequential()`
	`model.add(Dense(16,input_shape=(30,),activation='relu'))`
	`model.add(Dropout(0.1))`
	`model.add(Dense(8, activation='relu'))`
	`model.add(Dropout(0.1))`
	`model.add(Dense(1, activation='sigmoid'))`

	`# Σχεδίαση μοντέλου με L2 εξομάλυνση`
	`model = Sequential()`
	`model.add(Dense(16,input_shape=(30,),activation='relu',`
	`kernel_regularizer=regularizers.l2(0.0001)))`
	`model.add(Dense(8, activation='relu',`
	`kernel_regularizer=regularizers.l2(0.0001)))`
	`model.add(Dense(1, activation='sigmoid'))`

	`# Σχεδίαση μοντέλου με Batch Normalization εξομάλυνση`
	`model = Sequential()`
	`model.add(Dense(16,input_shape=(30,), activation='relu'))`
	`model.add(BatchNormalization())`
	`model.add(Dense(8, activation='relu'))`
	`model.add(BatchNormalization())`

Εργασία	Ανάπτυξη μοντέλου (Keras)
	`model.add(Dense(1, activation='sigmoid'))`

Εφαρμογή	`# Προετοιμασία του μοντέλου για εκπαίδευση με τη συνάρτηση`
	`# compile και ορίσματα`
	`# τη συνάρτηση κόστους (loss)`
	`# τον βελτιστοποιητή (optimizer)`
	`# τον επιθυμητό ρυθμό μάθησης (lr)`
	`# και την μετρική απόδοσης εκπαίδευσης (metrics)`
	`model.compile(loss ='binary_crossentropy',`
	`optimizer = Adam(lr=0.0001),`
	`metrics = ['accuracy'])`

Εξομάλυνση	`# Δημιουργία αντικειμένου για το πρόωρο σταμάτημα με τη`
	`# συνάρτηση EarlyStopping και τις επιθυμητές παραμέτρους`
	`earlystopper = EarlyStopping(monitor='val_loss', patience=5)`

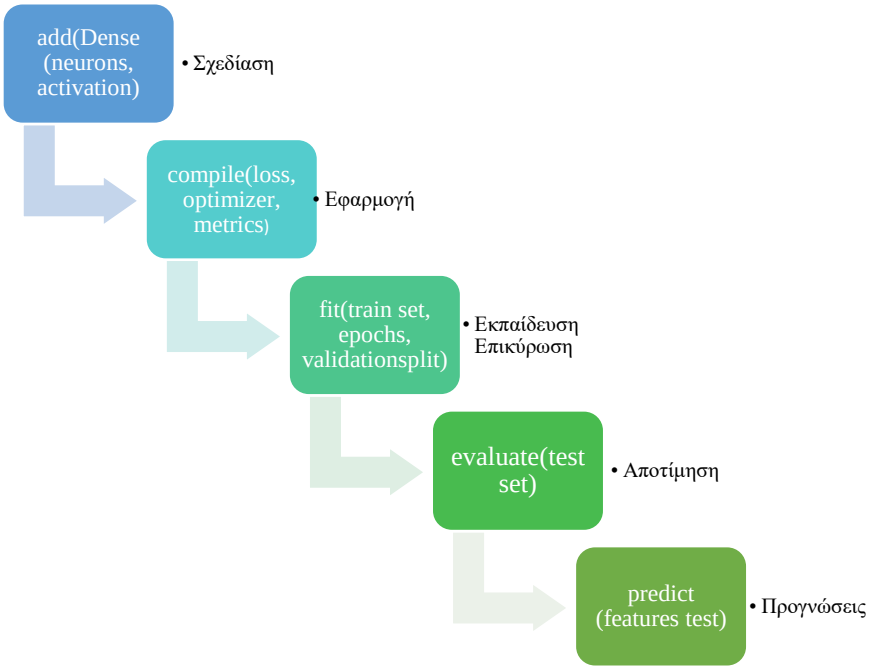
Εκπαίδευση - Επικύρωση	`# Εκπαίδευση του μοντέλου με τη συνάρτηση fit με ορίσματα`
	`# τα train sets`
	`# τις epochs`
	`# το batch_size`
	`# τον τρόπο επικύρωσης (validation_split)`
	`# την επιλογή εάν θα εμφανίζονται τα αποτελέσματα εκτέλεσης`
	`# (verbose)`
	`# τυχόν callbacks (callbacks)`
	`# καταγραφή του ιστορικού για κάθε εποχή και batch (history)`
	`history= model.fit(X_train, y_train,`
	`epochs = 2000,`
	`batch_size = 8`
	`validation_split = 0.1,`
	`verbose = 0,`
	`callbacks = [earlystopper])`

Αποτίμηση	`# Αποτίμηση του μοντέλου στα train sets με τη συνάρτηση`
	`# evaluate και ορίσματα τα σύνολα δοκιμής`
	`model.evaluate(X_test, y_test)`

Προγνώσεις	`# Δημιουργία προγνώσεων με τη συνάρτηση predict`
	`# όρισμα το σύνολο δοκιμής παραδειγμάτων`
	`model.predict(X_test)`

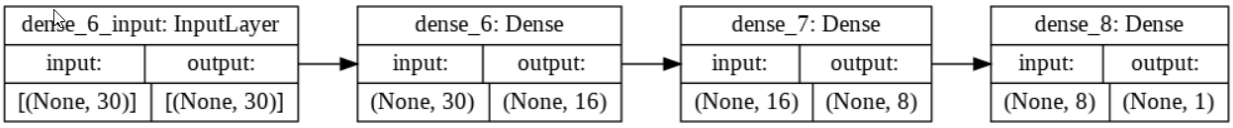
Πίνακας 6-3. Η ανάπτυξη του μοντέλου με την Keras

Στην Εικόνα 6-1 αντιστοιχίζουμε τις εντολές τις keras με τη ροή εργασιών ανάπτυξης του μοντέλου που δώσαμε στην παράγραφο 5.2.



Εικόνα 6-1. Η ανάπτυξη μοντέλου με την Keras

Στην Εικόνα 6-2 φαίνεται η απλή σχεδίαση (plot) αυτού του μοντέλου με συνάρτηση της Keras, έτσι όπως εξάγεται από το Colab[32].



Εικόνα 6-2. Ενδεικτική γραφική απεικόνιση μοντέλου από την Keras στο Colab

Τέλος, σε κάθε περίπτωση οι μετρικές απόδοσης που επιλέξαμε για την αποτίμηση του μοντέλου υπολογίζονται με τις εντολές που παρουσιάζονται στον Πίνακα 6-4.

Εργασία	**Μετρικές απόδοσης μοντέλου (Keras, scikit-learn, NumPy)**

Καμπύλη ROC-AUC	`from sklearn.metrics import roc_curve`
	`from sklearn.metrics import auc`
	`y_test_pred = model.predict(X_test)`
	`fpr_keras, tpr_keras, thresholds_keras =`

[32] Η τιμή None που εμφανίζεται στο διάγραμμα παραπάνω οφείλεται σε αδυναμία του λειτουργικού συστήματος (Ubuntu) πάνω από το οποίο τρέχει το Colab.

Εργασία	Μετρικές απόδοσης μοντέλου (Keras, scikit-learn, NumPy)
	`roc_curve(y_test,y_test_pred)` `auc_keras = auc(fpr_keras, tpr_keras)`

Πίνακας ταξινόμησης	`import numpy as np`
	`from sklearn.metrics import confusion_matrix`
	`y_pred = np.round(y_test_pred).flatten()`
	`cm = confusion_matrix(y_test, y_pred)`

Αναφορά ταξινόμησης	`from sklearn.metrics import classification_report` `cr = classification_report(y_test, y_pred)`

Πίνακας 6-4. Οι συναρτήσεις για την αποτίμηση ενός μοντέλου

Εφόσον κατά την εργασία της εκπαίδευσης - επικύρωσης είμαστε ικανοποιημένοι από το μοντέλο μας, μπορούμε να το αποθηκεύσουμε για μελλοντική χρήση σε νέα δεδομένα εκπαίδευσης. Στην περίπτωσή μας (όπου πειραματιζόμαστε με ένα συγκεκριμένο dataset), τα νέα δεδομένα εκπαίδευσης αφορούν τα test sets του αρχικού dataset. Η αποθήκευση του μοντέλου από την Keras γίνεται σε μορφή αρχείου .h5[33] με τη χρήση της συνάρτησης save. Για παράδειγμα, με βάση τον Πίνακα 6.3 μπορούμε να σώσουμε το απλό μοντέλο χωρίς επιπλέον εξομάλυνση:

```
model.save('hidden_16_8_8.h5')
```

και στην συνέχεια να το κατεβάσουμε από το Colab στον υπολογιστή μας:

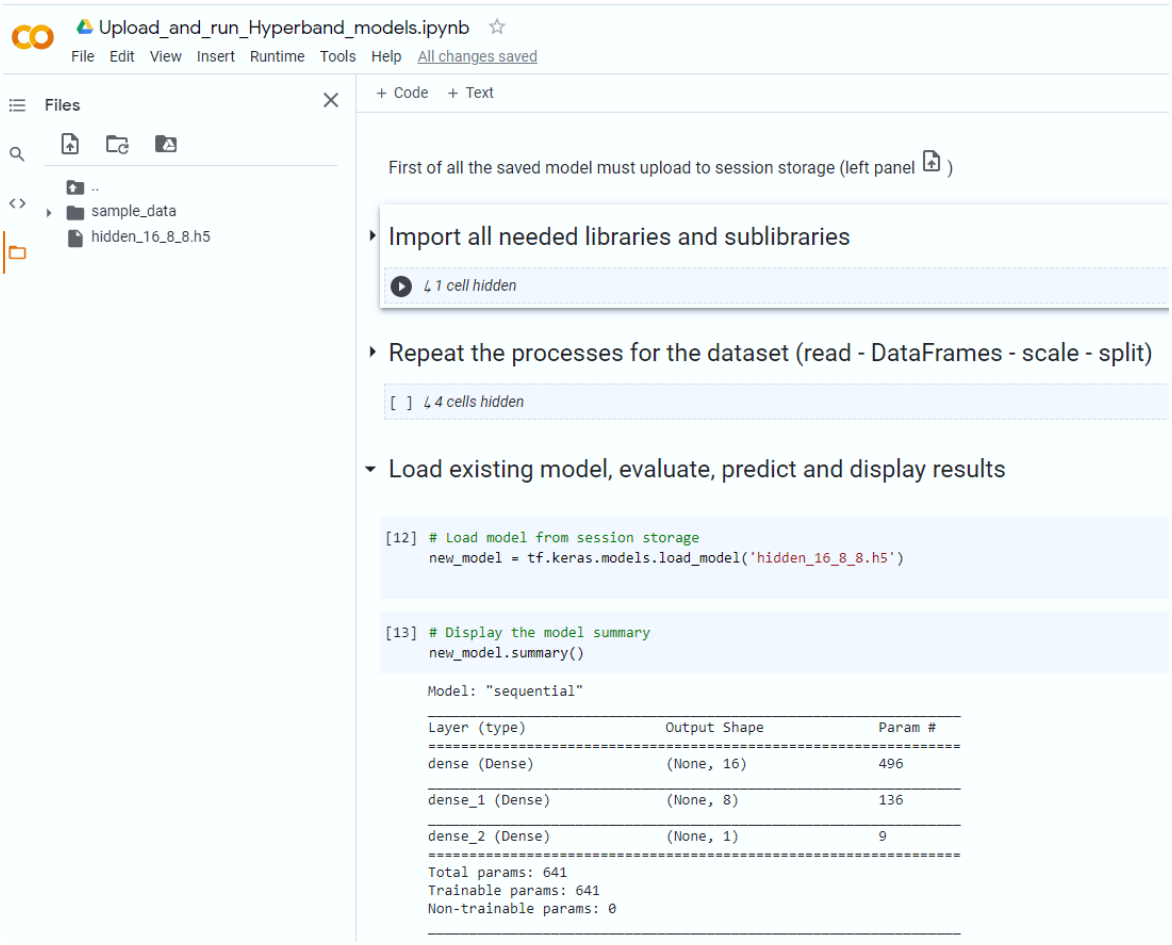
```
from google.colab import files
files.download('hidden_16_8_8.h5')
```

Στο αρχείο αποθηκεύονται οι παράμετροι μετά την ολοκλήρωση της φάσης της εκπαίδευσης-επικύρωσης. Όταν θα θελήσουμε να χρησιμοποιήσουμε ξανά το μοντέλο μας, αρκεί να ανεβάσουμε στο Colab το .h5 αρχείο, να προετοιμάσουμε τα δεδομένα και να εκτελέσουμε την αποτίμηση και την πρόγνωση για τα test sets, χωρίς να επαναλάβουμε τις φάσεις ανάπτυξης μέχρι και την εκπαίδευση – επικύρωση. Επίσης, να σημειώσουμε πληροφοριακά ότι, η αποθήκευση του μοντέλου μπορεί να γίνει και με callbacks.

Ένα παράδειγμα για το πώς μπορούμε να χρησιμοποιήσουμε ένα μοντέλο που αποθηκεύσαμε και στη συνέχει να πάρουμε τις προγνώσεις μας, δίνεται στο αρχείο

[33] https://www.h5py.org/

Upload_and_run_Hyperband_models.ipynb. Στην Εικόνα 6-3 δίνεται ένα στιγμιότυπο ανάκτησης αποθηκευμένου μοντέλου στο Colab.



Εικόνα 6-3. Στιγμιότυπο ανάκτησης αποθηκευμένου μοντέλου στο Colab

6.3 Αποτελέσματα Υλοποίησης Μοντέλων χωρίς Υπερρύθμιση

Σε πρώτη φάση αποφασίσαμε να πειραματιστούμε θέτοντας κάποιες υπερπαραμέτρους, χωρίς ρύθμιση από την Keras. Έτσι, κάναμε διάφορες δοκιμές για μοντέλα με δύο κρυφά επίπεδα, θέτοντας διάφορους αριθμούς νευρώνων, διάφορα learning rate, καθώς και batch_size. Αποφασίσαμε να παρουσιάσουμε μόνο ένα μοντέλο δύο κρυφών επιπέδων, με διαδικασίες όπως ακριβώς αναφέρονται στον Πίνακα 6-4. Πιο συγκεκριμένα, επιλέξαμε να παρουσιάσουμε τα αποτελέσματα των πειραματισμών με δύο κρυφά επίπεδα, με 16 νευρώνες στο πρώτο και 8 στο δεύτερο. Σε αυτό το μοντέλο αναφερόμαστε με τη σύμβαση 2- (16-8) και αυτή τη σύμβαση θα ακολουθήσουμε γενικότερα για την περιγραφή της αρχιτεκτονικής ενός μοντέλου. Το learning

rate για το μοντέλο αυτό είναι 0.0001 και το batch_size= 8. Υλοποιήσαμε άλλα τρία μοντέλα με επιπλέον εξομάλυνση και στα δύο επίπεδα: (i) Dropout= 0.1, (ii) L2-regularization=0.0001 και (iii) Batch normalization. Εκτός της παραπάνω μελέτης , αποφασίσαμε να πειραματιστούμε θέτοντας αποτελέσματα υλοποίησης μοντέλου χωρίς υπερρύθμιση με τεχνικές ανάλυσης (feature selection , feature extraction).

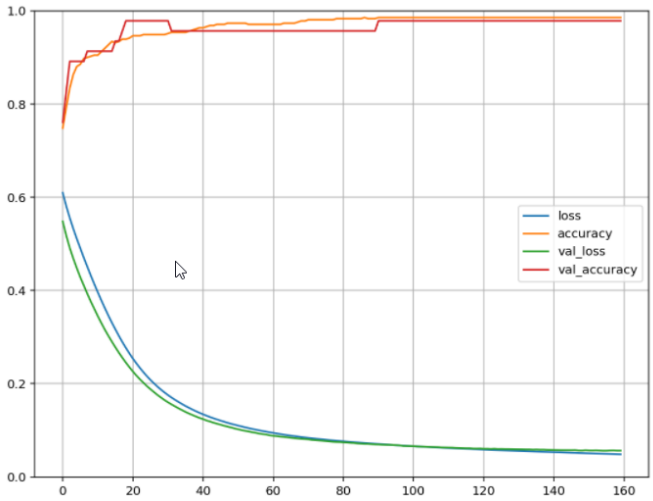
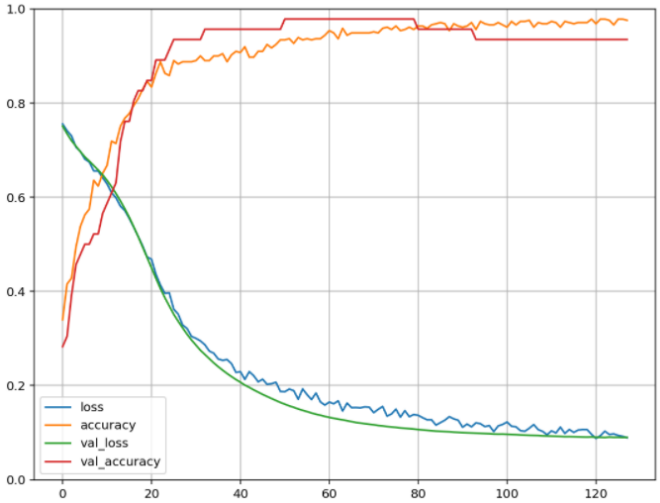
6.3.1 Αποτελέσματα MLP Δύο Κρυφών Επιπέδων (16-8)

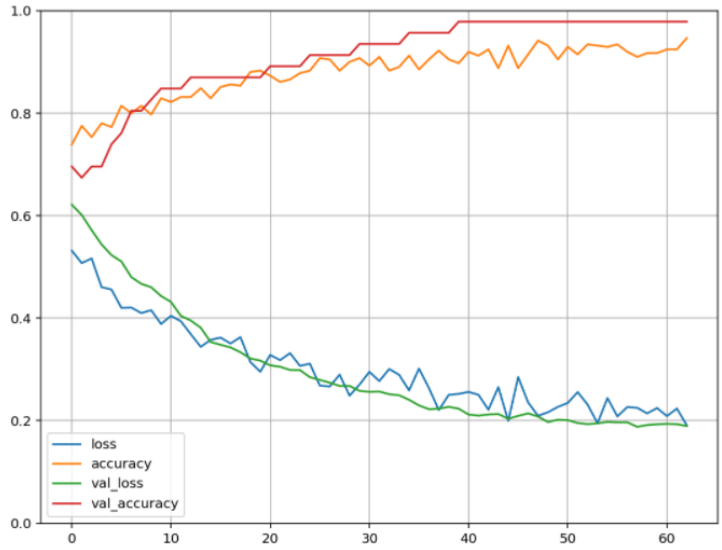
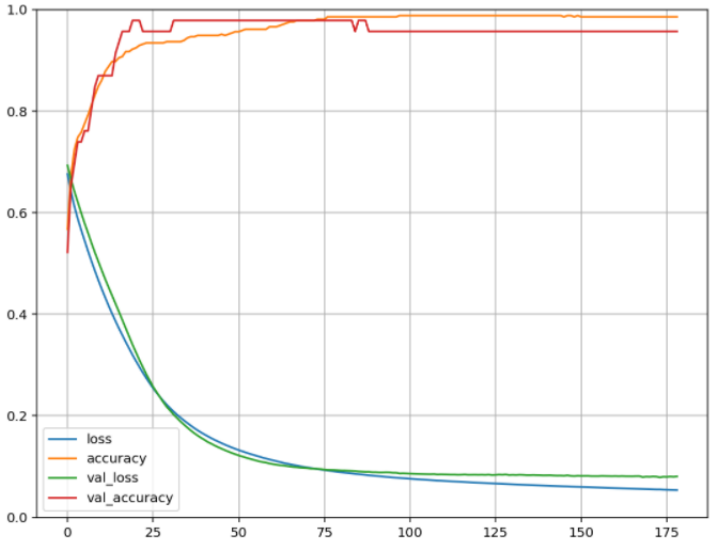
Στον Πίνακα 6-5 παραθέτουμε τα αποτελέσματα της ορθότητας και της συνάρτησης κόστους κατά τη φάση της εκπαίδευσης. Παρατηρούμε ότι, η εκπαίδευση για το απλό μοντέλο χωρίς επιπλέον εξομάλυνση πραγματοποιήθηκε σε σχεδόν διπλάσιες εποχές(epochs) από ότι στα μοντέλα με επιπλέον εξομάλυνση, όμως παρουσιάζει τις καλύτερες μετρικές.

Model	Train Accuracy	Train Loss	Epochs
Simple	0.9853	0.0484	160
Dropout 0.1	0.9779	0.0876	128
Batch Normalization	0.9462	0.1906	63
L2 Regularization	0.9877	0.0536	179

Πίνακας 6-5. Ορθότητα και συνάρτηση κόστους εκπαίδευσης MLP μοντέλων 2 επιπέδων- (16,8)

Στον Πίνακα 6-6 παραθέτουμε τις καμπύλες εκπαίδευσης κατά τη φάση της επικύρωσης-εκπαίδευσης για τα διάφορα 2-(16-8) μοντέλα, ανάλογα με την πρόσθετη μέθοδο εξομάλυνσης. Από τις καμπύλες αυτές μπορούμε να δούμε την εξέλιξη της εκπαίδευσης και της επικύρωσης. Παρατηρούμε την «κακή» συμπεριφορά της εξομάλυνσης batch normalization για ένα μοντέλο με μικρή χωρητικότητα, η οποία δεν συνίσταται για μοντέλα με μικρή χωρητικότητα και συνιστά λύση για μοντέλα με μεγαλύτερη χωρητικότητα.

Model	Καμπύλες εκπαίδευσης
Simple	
Dropout	

Model	Καμπύλες εκπαίδευσης
Batch Norm	
L2 Reg	

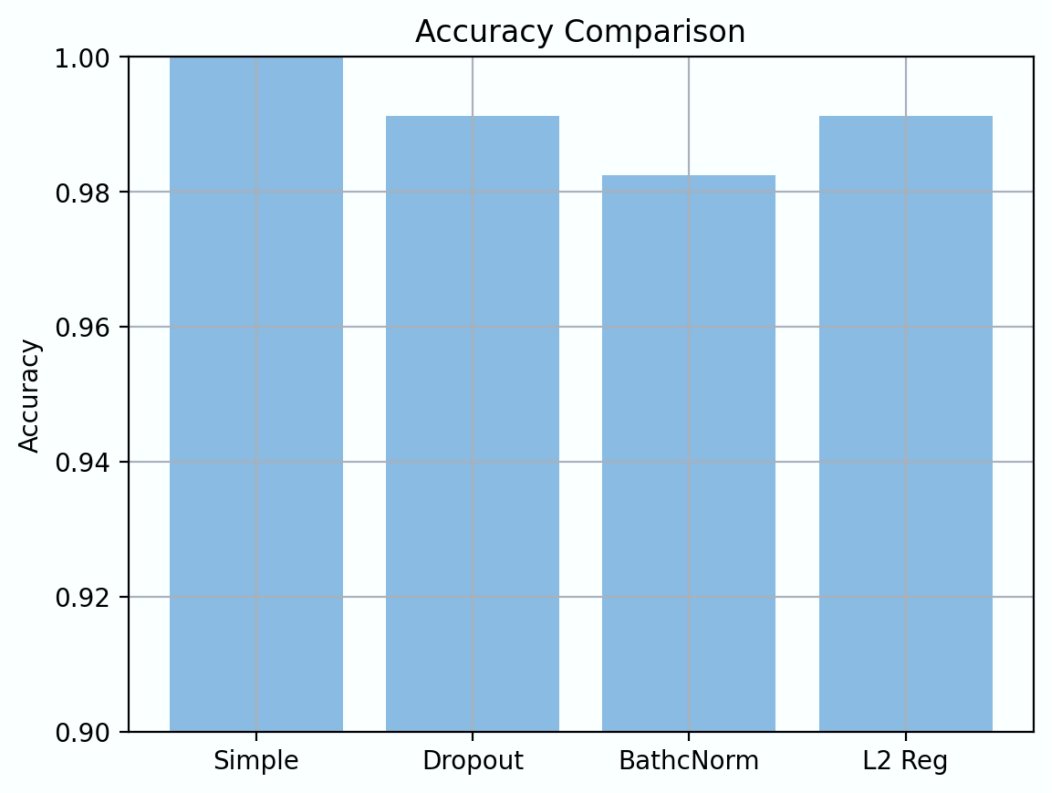
Πίνακας 6-6. Καμπύλες εκπαίδευσης MLP μοντέλων 2 επιπέδων- (16,8)

Όπως ήδη έχουμε αναφέρει, το πόσο ποιοτικό είναι ένα μοντέλο, κρίνεται από την αποτίμηση σε νέα δεδομένα. Συνεπώς, για τα τέσσερα μοντέλα παίρνουμε τις τιμές της ορθότητας και της συνάρτησης κόστους από την αποτίμηση στο σύνολο δοκιμής. Τα αποτελέσματα δίνονται στον Πίνακα 6-7, όπου φαίνεται ότι τις καλύτερες επιδόσεις είχε το απλό μοντέλο με μοναδική εξομάλυνση το πρόωρο σταμάτημα και τις χειρότερες το μοντέλο με Batch Normalization. Άλλωστε, αυτού του είδους η εξομάλυνση έχει νόημα σε MPL μεγαλύτερης χωρητικότητας.

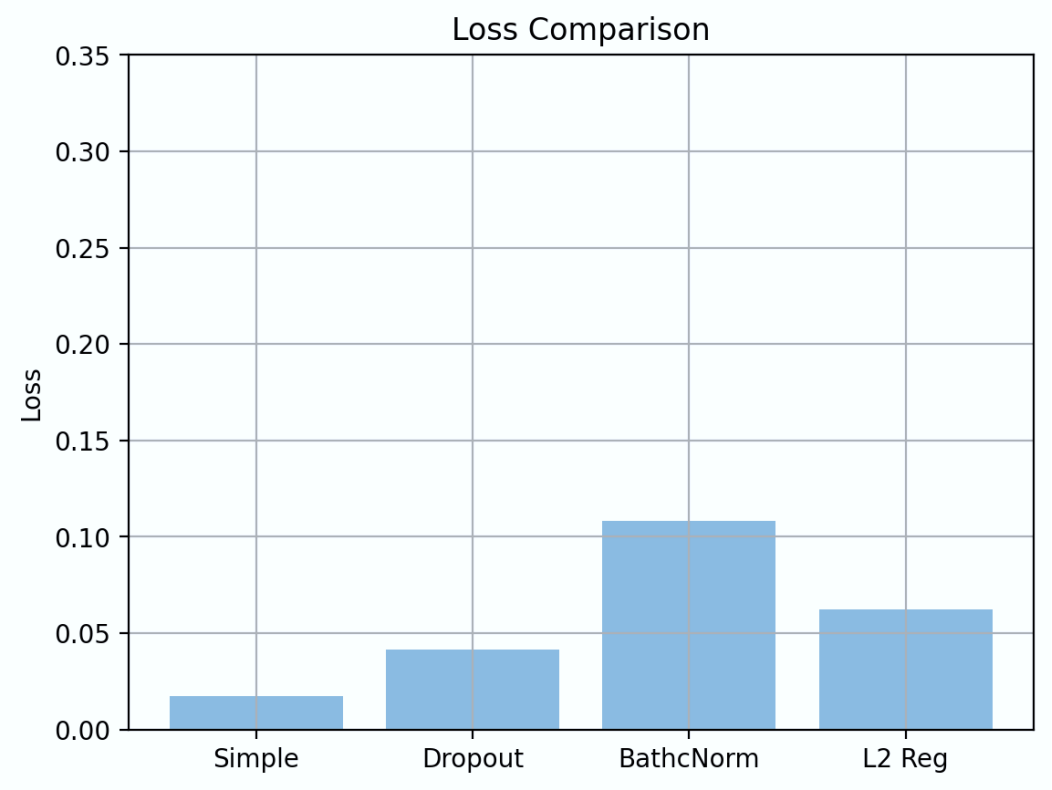
Μοντέλο 2- (16-8)	Αποτελέσματα αποτίμησης συνόλου δοκιμής
Simple	`Test loss: 0.0288 -  Test accuracy: 0.9912`
Dropout	`Test loss: 0.0510 - Test accuracy: 0.9824`
Batch Normalization	`Test loss: 0.1623 - Test accuracy: 0.9649`
L2 Regularization	`Test loss: 0.0505 - Test accuracy: 0.9736`

Πίνακας 6-7. Αποτελέσματα αποτίμησης MLP μοντέλων 2 επιπέδων- (16,8)

Η σύγκριση των μοντέλων όσον αφορά την ακρίβεια και τη συνάρτηση κόστους, φαίνεται γραφικά στις Εικόνες 6-4 και 6.5 αντίστοιχα.



Εικόνα 6-4. Σύγκριση ορθότητας μοντέλων 2-(16,8)



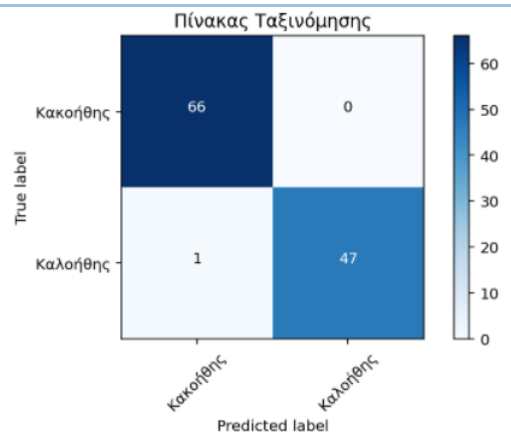
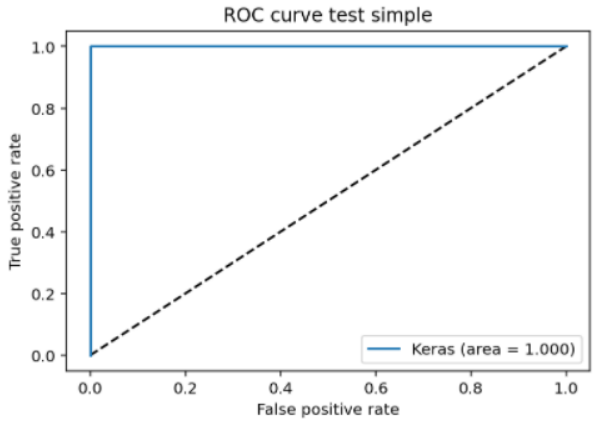
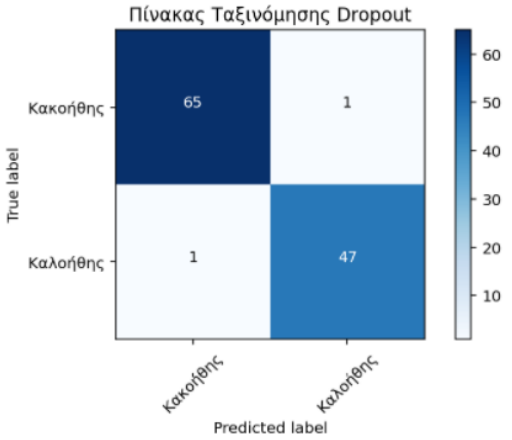
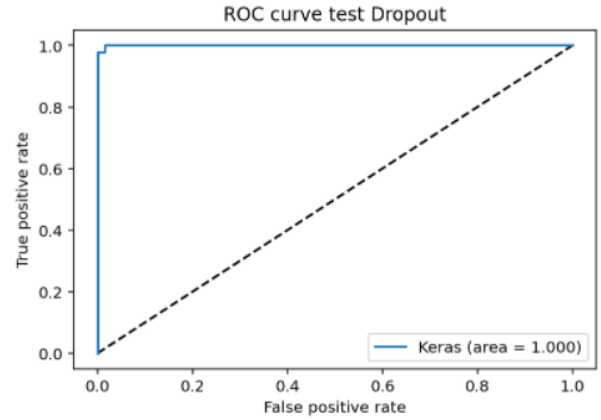
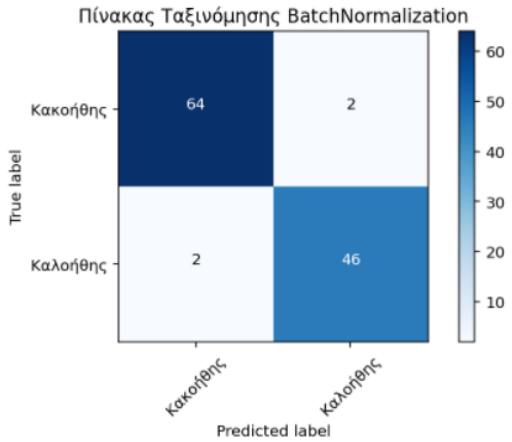
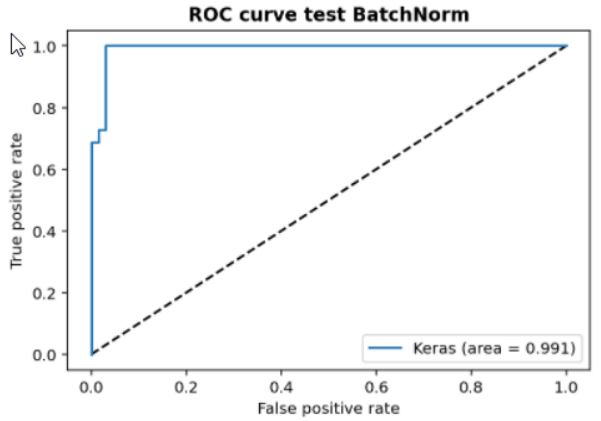
Εικόνα 6-5. Σύγκριση συνάρτησης κόστους μοντέλων 2-(16,8)

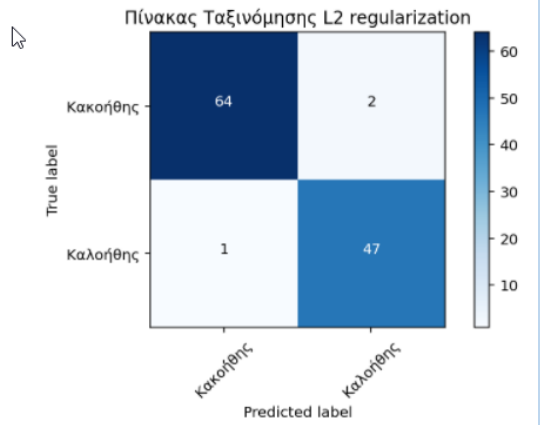
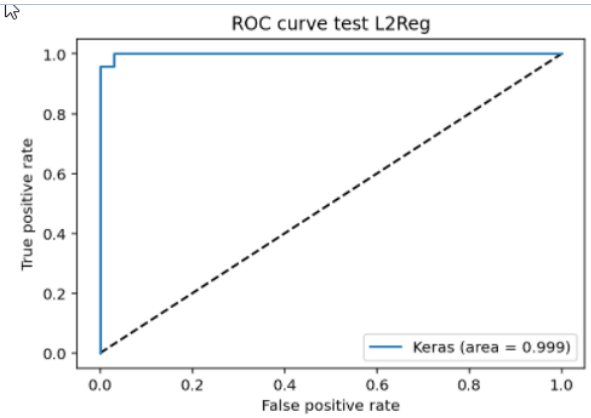
Διακρίνουμε μια σαφή υπεροχή του απλού μοντέλου μόνο με πρόωρο σταμάτημα, το οποίο μας έδωσε στην αποτίμηση ορθότητα 99%.

Όμως, για την αξιολόγηση της ποιότητας του μοντέλου, δεν αρκεί μόνο η μετρική της ορθότητας. Αφού ολοκληρώθηκε και η φάση της αποτίμησης, θα πρέπει να πάρουμε τις προγνώσεις από το σύνολο δοκιμής και να δούμε τις υπόλοιπες μετρικές. Με βάση τις προγνώσεις, υπολογίζουμε τον πίνακα ταξινόμησης για κάθε μοντέλο, την αναφορά της ταξινόμησης και τις καμπύλες ROC – AUC.

Στον Πίνακα 6-8 παραθέτουμε τα αποτελέσματα του πίνακα ταξινόμησης και τις καμπύλες ROC – AUC για κάθε μοντέλο. Διαπιστώνουμε ότι, το απλό μοντέλο απέδωσε τα μέγιστα, ενώ ικανοποιητικό αποτέλεσμα είχαν τα μοντέλα με dropout και L2 regularization, με το dropout να υπερτερεί ως προς την ROC – AUC καμπύλη.

Στον Πίνακα 6-9 παραθέτουμε την αναφορά της ταξινόμησης για κάθε μοντέλο. Διαπιστώνουμε ότι το απλό μοντέλο έχει άριστες μετρικές, ενώ εξίσου ικανοποιητικές ήταν οι αποδόσεις με dropout και L2 regularization.

Πίνακας Ταξινόμησης	**Καμπύλη ROC - AUC**
	
	
	

Πίνακας Ταξινόμησης	Καμπύλη ROC - AUC
	

Πίνακας 6-8. Πίνακες ταξινόμησης και καμπύλες ROC-AUC απλών μοντέλων 2-(16,8)

Μοντέλο 2-(16-8)	Αναφορά ταξινόμησης
Simple	precision recall f1-score support 0 0.99 1.00 0.99 66 1 1.00 0.98 0.99 48 accuracy 0.99 114 macro avg 0.99 0.99 0.99 114 weighted avg 0.99 0.99 0.99 114
Dropout	precision recall f1-score support 0 0.98 0.98 0.98 66 1 0.98 0.98 0.98 48 accuracy 0.98 114 macro avg 0.98 0.98 0.98 114 weighted avg 0.98 0.98 0.98 114
Batch Normalization	precision recall f1-score support 0 0.97 0.97 0.97 66 1 0.96 0.96 0.96 48 accuracy 0.96 114 macro avg 0.96 0.96 0.96 114 weighted avg 0.96 0.96 0.96 114

Μοντέλο 2-(16-8)	Αναφορά ταξινόμησης
L2 Regularization	precision recall f1-score support 0 0.98 0.97 0.98 66 1 0.96 0.98 0.97 48 accuracy 0.97 114 macro avg 0.97 0.97 0.97 114 weighted avg 0.97 0.97 0.97 114

Πίνακας 6-9. Αναφορές ταξινόμησης απλών μοντέλων 2-(16-8)

Βλέποντας τα αποτελέσματα συνολικά, για ένα μικρό σχετικά σύνολο δεδομένων, όπως το WDBC, συμπεραίνουμε ότι:

- *Οι αρχικές μας επιλογές για τις υπερπαραμέτρους, μετά βέβαια από πολλούς πειραματισμούς, μας οδήγησαν στο να επιτύχουμε να δημιουργήσουμε ένα προγνωστικό μοντέλο με άριστα αποτελέσματα.*
- *Με ένα MLP δύο κρυφών επιπέδων, με μικρό αριθμό νευρώνων ανά επίπεδο, με ρυθμό εκμάθησης και μικρό batch size ανά εποχή, χρησιμοποιώντας μόνο το πρόωρο σταμάτημα ως μέθοδο εξομάλυνσης, μπορούμε να πετύχουμε ένα άριστο μοντέλο.*

Σημειώνουμε όμως και τονίζουμε ξανά ότι, η ΜΜ είναι μια στατιστική διαδικασία και οι αλγόριθμοι στοχαστικοί. Συνεπώς, εκτελώντας ξανά την υλοποίηση στο Colab, θα πάρουμε διαφορετικά αποτελέσματα, γι' αυτό αποφασίσαμε να σώζουμε τα μοντέλα μας μετά τη φάση της εκπαίδευσης επικύρωσης, έτσι ώστε να κρατήσουμε τις παραμέτρους εκμάθησης. Επίσης, να σημειώσουμε ότι, πριν καταλήγουμε ποια ακριβώς εκτέλεση θα παρουσιάσουμε, τρέξαμε πολλές φορές το πρόγραμμα και σε κάθε εκτέλεση τα αποτελέσματα των προγνώσεων ήταν εξίσου ικανοποιητικά.

Αρχείο Colab

Breast_Cancer_Algorithms.ipynb

Αποθηκευμένα μοντέλα

Simple: hidden_16_8_8.h5

Με Dropout: hidden_16_8_8_drop.h5

Με Batch Normalization: hidden_16_8_8_batch.h5

Με L2 Regularizarion: hidden_16_8_8_L2.h5

6.3.2 Αποτελέσματα Υλοποίησης χωρίς Υπερρύθμιση με Τεχνικές Ανάλυσης (feature selection , feature extraction)

- Feature Selection

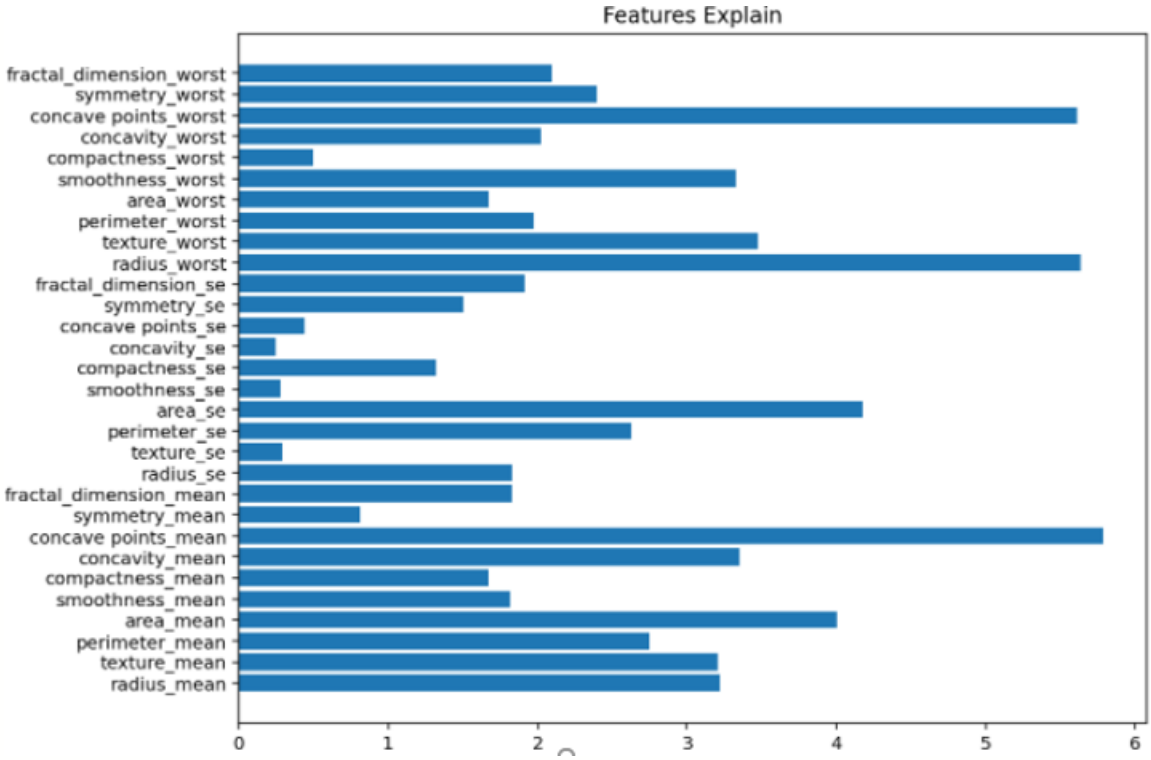
Η επιλογή χαρακτηριστικών είναι μία από τις δύο διαδικασίες μείωσης χαρακτηριστικών, η άλλη είναι η εξαγωγή χαρακτηριστικών. Η επιλογή χαρακτηριστικών είναι η διαδικασία με την οποία ένα υποσύνολο σχετικών χαρακτηριστικών ή μεταβλητών επιλέγεται από ένα μεγαλύτερο σύνολο δεδομένων για την κατασκευή μοντέλων. Επιλογή μεταβλητής, επιλογή χαρακτηριστικών ή επιλογή μεταβλητού υποσυνόλου είναι όλα τα άλλα ονόματα που χρησιμοποιούνται για την επιλογή χαρακτηριστικών. Το κύριο επίκεντρο της επιλογής χαρακτηριστικών είναι η επιλογή λειτουργιών που αντιπροσωπεύουν το σύνολο δεδομένων καλά, αποκλείοντας περιττά και άσχετα δεδομένα. Αυτό έρχεται σε αντίθεση με την εξαγωγή χαρακτηριστικών στην οποία δημιουργούνται νέες λειτουργίες ως συναρτήσεις των αρχικών χαρακτηριστικών. Το ίδιο είναι ότι η επιλογή χαρακτηριστικών και η εξαγωγή χαρακτηριστικών διασφαλίζουν ότι το μοντέλο μηχανικής μάθησης χρησιμοποιεί το πιο σχετικό και μη περιττό σύνολο δεδομένων. Ή επιλογή χαρακτηριστικών είναι χρήσιμη επειδή απλοποιεί τα μοντέλα εκμάθησης κάνοντας την ερμηνεία του μοντέλου και των αποτελεσμάτων ευκολότερη για τον χρήστη. Ένα άλλο πλεονέκτημα της επιλογής χαρακτηριστικών είναι η μείωση του χρόνου επεξεργασίας που μεταφράζεται σε μικρότερο χρόνο εκπαίδευσης για το μηχάνημα λόγω της χρήσης μόνο του σχετικού υποσυνόλου δεδομένων. Η ‘’κατάρα’’ της διαστατικότητας μπορεί επίσης να αποφευχθεί επειδή η επιλογή χαρακτηριστικών μπορεί να μειώσει τον αριθμό των διαστάσεων των δεδομένων. Η τελευταία λοιπόν αποτελεί ένα φαινόμενο όπου ένα σύνολο δεδομένων περιγράφεται σε τόσες πολλές διαστάσεις (ή από τόσα πολλά χαρακτηριστικά) που τα σημεία δεδομένων γίνονται αραιά και πλησιάζουν τη στατιστική ασήμαντη σημασία. Η επιλογή χαρακτηριστικών μειώνει τον αριθμό των διαστάσεων και μπορεί δυνητικά να κάνει τα δεδομένα αρκετά σημαντικά στατιστικά για να αποφευχθεί η ‘’κατάρα’’.

Εφαρμόζουμε Permutation Importance εγκαθιστώντας τις βιβλιοθήκες eli5,shap με σκοπό να παρατηρήσουμε ποια χαρακτηριστικά είναι τα πιο σημαντικά για να επιλεγούν για την εφαρμογή της τεχνικής feature selection. Παρακάτω παρατηρούμε με τα διαγράμματα ότι τα σημαντικά χαρακτηριστικά που συμβάλλουν για την επιλογή χαρακτηριστικών είναι τα :

- Concave_points_mean
- radius_worst
- Concave_points_worst

- area_se
- texture_mean
- perimeter_se

Weight	Feature
0.0123 ± 0.0044	area_se
0.0112 ± 0.0066	texture_mean
0.0107 ± 0.0049	concave points_mean
0.0096 ± 0.0029	perimeter_se
0.0093 ± 0.0052	radius_worst
0.0085 ± 0.0031	texture_worst
0.0078 ± 0.0038	smoothness_worst
0.0070 ± 0.0027	concave points_worst
0.0064 ± 0.0050	concavity_mean
0.0060 ± 0.0025	symmetry_worst
0.0057 ± 0.0040	fractal_dimension_worst
0.0051 ± 0.0017	fractal_dimension_se
0.0050 ± 0.0041	radius_se
0.0045 ± 0.0027	area_mean
0.0040 ± 0.0049	radius_mean
0.0039 ± 0.0024	smoothness_mean
0.0033 ± 0.0015	concavity_worst
0.0032 ± 0.0021	fractal_dimension_mean
0.0024 ± 0.0022	symmetry_mean
0.0022 ± 0.0027	area_worst
0.0022 ± 0.0014	compactness_se
0.0022 ± 0.0014	symmetry_se
0.0021 ± 0.0030	perimeter_mean
0.0020 ± 0.0015	perimeter_worst
0.0019 ± 0.0019	compactness_mean
0.0019 ± 0.0005	smoothness_se
0.0009 ± 0.0007	texture_se
0.0005 ± 0.0009	compactness_worst
0.0004 ± 0.0008	concave points_se
0.0004 ± 0.0003	concavity_se



Στο δικό μας σύνολο δεδομένων, θα εξετάσουμε την μέθοδο αυτή επιλέγοντας για n_components αυτά που είναι τα πιο σημαντικά και θα δούμε στο παρακάτω πίνακα τα αποτελέσματα αυτής. Τρέχοντας τον αλγόριθμο για την επιλογή κάθε φορά 4 attributes με τη σειρά όπως εμφανίζονται στο παραπάνω διάγραμμα έχουμε τον παρακάτω πίνακα με τα αποτελέσματα.

a) concave points_worst , radius_worst , concave points_mean , area_se

Test loss	Test accuracy	Train loss	Train accuracy
0.083010	0.964912	0.125046	0.948655

b) area_mean , texture_worst , smoothness_worst , concavity_mean

Test loss	Test accuracy	Train loss	Train accuracy
0.117576	0.960880	0.075927	0.973684

c) perimeter_se , perimeter_mean , texture_mean , radius_mean

Test loss	Test accuracy	Train loss	Train accuracy
0.268303	0.897310	0.219213	0.921052

Κάνοντας και άλλα πειράματα με n_components = 3 (concave points_worst , concave points_mean , radius_worst) θα παρατηρήσουμε ότι διαφέρουν τα αποτελέσματα όπως φαίνεται και στο παρακάτω πίνακα

Test loss	Test accuracy	Train loss	Train accuracy
0.078596	0.973684	0.144490	0.936430

Παίρνοντας και components 2 και 1 προκύπτουν αποτελέσματα τα οποία τα ποσοστά τους είναι χαμηλότερα σε σχέση με τα παραπάνω.
Συμπεραίνουμε, ότι η τεχνική του Feature Selection λειτουργεί πιο αποτελεσματικά με 4 components από όλο το σύνολο δεδομένων μας.

- Feature Extraction

Εισάγουμε τεχνικές για μείωση διαστατικότητας για την ανάλυση πολλών παραλλαγών σε δεδομένα. Ειδικότερα, θα εξηγήσουμε πως να χρησιμοποιήσουμε την τεχνική της Γραμμικής Διακριτικής Ανάλυσης (LDA) για να μειώσουμε τη διάσταση του χώρου των μεταβλητών και να τη συγκρίνουμε με την τεχνική της ανάλυσης βασικών στοιχείων(PCA) , έτσι ώστε να μπορούμε να έχουμε κάποια κριτήρια για τα οποία πρέπει να χρησιμοποιηθούν σε δεδομένη υπόθεση. Τόσο η LDA όσο και η PCA είναι τεχνικές γραμμικού μετασχηματισμού που χρησιμοποιούνται συνήθως για την μείωση της διάστασης. Η πιο σημαντική διαφορά μεταξύ τους είναι ότι η PCA μπορεί να περιγραφεί ως «μη επιτηρούμενος αλγόριθμος» , καθώς «αγνοεί» τις ετικέτες τάξης και ο στόχος της είναι να βρει τα κύρια στοιχεία που μεγιστοποιούν τη διακύμανση σε ένα σύνολο δεδομένων , ενώ το LDA είναι ένας «εποπτευόμενος αλγόριθμος» που υπολογίζει τις κατευθύνσεις («γραμμικές διακρίσεις») που αντιπροσωπεύουν τους άξονες που μεγιστοποιούν τον διαχωρισμό μεταξύ πολλαπλών κατηγοριών. Διαισθητικά , θα μπορούσαμε να πιστεύομε ότι το LDA είναι ανώτερο από το PCA για μια εργασία ταξινόμησης πολλαπλών κατηγοριών όπου οι ετικέτες κλάσης είναι γνωστές. Ωστόσο, αυτό δεν ισχύει πάντα. Για παράδειγμα, οι συγκρίσεις μεταξύ της ακρίβειας ταξινόμησης για την αναγνώριση εικόνας μετά τη χρήση PCA ή LDA δείχνουν ότι το PCA τείνει να ξεπεράσει το LDA εάν ο αριθμός των δειγμάτων ανά τάξη είναι σχετικά μικρός. Στη πράξη, δεν είναι ασυνήθιστο να χρησιμοποιείται τόσο LDA όσο και PCA σε συνδυασμό : πχ PCA για μείωση διαστάσεων ακολουθούμενο από LDA. Με λίγα λόγια, μπορούμε να πούμε ότι το PCA είναι ένας μη εποπτευόμενος αλγόριθμος που προσπαθεί να βρει τους ορθογώνιους άξονες συνιστωσών μέγιστης διακύμανσης σε ένα σύνολο δεδομένων, ενώ ο στόχος του LDA ως εποπτευόμενου αλγορίθμου είναι να βρει το δευτερεύον χώρο λειτουργιών που

βελτιστοποιεί τη διαχωριστικότητα κλάσης. Παρακάτω παρουσιάζουμε τις δυο τεχνικές εξαγωγής χαρακτηριστικών σύμφωνα με το σύνολο δεδομένων μας.

a) Με PCA

Σε πολλά σύνολα δεδομένων διαπιστώνουμε ότι ο αριθμός των χαρακτηριστικών είναι πολύ μεγάλος και αν θέλουμε να εκπαιδεύσουμε το μοντέλο , χρειάζεται περισσότερο υπολογιστικό κόστος. Για να μειώσουμε τον αριθμό των χαρακτηριστικών μπορούμε να χρησιμοποιήσουμε τη μέθοδο της ανάλυσης βασικών στοιχείων (PCA- Principal Components Analysis). Πιο συγκεκριμένα, είναι μια μέθοδος που χρησιμοποιείται για τη μείωση της διάστασης των μεγάλων συνόλων δεδομένων, μετατρέποντας ένα μεγάλο σύνολο μεταβλητών σε μικρότερο, το οποίο εξακολουθεί να περιέχει τις περισσότερες πληροφορίες στο μεγάλο σύνολο. Για να το επιτύχουμε αυτό παίρνουμε υπόψιν μας μια σειρά βημάτων. Η μείωση του αριθμού των μεταβλητών ενός συνόλου δεδομένων έρχεται φυσικά σε βάρος ακρίβειας αλλά το κόλπο στη μείωση των διαστάσεων είναι να ανταλλάξετε ακρίβεια με απλότητα. Τα μικρότερα σύνολα δεδομένων είναι ευκολότερα στην εξερεύνηση και την οπτικοποίηση και καθιστούν την ανάλυση δεδομένων πολύ πιο εύκολη και ταχύτερη για αλγορίθμους μηχανικής εκμάθησης χωρίς επεξεργασία εξωτερικών μεταβλητών. Συνοψίζοντας , η ιδέα του PCA είναι απλή – μείωση αριθμού μεταβλητών ενός συνόλου δεδομένων , διατηρώντας παράλληλα όσο το δυνατόν περισσότερες πληροφορίες. Πριν προχωρήσουμε στην υλοποίηση της ιδέας αυτής , θα πρέπει όλες οι μεταβλητές να έχουν μετατραπεί στην ίδια κλίμακα- scaling(Εξασφαλίζεται και ένα ποσοστό 10% σε ακρίβεια). Αφού πραγματοποιηθεί αυτό θα εξετάσουμε τον πίνακα συν διακύμανσης όπου εκεί θα προσδιοριστούν και οι συσχετίσεις μεταξύ των μεταβλητών μας. Στο δικό μας σύνολο δεδομένων θα εξετάσουμε την μέθοδο PCA με n= 1, 2, 3, 4 components και θα δούμε στο παρακάτω πίνακα τα αποτελέσματα αυτής.

	Epochs (early stopping)	Train loss	Train accuracy	Test loss	Test accuracy
N=1	128	0.0778	0.9731	0.0624	0.9824
N=2	269	0.1399	0.9462	0.0908	0.973

N=3	232	0.1194	0.9462	0.0834	0.9649
N=4	128	0.1087	0.9682	0.0765	0.9736

b) Με LDA

Η Ανάλυση Γραμμικής Διάκρισης (LDA) είναι η μέθοδος μετασχηματισμού δεδομένων που ανήκουν σε κατηγορίες (κλάσεις) με σκοπό τον καλύτερο διαχωρισμό των κλάσεων και ταυτόχρονα την ελάττωση της διαστατικότητας των δεδομένων. Για την επίτευξη της καλύτερης διαχωρισιμότητας, τα δεδομένα κατά τον μετασχηματισμό τους προβάλλονται σε χώρο μικρότερης διάστασης από τον αρχικό, με αποτέλεσμα να ελαττώνεται το πλήθος των διαστάσεων τους , έτσι ώστε στη συνέχεια η ταξινόμηση νέων δεδομένων να είναι ευχερέστερη και ακριβέστερη. Για να εφαρμοστεί η LDA τα δεδομένα πρέπει να είναι αριθμητικά με συνεχείς τιμές και να ανήκουν σε 2 ή περισσότερες κλάσεις(ισχύει ότι κλάσεις = 2 → συστατικά = κλάσεις - 1 → συστατικά = 1). Στο δικό μας σύνολο δεδομένων θα εξετάσουμε την μέθοδο LDA με n= 1 component(διότι ισχύει το παραπάνω) και θα δούμε στο παρακάτω πίνακα το αποτελέσματα αυτής.

	Epochs (early stopping)	Train loss	Train accuracy	Test loss	Test accuracy
N = None or 1	196	0.0767	0.9731	0.0594	0.9824

Το συμπέρασμα από την συγκεκριμένη μελέτη είναι ότι οι τεχνικές Feature Selection και Feature Extraction λειτουργούν σχεδόν το ίδιο για μικρό σχετικά σύνολο δεδομένων. Μειώνουν τη πολυπλοκότητα και τη διάσταση του συνόλου δεδομένων καθώς με λιγότερα attributes μπορούμε να εξάγουμε εξίσου αρκετά ικανοποιητικά αποτελέσματα.

Θέλοντας να πειραματιστούμε περισσότερο στις υλοποιήσεις των μοντέλων, αλλά χωρίς να επιλέξουμε εμείς την αρχιτεκτονική και τον ρυθμό μάθησης, αποφασίσαμε να εκμεταλλευτούμε τις δυνατότητες που μας δίνει η βιβλιοθήκη Keras tuner για τη ρύθμιση αυτών των υπερπαραμέτρων και παρουσιάζουμε τα αποτελέσματα στην επόμενη παράγραφο.

6.4 Αποτελέσματα Υλοποίησης με Ρύθμιση Υπερπαραμέτρων

Η βιβλιοθήκη Keras Tuner[34] προσφέρει διάφορες μεθόδους την ρύθμιση των υπερπαραμέτρων (ή υπερρύθμιση) των μοντέλων νευρωνικών δικτύων, όπως οι RandomSearch and Hyperband. Επιλέξαμε τον Hyperband tuner της Keras για την εύρεση δύο υπερπαραμέτρων: του αριθμού των νευρώνων ανά κρυφό επίπεδο και τον ρυθμό μάθησης. Για την υλοποίηση της Hyperband απαιτείται στο Colab η εγκατάσταση της βιβλιοθήκης Keras Tuner.

Όταν σχεδιάζουμε ένα μοντέλο με τη μέθοδο Hyperband, ορίζουμε σε ποιο διάστημα τιμών θα γίνει η αναζήτηση των βέλτιστων υπερπαραμέτρων, καθώς και την αρχιτεκτονική του μοντέλου και η εφαρμογή του. Το μοντέλο ονομάζεται **υπερμοντέλο (hypermodel)**. Για παράδειγμα, η συνάρτηση σχεδιασμού ενός MLP υπερμοντέλου για το πρόβλημα μας, με δύο κρυφά επίπεδα και ομαλοποίηση L2 στο πρώτο επίπεδο θα είναι:

```
def model_builder(hp):
  model = keras.Sequential()
  # Tune the number of units in the Dense layers
  # Choose an optimal value between 8-64 for units per layer
  hp_units = hp.Int('units', min_value = 8, max_value =64, step = 8)
  model.add(keras.layers.Dense(units = hp_units, input_shape=(30,),
  activation = 'relu', kernel_regularizer='l2'))
  model.add(keras.layers.Dense(units = hp_units,activation = 'relu'))
  model.add(keras.layers.Dense(1, activation='sigmoid'))
  # Tune the learning rate for the optimizer
  # Choose an optimal value from 0.01, 0.001, or 0.0001
  hp_learning_rate = hp.Choice('learning_rate', values =
  [1e-2, 1e-3, 1e   4])
  # Prepare the model for training
  model.compile(optimizer =
  keras.optimizers.Adam(learning_rate = hp_learning_rate),
                loss = keras.losses.binary_crossentropy,
```

[34] https://github.com/keras-team/keras-tuner

```
                metrics = ['accuracy'])
  return model
```

Για να δημιουργήσουμε ένα στιγμιότυπο του Hyperband tuner, πρέπει να ορίσουμε το υπερμοντέλο, τον στόχο της βελτιστοποίησης και τον μέγιστο αριθμό των εποχών (max_epochs). Για παράδειγμα, εάν ο στόχος είναι η validation accuracy, το στιγμιότυπο του tuner δημιουργείται ως εξής:

```
tuner = kt.Hyperband(model_builder,
                     objective = 'val_accuracy',
                     max_epochs = 100,
                     factor = 3)
```

Πριν τρέξουμε το ψάξιμο των υπερπαραμέτρων, πρέπει να ορίσουμε ένα callback για την εκκαθάριση των εξόδων (outputs) της εκπαίδευσης στο τέλος κάθε βήματος εκπαίδευσης:

```
class ClearTrainingOutput(tf.keras.callbacks.Callback):
  def on_train_end(*args, **kwargs):
    IPython.display.clear_output(wait = True)
```

Στην συνέχεια, τρέχουμε το ψάξιμο των υπερπαραμέτρων. Τα ορίσματα για τη μέθοδο αναζήτησης είναι τα ίδια με αυτά που χρησιμοποιούμε στη συνάρτηση fit της Keras, συν το παραπάνω callback:

```
tuner.search(X_train, y_train, epochs = 100, validation_split=0.1,
callbacks = [ClearTrainingOutput()])
```

Για να πάρουμε τις βέλτιστες υπερπαραμέτρους δίνουμε την εντολή:

```
# Get the optimal hyperparameters
best_hps = tuner.get_best_hyperparameters(num_trials = 1)[0]
print(f"""
The hyperparameter search is complete. The optimal number of units in the
first densely-connected
layer is {best_hps.get('units')} and the optimal learning rate for the opt
imizer
is {best_hps.get('learning_rate')}.
```

```
""")
```

Ένα αποτέλεσμα από την παραπάνω εκτέλεση μπορεί να είναι:

```
The hyperparameter search is complete. The optimal number of units in the
first densely-connected layer is 40 and the optimal learning rate for the
optimizer is 0.001.
```

Τέλος, δημιουργούμε ένα νέο μοντέλο με βάση το υπερμοντέλο:

```
# Build the model with the optimal hyperparameters and train it on  the d
ata
model = tuner.hypermodel.build(best_hps)
```

Εφόσον έχουμε πάρει τα αποτελέσματα για τις βέλτιστες παραμέτρους που αφορούν τον αριθμό των μονάδων ανά επίπεδο και το βέλτιστο ρυθμό μάθησης με κριτήριο την ορθότητα της επικύρωσης, ξεκινάμε πάλι με τα γνωστά βήματα την ανάπτυξη του υπερμοντέλου.

Στις επόμενες παραγράφους δίνουμε τα σχετικά αποτελέσματα για κάθε επιμέρους μοντέλο που δημιουργήσαμε. Γενικά:

- Εκτελούμε τα βήματα του Hyperband tuner, όπου ορίζουμε τη σχεδίαση και την εφαρμογή του μοντέλου ορίζοντας για κάθε μοντέλο την αντίστοιχη συνάρτηση `model_builder(hp)`
- Αφού πάρουμε τα αποτελέσματα από τον Hyperband για τον βέλτιστο αριθμό νευρώνων και τον ρυθμό μάθησης δημιουργούμε το υπερμοντέλο και ακολουθούμε τη διαδικασία ανάπτυξης με την ίδια ροή εργασιών ξεκινώντας από τη φάση εκπαίδευσης- αποτίμησης.

6.4.1 Υπερμοντέλα MLP Ενός Κρυφού Επιπέδου

Η πρώτη υλοποίηση αφορά υπερμοντέλο με ένα κρυφό επίπεδο. Όπως ήδη έχουμε αναφέρει, η επιλογή του βελτιστοποιητή Adam βασίστηκε κατά κύριο λόγο στις πηγές μας. Παρόλα αυτά, αποφασίσουμε να δοκιμάσουμε και την βελτιστοποίηση με άλλο βελτιστοποιητή, προκειμένου να διαπιστώσουμε γιατί στα MLP για προβλήματα παρόμοια με της δικής μας ΜΔΕ προτείνεται ως βελτιστοποιητής ο Adam. Έτσι, δοκιμάσαμε τον Hyperband tuner για τον βελτιστοποιητή SGD, που βασίζεται στην υλοποίηση του αλγόριθμου Stochastic Gradient Descent.
Παρακάτω, για κάθε υπερμοντέλο παραθέτουμε:

- Σύνοψη του κάθε μοντέλου
- Αποτελέσματα εκπαίδευσης και επικύρωσης

- Αποτελέσματα αποτίμηση
- Αποτελέσματα προγνώσεων
- Το αντίστοιχο notebook, καθώς και το αποθηκευμένο μοντέλο

6.4.1.1 Υπερμοντέλο με Adam βελτιστοποιητή

Η εφαρμογή του Hyperband tuner με τον βελτιστοποιητή Adam είχε ως αποτέλεσμα 16 νευρώνες στο κρυφό επίπεδο και ρυθμό εκμάθησης 0.01. Παραθέτουμε τα αποτελέσματα:

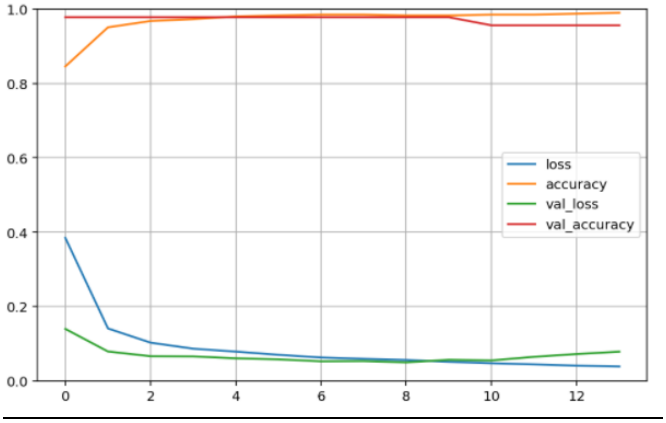
Σύνοψη

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 16)                496
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 17
=================================================================
Total params: 513
Trainable params: 513
Non-trainable params: 0
```

Εκπαίδευση – Επικύρωση

```
Epoch 14/2000
13/13 [==============================] - 0s 3ms/step - loss: 0.0383 - accuracy: 0.9902 - val_loss: 0.0780 - val_accuracy: 0.9565
Epoch 00014: early stopping
```

Καμπύλες εκπαίδευσης:



Εικόνα 6-6. Καμπύλες εκπαίδευσης υπερμοντέλου 1-(16)

Αποτίμηση

```
4/4 [==============================] - 0s 2ms/step - loss: 0.0221 - accuracy: 0.9912
Test loss:  0.02206709235906601
Test accuracy:  0.9912280440330505
```

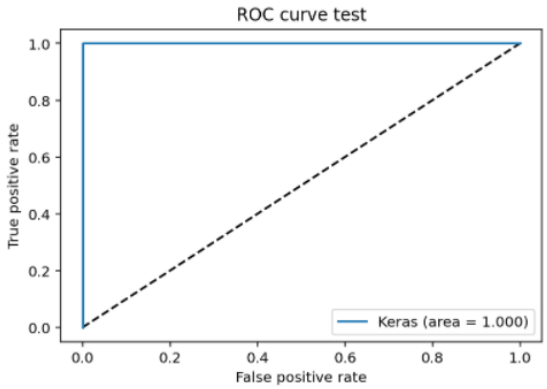
Μετρικές απόδοσης για τις προγνώσεις

α. Υπολογισμός AUC

Αποτέλεσμα AUC:

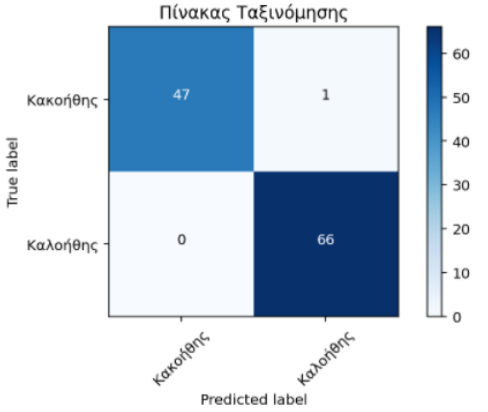
```
Testing data AUC:  1.0
```

Καμπύλη ROC AUC:



Εικόνα 6-7. Καμπύλη ROC-AUC υπερμοντέλου 1-(64)

β. Πίνακας ταξινόμησης



Εικόνα 6-8. Πίνακας ταξινόμησης υπερμοντέλου 1-(64)

γ. Αναφορά ταξινόμησης

	precision	recall	f1-score	support
0	1.00	0.98	0.99	48
1	0.99	1.00	0.99	66
accuracy			0.99	114
macro avg	0.99	0.99	0.99	114
weighted avg	0.99	0.99	0.99	114

Αποθήκευση και σώσιμο μοντέλου

Σχετικό notebook:

Hyperband_tuning_simple_16.ipynb

Αποθηκευμένο μοντέλο

Hyperband_simple_16.h5

6.4.1.2 Υπερμοντέλο με SGD βελτιστοποιητή

Η εφαρμογή του Hyperband tuner με τον βελτιστοποιητή υλοποιήσουμε ένα υπερμοντέλο με ένα κρυφό επίπεδο με είχε ως αποτέλεσμα 56 νευρώνες στο κρυφό επίπεδο και ρυθμό εκμάθησης 0.01. Παραθέτουμε τα αποτελέσματα:

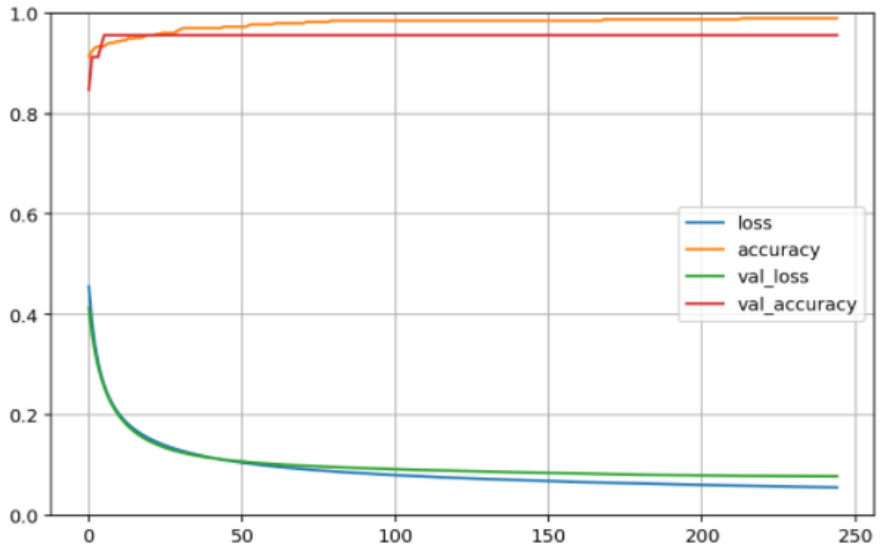
Σύνοψη μοντέλου

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 56)                1736
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 57
=================================================================
Total params: 1,793
Trainable params: 1,793
Non-trainable params: 0
_________________________________________________________________
```

Εκπαίδευση – Επικύρωση

```
Epoch 245/2000
13/13 [==============================] - 0s 4ms/step - loss: 0.0552 - accuracy: 0.9902 - val_loss: 0.0773 - val_accuracy: 0.9565
Epoch 00245: early stopping
```

Καμπύλες εκπαίδευσης:



Εικόνα 6-9. Καμπύλες εκπαίδευσης υπερμοντέλου 1-(56) με SGD

Αποτίμηση

```
4/4 [==============================] - 0s 3ms/step - loss: 0.0391 - accuracy: 0.9912
Test loss:  0.039122994989156 72
Test accuracy:  0.9912280440330505
```

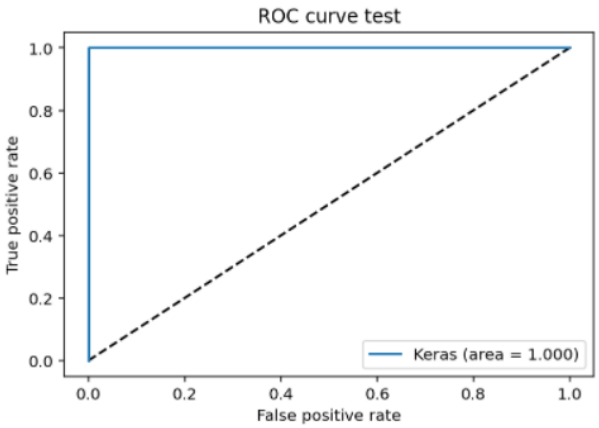
Μετρικές απόδοσης προγνώσεων

α. Υπολογισμός AUC

Αποτέλεσμα AUC:

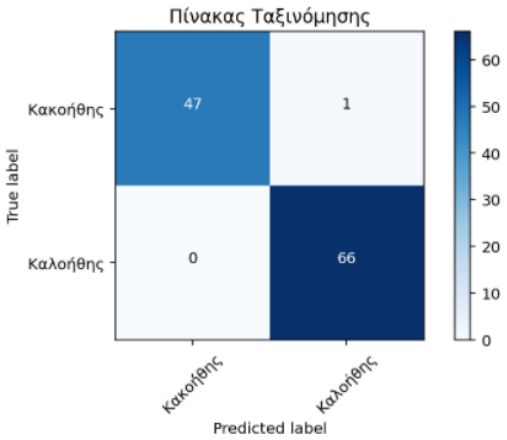
```
Testing data AUC:  1.0
```

Καμπύλη ROC AUC:



Εικόνα 6-10. Καμπύλη ROC AUC υπερμοντέλου 1-(56) με SGD

β. Πίνακας ταξινόμησης



Εικόνα 6-11. Πίνακας ταξινόμησης υπερμοντέλου 1-(56) με SGD

γ. Αναφορά ταξινόμησης

	precision	recall	f1-score	support
0	1.00	0.98	0.99	48
1	0.99	1.00	0.99	66
accuracy			0.99	114
macro avg	0.99	0.99	0.99	114
weighted avg	0.99	0.99	0.99	114

Αποθήκευση και σώσιμο μοντέλου

Σχετικό notebook:

Hyperband_tuning_simple_SGD_56.ipynb

Αποθηκευμένο μοντέλο

Hyperband_simple_SGD_56.h5

Από ότι διαπιστώνουμε, τα δύο μοντέλα αποδίδουν εξίσου καλά όσον αφορά τις μετρικές απόδοσης. Ποιες ήταν όμως οι διαφορές ανάμεσα στα δύο μοντέλα:

- Ο αριθμός των νευρώνων στο κρυφό επίπεδο: 16 με τον Adam, 56 με τον SGD, συνεπώς περισσότερες παράμετροι εκμάθησης
- Ο αριθμός των εποχών της εκπαίδευσης: όπως ήταν αναμενόμενο, με τον Adam η εκπαίδευση- επικύρωση ολοκληρώθηκε σε 14 εποχές, με τον SGD η δοκιμή ολοκληρώθηκε σε 245 εποχές και προφανώς σε μεγαλύτερο χρονικό διάστημα. Σε προβλήματα του πραγματικού κόσμου, όπου τα δεδομένα είναι πολύ μεγαλύτερου όγκου, παίζει πολύ μεγάλο ρόλο ο χρόνος εκπαίδευσης - επικύρωσης, ο οποίος μπορεί να διαρκέσει από πολλές ώρες, μέχρι και ημέρες.

Συνεπώς, επιβεβαιώνεται η υπεροχή του Adam έναντι του SGD.

6.4.2 Υπερμοντέλα MLP Δύο Κρυφών Επιπέδων

Οι επόμενες υλοποιήσεις αφορούν υπερμοντέλα με δύο κρυφά επίπεδα χωρίς επιπλέον εξομάλυνση πέραν του πρόωρου σταματήματος, καθώς και με επιπλέον μεθόδους εξομάλυνσης. Για κάθε υπερμοντέλο παραθέτουμε τη σύνοψη του υπερμοντέλου και τα σχετικά αρχεία υλοποίησης και αποθηκευμένου μοντέλου και στην επόμενη παράγραφο θα παρουσιάσουμε και θα συγκρίνουμε τα αποτελέσματά τους.

6.4.2.1 MLP Δύο Κρυφών Επιπέδων χωρίς επιπλέον εξομάλυνση

Η εφαρμογή του Hyperband tuner με πρόωρο σταμάτημα, είχε ως αποτέλεσμα 8 νευρώνες ανά κρυφό επίπεδο και ρυθμό εκμάθησης 0.01.

Παραθέτουμε τη σύνοψη του υπερμοντέλου και τα σχετικά αρχεία:

Σύνοψη μοντέλου

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 8)                 248
_________________________________________________________________
dense_1 (Dense)              (None, 8)                 72
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 9
=================================================================
Total params: 329
Trainable params: 329
Non-trainable params: 0
```

Αποθήκευση και σώσιμο μοντέλου

Σχετικό notebook:

Hyperband_tuning_Hidden_best_8.ipynb

Αποθηκευμένο μοντέλο

Hyperband_2_8_best.h5

6.4.2.2 MLP Δύο Κρυφών Επιπέδων με Dropout

Η εφαρμογή του Hyperband tuner με πρόωρο σταμάτημα και επιπλέον εξομάλυνση Dropout= 0.2, είχε ως αποτέλεσμα 32 νευρώνες ανά κρυφό επίπεδο και ρυθμό εκμάθησης 0.01.

Παραθέτουμε τη σύνοψη του υπερμοντέλου και τα σχετικά αρχεία:

Σύνοψη μοντέλου Dropout

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 32)                992
_________________________________________________________________
module_wrapper (ModuleWrappe (None, 32)                0
_________________________________________________________________
dense_1 (Dense)              (None, 32)                1056
_________________________________________________________________
module_wrapper_1 (ModuleWrap (None, 32)                0
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 33
=================================================================
Total params: 2,081
Trainable params: 2,081
Non-trainable params: 0
_________________________________________________________________
```

Αποθήκευση και σώσιμο μοντέλου Dropout

Σχετικό notebook:

Hyperband_tuning_Hidden_best_drop_32.ipynb

6.4.2.3 MLP Δύο Κρυφών Επιπέδων με Batch Normalization

Η εφαρμογή του Hyperband tuner με πρόωρο σταμάτημα και επιπλέον εξομάλυνση Batch Normalization σε κάθε επίπεδο, είχε ως αποτέλεσμα 32 νευρώνες ανά κρυφό επίπεδο και ρυθμό εκμάθησης 0.01. Παραθέτουμε τη σύνοψη του υπερμοντέλου και τα σχετικά αρχεία:

Σύνοψη μοντέλου Batch Normalization

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 32)                992
_________________________________________________________________
module_wrapper (ModuleWrappe (None, 32)                128
_________________________________________________________________
dense_1 (Dense)              (None, 32)                1056
_________________________________________________________________
module_wrapper_1 (ModuleWrap (None, 32)                128
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 33
=================================================================
Total params: 2,337
Trainable params: 2,209
Non-trainable params: 128
_________________________________________________________________
```

Αποθήκευση και σώσιμο μοντέλου Batch Normalization

Σχετικό notebook:

Hyperband_tuning_Hidden_best_batch_32.ipynb

6.4.2.4 MLP Δύο Κρυφών Επιπέδων με L2 Regularization

Η εφαρμογή του Hyperband tuner με πρόωρο σταμάτημα και επιπλέον εξομάλυνση L2 Regularization στο πρώτο επίπεδο, είχε ως αποτέλεσμα 40 νευρώνες ανά κρυφό επίπεδο και ρυθμό εκμάθησης 0.001. Παραθέτουμε τη σύνοψη του υπερμοντέλου και τα σχετικά αρχεία:

Σύνοψη μοντέλου L2 Regularization

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 8)                 248
_________________________________________________________________
dense_1 (Dense)              (None, 8)                 72
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 9
=================================================================
Total params: 329
Trainable params: 329
Non-trainable params: 0
```

6.4.2.5 Αποθήκευση και σώσιμο μοντέλου L2 Regularization

Σχετικό notebook:

Hyperband_tuning_Hidden_8_Best_L2.ipynb

Αποθηκευμένο μοντέλο

Hyperband_2_8_L2.h5

6.4.3 Συγκριτικά Αποτελέσματα MLPs Υπερμοντέλων Δύο Κρυφών Επιπέδων

Έχοντας ολοκληρώσει πλέον την ανάπτυξη των μοντέλων, παραθέτουμε τα αποτελέσματα για κάθε μοντέλο συγκεντρωτικά σε πίνακες. Σε κάθε πίνακα η πρώτη στήλη (Model- Regularizer) αφορά την αρχιτεκτονική και την προετοιμασία του μοντέλου. Για παράδειγμα, η αναφορά 2-(8-8), olr= .01 L2 Regularization αντιστοιχεί στα αποτελέσματα του μοντέλου με δύο κρυφά επίπεδα με 8 μονάδες στο καθένα, το olr (optimal learning rate) αφορά το βέλτιστο ρυθμό μάθησης για τον Adam και το L2 Regularization αντιστοιχεί στην πρόσθετη μέθοδο εξομάλυνσης, εκτός από το EarlyStopping που εφαρμόζεται σε όλα τα μοντέλα. Πιο συγκεκριμένα:

Στον Πίνακα 6-10 παρουσιάζονται τα αποτελέσματα της εκπαίδευσης και της επικύρωσης με μετρικές την ακρίβεια και τη συνάρτηση κόστους, καθώς και ο αριθμός των εποχών που διήρκησε η εκπαίδευση.

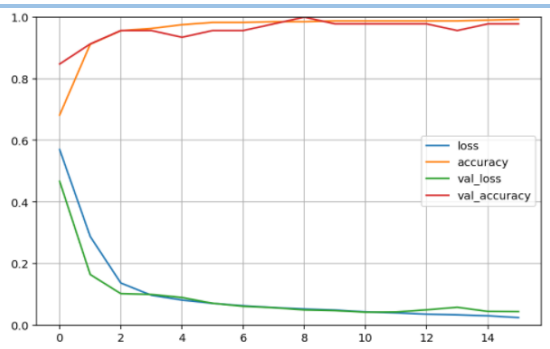
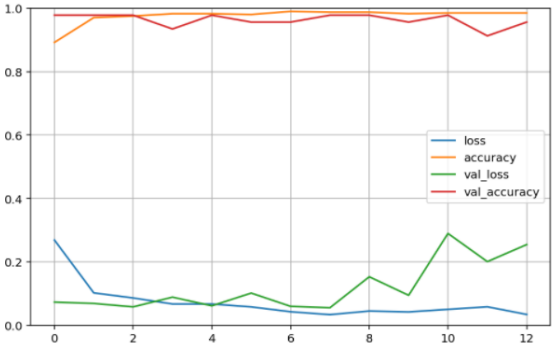
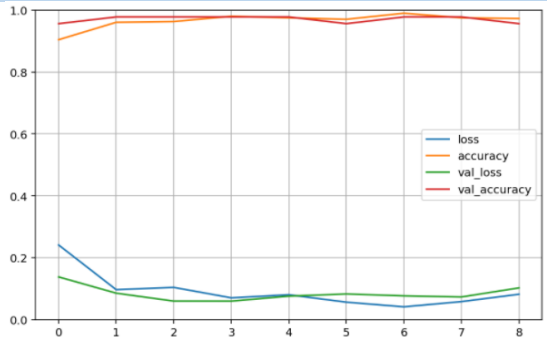
Όπως ήταν αναμενόμενο, η εκπαίδευση ενός μοντέλου διαρκεί λιγότερες εποχές για ρυθμό μάθησης 0.01. Η ορθότητα για όλα τα μοντέλα στο train set είναι πάνω από 99%, ενώ τις καλύτερες μετρικές έχει το μοντέλο 2-(32-32) με πρόσθετη εξομάλυνση Batch Normalization με ρυθμό μάθησης 0.01.

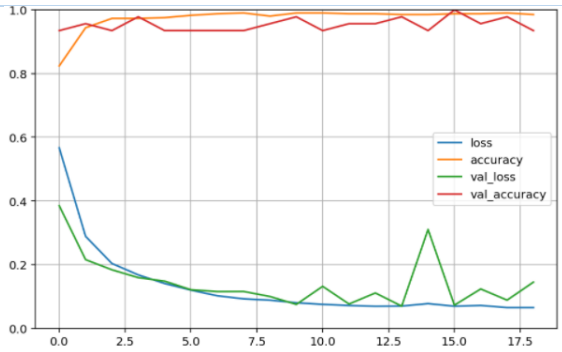
Model - Regularizer	Accuracy	Loss	Epochs
2-(8,8), olr= .01	0.991228	0.0230	16
2-(32-32), olr= .01 Dropout 0.2	0.991228	0.0373	13
2-(32-32), olr= .01 Batch Normalization	0.991228	0.0587	9

2-(8-8), olr= .01 L2 Regularization	0.991228	0.0483	19

Πίνακας 6-10. Σύγκριση accuracy και loss εκπαίδευσης υπερμοντέλων

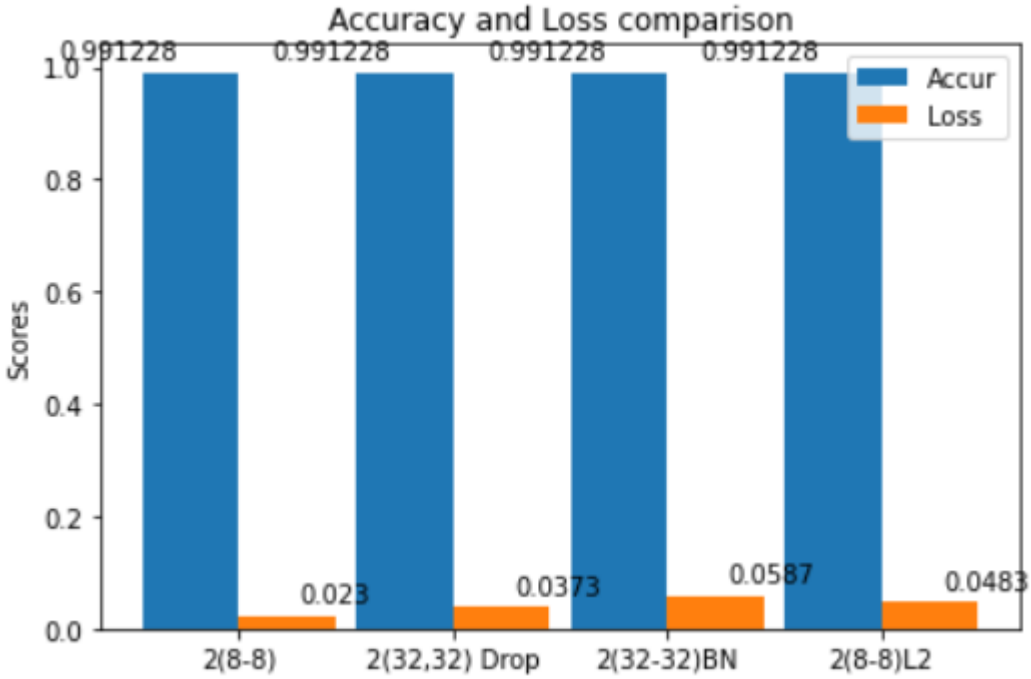
Στον Πίνακα 6-11 παρουσιάζονται οι καμπύλες εκπαίδευσης, όπου μπορούμε να δούμε την εξέλιξη της εκπαίδευσης – επικύρωσης στις εποχές.

Model - Regularizer	**Καμπύλες εκπαίδευσης - αποτίμησης**
2-(8,8), olr= .01	
2-(32-32), olr= .01 Dropout	
2-(32-32), olr= .01 Batch Normalization	

2-(8-8), olr= .01 L2 Regularization	

Πίνακας 6-11. Καμπύλες εκπαίδευσης – αποτίμησης υπερμοντέλων

Η συγκριτική αποτίμηση των μοντέλων στο σύνολο δοκιμής, όσον αφορά την ακρίβεια και τη συνάρτηση κόστους, δίνεται στην Εικόνα 6.-12. Στην αποτίμηση, τα καλύτερα αποτελέσματα έδωσε το μοντέλο 2(32-32) με BN, γιατί έχει την χαμηλότερη τιμή συνάρτησης κόστους.

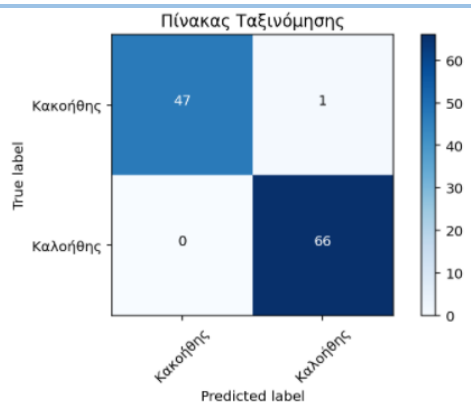
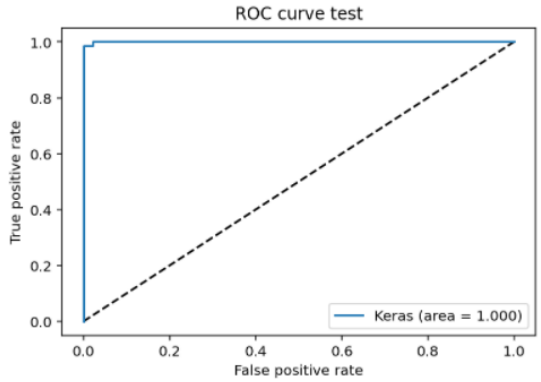
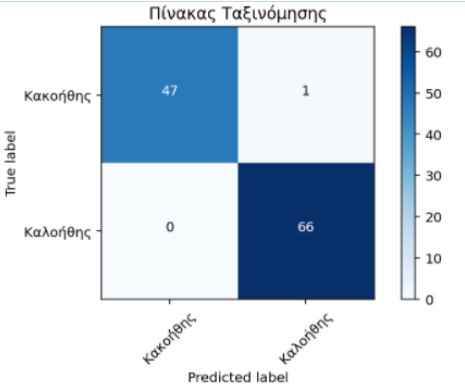
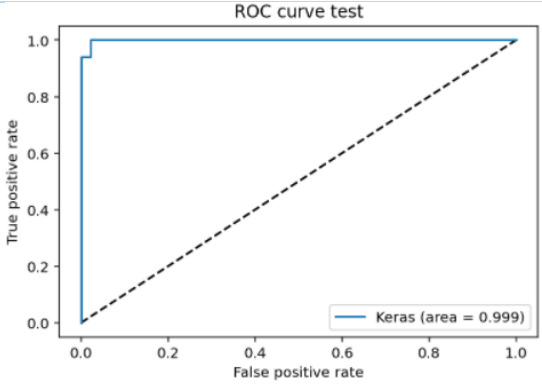
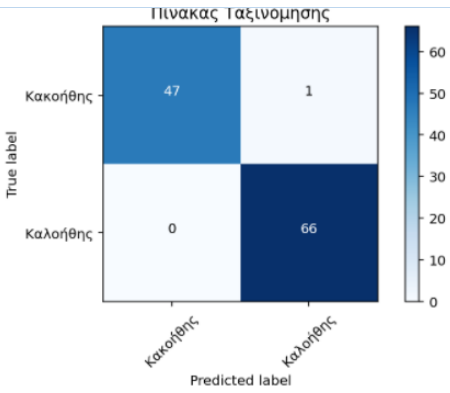
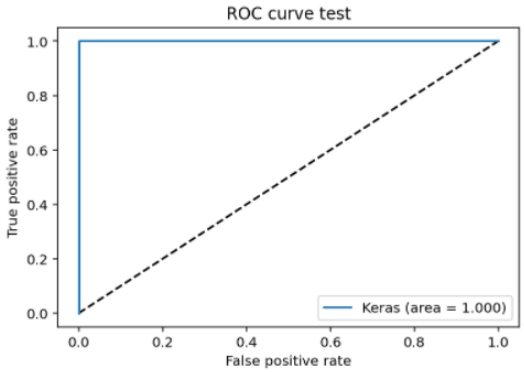


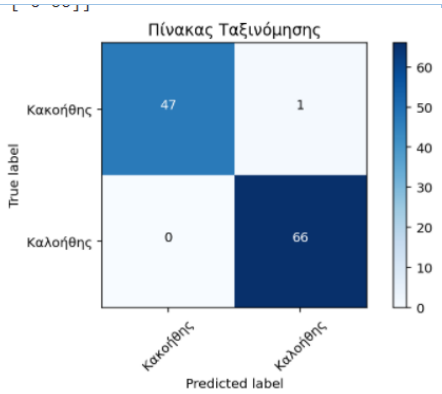
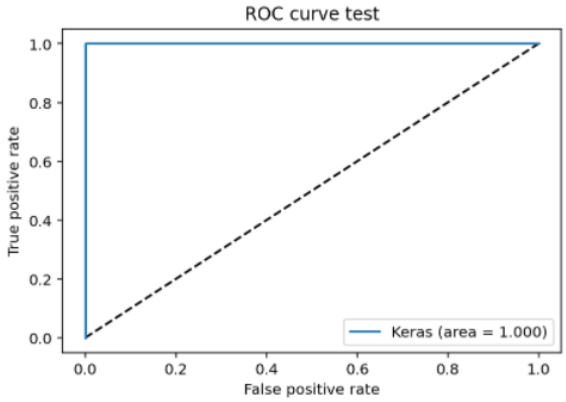
Εικόνα 6-12. Σύγκριση υπερμοντέλων στην αποτίμηση ως προς ακρίβεια και συνάρτηση κόστους

Στον Πίνακα 6-12 παρουσιάζονται τα συνολικά αποτελέσματα της αναφοράς της ταξινόμησης με μετρικές accuracy, loss, precision, recall, F1 score και στον Πίνακα 6-13 παρουσιάζονται συγκεντρωτικά τα γραφήματα για τους πίνακες ταξινόμησης και τις ROC - AUC καμπύλες των μοντέλων.

Model	Αναφορά ταξινόμησης
2-(8,8), olr= .01	precision recall f1-score support 0 1.00 0.98 0.99 48 1 0.99 1.00 0.99 66 accuracy 0.99 114 macro avg 0.99 0.99 0.99 114 weighted avg 0.99 0.99 0.99 114
2-(32-32), olr= .01 Dropout 0.2	precision recall f1-score support 0 1.00 0.98 0.99 48 1 0.99 1.00 0.99 66 accuracy 0.99 114 macro avg 0.99 0.99 0.99 114 weighted avg 0.99 0.99 0.99 114
2-(32-32), olr= .01 Batch Norm	precision recall f1-score support 0 1.00 0.98 0.99 48 1 0.99 1.00 0.99 66 accuracy 0.99 114 macro avg 0.99 0.99 0.99 114 weighted avg 0.99 0.99 0.99 114
2-(8-8), olr= .01 L2 Reg	precision recall f1-score support 0 1.00 0.98 0.99 48 1 0.99 1.00 0.99 66 accuracy 0.99 114 macro avg 0.99 0.99 0.99 114 weighted avg 0.99 0.99 0.99 114

Πίνακας 6-12. Σύγκριση μετρικών απόδοσης συνόλου δοκιμής υπερμοντέλων

Model	Πίνακας Ταξινόμησης	Καμπύλη ROC - AUC
2- (32,32) olr= .01		
2- (64,32) olr= .01 Dropou t 0.2		
2- (32-32) olr= .01 Batch Norm		

Model	Πίνακας Ταξινόμησης	Καμπύλη ROC - AUC
2- (8-8) olr= .01 L2 Reg		

Πίνακας 6-13. Σύγκριση πινάκων ταξινόμησης και καμπύλης ROC - AUC μοντέλων

Το αξιοσημείωτο είναι ότι, όλα τα μοντέλα αποτιμώνται ως τέλεια με AUC= 1.00 και πολύ ικανοποιητικές έως άριστες τιμές για τις υπόλοιπες μετρικές. Αυτό σημαίνει ότι, για όλα τα μοντέλα έχει επιτευχθεί ο στόχος της γενίκευσης, ειδικότερα για το απλό μοντέλο χωρίς επιπλέον εξομάλυνση.

Τα καλύτερα αποτελέσματα πρόγνωσης με άριστες μετρικές, παρουσιάζουν τα μοντέλα 2(32-32) με dropout= 0.2 και ρυθμό μάθησης 0.01, καθώς και μοντέλο 2-(8-8) με L2 Regularization στο πρώτο επίπεδο και ρυθμό μάθησης 0.01.

Συνεπώς, αποδεικνύεται ότι, με μοντέλα μικρής χωρητικότητας, δηλαδή δύο επιπέδων με μικρό αριθμό νευρώνων ανά επίπεδο, με λίγες παραμέτρους εκμάθησης και τη χρήση προηγμένων βιβλιοθηκών για ΒΜ, επιλύεται το πρόβλημα της δυαδικής ταξινόμησης της διάγνωσης του καρκίνου του στήθους για ένα μικρό σύνολο δεδομένων, όπως αυτό του Wisconsin breast cancer diagnosis.

7 Συμπεράσματα και Μελλοντικές Κατευθύνσεις

Στο κεφάλαιο αυτό παρουσιάζονται τα συμπεράσματα που εξήχθησαν από την παρούσα ΜΔΕ, καθώς και οι μελλοντικές κατευθύνσεις.

7.1.1 Συμπεράσματα

Στην παρούσα ΜΔΕ προσπαθήσαμε να υλοποιήσουμε MLPs για τη διάγνωση του καρκίνου του μαστού με βάση το σύνολο δεδομένων WDBC, με τη βοήθεια της βιβλιοθήκης BM Keras και τις δυνατότητες που προσφέρουν τα notebooks και το υπολογιστικό νέφος της Google μέσω του Colaboratory. Αντλώντας την πληροφορία από πολλαπλές πηγές και φιλτράροντας τη γνώση που αποκομίσαμε, καταφέραμε να υλοποιήσουμε μοντέλα τα οποία είχαν άριστες επιδόσεις όσο αφορά όχι μόνο την βελτιστοποίηση, αλλά και τη γενίκευση.

Με πολλούς πειραματισμούς δημιουργήσαμε διάφορα απλά μοντέλα με χρήση μεθόδων εξομάλυνσης. Πετύχαμε να υλοποιήσουμε ένα απλό μοντέλο δύο κρυφών επιπέδων, μικρής χωρητικότητας με 16 και 8 νευρώνες ανά κρυφό επίπεδο αντίστοιχα, το οποίο κατά την εκπαίδευση- αποτίμηση μας έδωσε ορθότητα 99,12% και 2,88% απώλεια στο σύνολο εκπαίδευσης που διήρκησε 160 εποχές εφαρμόζοντας ως μέθοδο εξομάλυνσης μόνο το πρόωρο σταμάτημα. Είχαμε 100% σχεδόν ορθότητα και μετρικές προγνώσεων με F1 = 1.00 και AUC=1.0, πετύχαμε δηλαδή ένα τέλειο μοντέλο.

Κατόπιν εκμεταλλευτήκαμε τις δυνατότητες που μας δίνει η βιβλιοθήκη Keras Tuner για αυτόματη ρύθμιση κάποιων υπερπαραμέτρων και υλοποιήσαμε υπερμοντέλα ενός και δύο επιπέδων. Πετύχαμε να υλοποιήσουμε ένα υπερμοντέλο με ένα κρυφό επίπεδο με 16 νευρώνες, το οποίο κατά την εκπαίδευση- αποτίμηση μας έδωσε ορθότητα 99,12% και 2,206% απώλεια στο σύνολο εκπαίδευσης που διήρκησε 14 εποχές εφαρμόζοντας ως μέθοδο εξομάλυνσης μόνο το πρόωρο σταμάτημα. Είχαμε 100% ορθότητα και μετρικές προγνώσεων με F1 = 1.00 και AUC=1.0, πετύχαμε δηλαδή ένα τέλειο μοντέλο.
Επίσης, πετύχαμε να υλοποιήσουμε ένα υπερμοντέλο με δύο κρυφά επίπεδα με 32 νευρώνες ανά κρυφό επίπεδο και μέθοδο εξομάλυνσης dropout= 0.2, το οποίο κατά την εκπαίδευση- αποτίμηση μας έδωσε ορθότητα 99,12% και 3,73% απώλεια στο σύνολο εκπαίδευσης που διήρκησε μόλις 13

εποχές, εφαρμόζοντας ως μέθοδο εξομάλυνσης το πρόωρο σταμάτημα. Είχαμε 100% ορθότητα και μετρικές προγνώσεων με F1 = 1.00 και AUC=1.0, πετύχαμε δηλαδή την υλοποίηση ενός άριστου μοντέλου.

Τα MLPs μοντέλα και τα υπερμοντέλα που υλοποιήσαμε, υπερβαίνουν σε επιδόσεις τα MPL μοντέλα αντίστοιχων εργασιών του Agarap [60], των Hasan, Haque και Kabir [61] και των Prakash και Visakha [62].

Συνεπώς, θεωρούμε ότι η παρούσα ΜΔΕ πέτυχε τον σκοπό της.

7.1.2 Μελλοντικές κατευθύνσεις

Η κυριαρχία της ΒΜ σε προβλήματα επιβλεπόμενης μάθησης, όπως αυτό της ταξινόμησης είναι πλέον αναμφισβήτητη. Στη σημερινή εποχή, με την μεγάλη τεχνολογική πρόοδο των ιατρικών απεικονιστικών συστημάτων, η ιατρική διάγνωση βασίζεται πλέον στις πρωτογενείς εικόνες. Επίσης, τα δεδομένα στον ιατρικό τομέα είναι πλέον μεγάλα δεδομένα και τα σύνολα δεδομένων περιέχουν από χιλιάδες έως εκατομμύρια εγγραφές. Τα Πολυεπίπεδα perceptron είναι το σημείο εκκίνησης για προβλήματα ταξινόμησης με σύνολα δεδομένων που αποτελούνται από εικόνες, όμως η μεγάλη δύναμη των ΤΝΔ είναι τα συνελικτικά νευρωνικά δίκτυα. Έτσι, στρέφουμε τη μελλοντική μας εργασία σε δύο κατευθύνσεις:

- την υλοποίηση MLP για τη διάγνωση του καρκίνου του μαστού με σύνολα δεδομένων εικόνων
- την υλοποίηση CNN για τη διάγνωση του καρκίνου του μαστού με σύνολα δεδομένων εικόνων

Βιβλιογραφία

[1] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[2] A. Smiti, "When machine learning meets medical world: Current status and future challenges," *Comput. Sci. Rev.*, vol. 37, p. 100280, 2020.

[3] N. I. R. Yassin, S. Omran, E. M. F. El Houby, and H. Allam, "Machine learning techniques for breast cancer computer aided diagnosis using different image modalities: A systematic review," *Comput. Methods Programs Biomed.*, vol. 156, pp. 25–45, 2018.

[4] D. Houfani, S. Slatnia, O. Kazar, N. Zerhouni, A. Merizig, and H. Saouli, "Machine Learning Techniques for Breast Cancer Diagnosis: Literature Review," in *Advanced Intelligent Systems for Sustainable Development (AI2SD'2019)*, 2020, pp. 247–254.

[5] N. Fatima, L. Liu, S. Hong, and H. Ahmed, "Prediction of Breast Cancer, Comparative Review of Machine Learning Techniques, and Their Analysis," *IEEE Access*, vol. 8, pp. 150360–150376, 2020.

[6] G. Murtaza *et al.*, "Deep learning-based breast cancer classification through medical imaging modalities: state of the art and research challenges," *Artif. Intell. Rev.*, vol. 53, no. 3, pp. 1655–1720, 2020.

[7] A. F. M. Agarap, "On breast cancer detection: An application of machine learning algorithms on the Wisconsin diagnostic dataset," *ACM Int. Conf. Proceeding Ser.*, no. 1, pp. 5–9, 2018.

[8] F. Basciftci and H. T. ÜNAL, "An Empirical Comparison of Machine Learning Algorithms for Predicting Breast Cancer," *Bilge Int. J. Sci. Technol. Res.*, vol. 3, no. 201 9, pp. 9–20, 2019.

[9] UCI Machine Learning Repository, "Breast Cancer Wisconsin (Diagnostic) Data Set." [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Diagnostic%29. [Accessed: 31-Aug-2020].

[10] "UC Irvine Machine Learning Repository." [Online]. Available: https://archive.ics.uci.edu/ml/index.php. [Accessed: 31-Aug-2020].

[11] O. L. Mangasarian, W. N. Street, and W. H. Wolberg, "Breast Cancer Diagnosis and

Prognosis Via Linear Programming," *Oper. Res.*, vol. 43, no. 4, pp. 570–577, 1995.

[12] World Health Organization, "Cancer." [Online]. Available: https://www.who.int/health-topics/cancer#tab=tab_1. [Accessed: 31-Aug-2020].

[13] International Agency for Recearch on Cancer (IACR), "Cancer Today." [Online]. Available: https://gco.iarc.fr/today/online-analysis-pie. [Accessed: 31-Aug-2020].

[14] European Cancer Information System, "ECIS - European Cancer Information System Measuring cancer burden and its time trends across Europe." [Online]. Available: https://ecis.jrc.ec.europa.eu/index.php. [Accessed: 31-Aug-2020].

[15] Wikipedia, "Breast." [Online]. Available: https://en.wikipedia.org/w/index.php?title=Breast&oldid=968115085. [Accessed: 31-Aug-2020].

[16] American Cancer Society, "What Is Breast Cancer?" [Online]. Available: https://www.cancer.org/cancer/breast-cancer/about/what-is-breast-cancer.html. [Accessed: 31-Aug-2020].

[17] Άγιος Σάββας - Γενικό Αντικαρκινικό Ογκολογικό Νοσοκομείο Αθηνών, "Καρκίνος μαστού." [Online]. Available: http://agsavvas-hosp.gr/Μάθεγιατονκαρκίνο/Πρόληψη/Πρωτογενήςπρόληψη/Καρκίνοςμαστού.aspx. [Accessed: 31-Aug-2020].

[18] American Cancer Society, "Imaging Tests to Find Breast Cancer." [Online]. Available: https://www.cancer.org/cancer/breast-cancer/screening-tests-and-early-detection.html. [Accessed: 31-Aug-2020].

[19] American Cancer Society, "Breast Biopsy." [Online]. Available: https://www.cancer.org/cancer/breast-cancer/screening-tests-and-early-detection/breast-biopsy.html. [Accessed: 31-Aug-2020].

[20] P. Sarkar, V. Davoodnia, and A. Etemad, "Computer-Aided Diagnosis using Class-Weighted Deep Neural Network," in *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, 2019, pp. 410–413.

[21] Α. Γεωργούλη, *Τεχνητή νοημοσύνη. [ηλεκτρ. βιβλ.]*. Αθήνα:Σύνδεσμος Ελληνικών Ακαδημαϊκών Βιβλιοθηκών.Διαθέσιμο στο: http://hdl.handle.net/11419/3381, 2015.

[22] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd Edition*. O'Reilly Media, Inc., 2019.

[23] F. Chollet, *Deep Learning with Python, Second Edition*. Manning Early Access Program (MEAP), 2020.

[24] L. Fridman, “MIT Deep Learning Basics: Introduction and Overview with TensorFlow.” [Online]. Available: https://medium.com/tensorflow/mit-deep-learning-basics-introduction-and-overview-with-tensorflow-355bcd26baf0. [Accessed: 31-Aug-2020].

[25] Massachusetts Institute of Technology (MIT), “MIT 6.S191 Introduction to Deep Learning.” [Online]. Available: http://introtodeeplearning.com/. [Accessed: 31-Aug-2020].

[26] M. A. Nielsen, *Neural Networks and Deep Learning*. Determination Press, 2015.

[27] L. Fridman, “Deep Learning Basics.” [Online]. Available: https://www.dropbox.com/s/c0g3sc1shi63x3q/deep_learning_basics.pdf?dl=0. [Accessed: 30-Sep-2020].

[28] A. Anandkumar, “What are the most important machine learning trends as we head into 2020?” [Online]. Available: https://www.quora.com/What-are-the-most-important-machine-learning-trends-as-we-head-into-2020. [Accessed: 28-Sep-2020].

[29] Y. Zhou, F. Dong, Y. Liu, Z. Li, J. Du, and L. Zhang, “Forecasting emerging technologies using data augmentation and deep learning,” *Scientometrics*, vol. 123, no. 1, pp. 1–29, 2020.

[30] A. Burkov, *The Hundred-Page Machine Learning Book*. Andriy Burkov (January 12, 2019), 2019.

[31] J. Watt, R. Borhani, and A. K. Katsaggelos, “Machine Learning Refined.” [Online]. Available: https://github.com/jermwatt/machine_learning_refined. [Accessed: 30-Sep-2020].

[32] B. Ding, H. Qian, and J. Zhou, “Activation functions and their characteristics in deep neural networks,” in *2018 Chinese Control And Decision Conference (CCDC)*, 2018, pp. 1836–1841.

[33] S. Ruder, “An overview of gradient descent optimization algorithms,” *CoRR*, vol. abs/1609.0, 2016.

[34] L. Bottou, “Stochastic Gradient Descent Tricks,” in *Montavon G., Orr G.B., Müller KR. (eds) Neural Networks: Tricks of the Trade. Lecture Notes in Computer Science, vol 7700.*, Springer, Berlin, Heidelberg, 2012.

[35] S. Ruder, “An overview of gradient descent optimization algorithms.” [Online]. Available: https://ruder.io/optimizing-gradient-descent/. [Accessed: 15-Sep-2020].

[36] J. Chen, “An updated overview of recent gradient descent algorithms.” [Online]. Available: https://johnchenresearch.github.io/demon/. [Accessed: 25-Sep-2020].

[37] 3Blue1Brown, “Backpropagation calculus.” [Online]. Available: https://www.youtube.com/watch?v=tIeHLnjs5U8. [Accessed: 15-Sep-2020].

[38] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A Simple Way to Prevent Neural Networks from Overfitting,” *J. Mach. Learn. Res.*, vol. 15, no. 56, pp. 1929–1958, 2014.

[39] J. Kukacka, V. Golkov, and D. Cremers, “Regularization for Deep Learning: A Taxonomy,” *CoRR*, vol. abs/1710.1, 2017.

[40] S. Ioffe and C. Szegedy, “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift,” *CoRR*, vol. abs/1502.03167, 2015.

[41] “Python.” [Online]. Available: https://www.python.org/. [Accessed: 10-Oct-2020].

[42] “Colaboratory.” [Online]. Available: https://colab.research.google.com/. [Accessed: 10-Oct-2020].

[43] P. Virtanen *et al.*, “SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python,” *Nat. Methods*, vol. 17, pp. 261–272, 2020.

[44] “SciPy.” [Online]. Available: https://www.scipy.org/. [Accessed: 10-Oct-2020].

[45] C. R. Harris *et al.*, “Array programming with NumPy,” *Nature*, vol. 585, pp. 357–362, 2020.

[46] “NumPy - The fundamental package for scientific computing with Python.” [Online]. Available: https://numpy.org/.

[47] “pandas.” [Online]. Available: https://pandas.pydata.org/. [Accessed: 10-Oct-2020].

[48] J. D. Hunter, “Matplotlib: A 2D Graphics Environment,” *Comput. Sci. Eng.*, vol. 9, no. 3, pp. 90–95, 2007.

[49] “Matplotlib: Visualization with Python.” [Online]. Available: https://matplotlib.org/. [Accessed: 10-Oct-2020].

[50] “seaborn: statistical data visualization.” [Online]. Available: https://seaborn.pydata.org/. [Accessed: 10-Oct-2020].

[51] F. Pérez and B. E. Granger, “IPython: a System for Interactive Scientific Computing,” *Comput. Sci. Eng.*, vol. 9, no. 3, pp. 21–29, 2007.

[52] F. Pedregosa *et al.*, “Scikit-learn: Machine Learning in Python,” *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.

[53] “scikit-learn Machine Learning in Python.” [Online]. Available: https://scikit-learn.org/. [Accessed: 10-Oct-2020].

[54] Martin Abadi *et al.*, “TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems.” 2015.

[55] “TensorFlow.” [Online]. Available: https://www.tensorflow.org/. [Accessed: 10-Oct-2020].

[56] F. Chollet and others, “Keras.” GitHub, 2015.

[57] “Keras.” [Online]. Available: https://keras.io/. [Accessed: 10-Oct-2020].

[58] “Keras API.” [Online]. Available: https://keras.io/api/. [Accessed: 10-Oct-2020].

[59] P. Bhatia, “On Implementing Deep Learning Library from Scratch in Python.” [Online]. Available: https://towardsdatascience.com/on-implementing-deep-learning-library-from-scratch-in-python-c93c942710a8. [Accessed: 05-Oct-2020].

[60] A. F. Agarap, “Deep Learning using Rectified Linear Units (ReLU),” *CoRR*, vol. abs/1803.0, 2018.

[61] M. M. Hasan, M. R. Haque, and M. M. J. Kabir, “Breast Cancer Diagnosis Models Using PCA and Different Neural Network Architectures,” in *2019 International Conference on Computer, Communication, Chemical, Materials and Electronic Engineering (IC4ME2)*, 2019, pp. 1–4.

[62] S. S. Prakash and K. Visakha, “Breast Cancer Malignancy Prediction Using Deep Learning Neural Networks,” in *2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA)*, 2020, pp. 88–92.

Παράρτημα Α: Εκδόσεις Βιβλιοθηκών Έργου

Αρχείο Colab: Versions.ipynb

```
[1]  import tensorflow as tf
     import keras
     import sklearn
     import IPython
```

```
[2]  tf.__version__
```

```
'2.5.0'
```

```
[3]  keras.__version__
```

```
'2.5.0'
```

```
[4]  IPython.version_info
```

```
(5, 5, 0, '')
```

```
sklearn.show_versions()
```

```
System:
    python: 3.7.10 (default, May  3 2021, 02:48:31)  [GCC 7.5.0]
executable: /usr/bin/python3
   machine: Linux-5.4.104+-x86_64-with-Ubuntu-18.04-bionic

Python dependencies:
       pip: 19.3.1
setuptools: 57.0.0
   sklearn: 0.22.2.post1
     numpy: 1.19.5
     scipy: 1.4.1
    Cython: 0.29.23
    pandas: 1.1.5
matplotlib: 3.2.2
    joblib: 1.0.1

Built with OpenMP: True
```

Εικόνα ΠΑ1. Ισχύουσες εκδόσεις βιβλιοθηκών κατά την δημιουργία του έργου