Η μέθοδος Gradient Descent και η εφαρμογή της σε προβλήματα βελτιστοποίησης Νευρωνικών δικτύων.

Ονοματεπώνυμο: Σπηλιώτη Ειρήνη

Επιβλέπων Καθηγητής: Ανούσης Μιχαήλ , Καραχάλιος Νικόλαος

Ημερομηνία: 12/09/26

Περίληψη

Η παρούσα διπλωματική εργασία εξετάζει τη μέθοδο Gradient Descent και τον ρόλο της στην εκπαίδευση τεχνητών νευρωνικών δικτύων, συνδυάζοντας τη θεωρητική ανάλυση με μια πρακτική εφαρμογή σε περιβάλλον Python.

Αρχικά παρουσιάζονται οι βασικές έννοιες της Τεχνητής Νοημοσύνης, της Μηχανικής Μάθησης και των Τεχνητών Νευρωνικών Δικτύων, καθώς και η ιστορική εξέλιξη που οδήγησε από το μοντέλο του Perceptron στα σύγχρονα συστήματα Βαθιάς Μάθησης. Στη συνέχεια παρουσιάζονται οι θεμελιώδεις έννοιες της Γραμμικής Άλγεβρας, όπως τα διανύσματα, οι πίνακες, οι γραμμικοί μετασχηματισμοί, οι συναρτήσεις ενεργοποίησης και οι συναρτήσεις σφάλματος, οι οποίες αποτελούν τη μαθηματική βάση της λειτουργίας των νευρωνικών δικτύων.

Έπειτα παρουσιάζεται αναλυτικά η μέθοδος Gradient Descent ως πρόβλημα βελτιστοποίησης, περιγράφεται ο τρόπος με τον οποίο ενημερώνονται τα βάρη ενός νευρωνικού δικτύου, εξετάζεται η επίδραση του ρυθμού μάθησης στη σύγκλιση του αλγορίθμου και γίνεται σύγκριση των κυριότερων παραλλαγών της μεθόδου (Batch, Stochastic και Mini-batch Gradient Descent). Παράλληλα αναφέρονται βασικές δυσκολίες που εμφανίζονται κατά την εκπαίδευση των μοντέλων.

Στο πρακτικό μέρος της εργασίας αναπτύσσεται ένα απλό μοντέλο Perceptron με χειροκίνητη υλοποίηση του αλγορίθμου Gradient Descent, χωρίς χρήση έτοιμων αλγορίθμων εκπαίδευσης. Η υλοποίηση πραγματοποιείται σε περιβάλλον Jupyter Notebook με τη γλώσσα προγραμματισμού Python και αξιοποιεί πραγματικό σύνολο δεδομένων για την πρόβλεψη της τελικής επίδοσης μαθητών. Παρουσιάζονται όλα τα στάδια της διαδικασίας, από την προεπεξεργασία των δεδομένων και την εκπαίδευση του μοντέλου έως την αξιολόγηση της απόδοσής του με κατάλληλους δείκτες και την ερμηνεία των αποτελεσμάτων.

Η εργασία αναδεικνύει τη στενή σχέση μεταξύ της Γραμμικής Άλγεβρας, των μεθόδων βελτιστοποίησης και των νευρωνικών δικτύων, δείχνοντας πώς οι θεωρητικές μαθηματικές έννοιες μπορούν να εφαρμοστούν στην πράξη για την ανάπτυξη ενός απλού αλλά λειτουργικού συστήματος πρόβλεψης.

Λέξεις-κλειδιά: Τεχνητή Νοημοσύνη, Μηχανική Μάθηση, Perceptron, Gradient Descent, Γραμμική Άλγεβρα, Βελτιστοποίηση, Πρόβλεψη Επίδοσης Μαθητών, Python.

Abstract

This thesis examines the Gradient Descent optimization method and its role in the training of artificial neural networks by combining theoretical analysis with a practical implementation developed in Python.

The first part of the thesis introduces the fundamental concepts of Artificial Intelligence, Machine Learning and Artificial Neural Networks, presenting their historical evolution from the Perceptron model to modern Deep Learning architectures. It also discusses the mathematical foundations of neural networks through the concepts of vectors, matrices, linear transformations, activation functions and loss functions, highlighting the essential contribution of Linear Algebra to their operation.

The thesis then focuses on the Gradient Descent algorithm as a numerical optimization technique for training neural networks. Its mathematical formulation, the influence of the learning rate on convergence, and the main variants of the algorithm, namely Batch, Stochastic and Mini-batch Gradient Descent, are presented and analyzed. In addition, common challenges encountered during the training process are discussed.

The practical part of the thesis presents the implementation of a simple Perceptron model using a manually implemented Gradient Descent algorithm without relying on built-in optimization routines. The implementation was developed in Python using Jupyter Notebook and applied to a real educational dataset for predicting students' final academic performance. The complete workflow, including data preprocessing, model training, evaluation and interpretation of the obtained results, is described in detail.

Overall, this work demonstrates the close relationship between Linear Algebra, optimization techniques and Artificial Neural Networks, illustrating how fundamental mathematical concepts can be effectively applied to the development of an interpretable machine learning model.

Keywords: Artificial Intelligence, Machine Learning, Perceptron, Gradient Descent, Linear Algebra, Optimization, Student Performance Prediction, Python.

Περιεχόμενα

Περίληψη 2
Abstract 3
Κατάλογος Σχημάτων 5
Κατάλογος Πινάκων 6
Κατάλογος Συντομογραφιών 6
ΕΙΣΑΓΩΓΗ 7
Κεφάλαιο 1: Εισαγωγή στα Νευρωνικά Δίκτυα. 8
Ενότητα 1.1: Η Τεχνητή Νοημοσύνη ως Απαρχή των Νευρωνικών Δικτύων. 8
Ενότητα 1.2: Τι είναι η Μηχανική Μάθηση-Κλάδος της Τεχνητής Νοημοσύνης. 11
Ενότητα 1.3: Τι είναι τα Νευρωνικά Δίκτυα -Το Perceptron. 16
1.3.1 Από τα Βιολογικά στα Τεχνητά Νευρωνικά Δίκτυα. 16
1.3.2 Η δομή ενός τεχνητού νευρώνα. 17
1.3.3 Βασικές λειτουργίες ενός Νευρωνικού Δικτύου. 21
1.3.4 Χαρακτηριστικά και Περιορισμοί των Απλών Νευρωνικών Δικτύων. 23
1.3.5 Πολυεπίπεδα Νευρωνικά Δίκτυα (Multilayer Perceptrons - MLP). 24
1.4 Deep Learning: Θεμελιώδεις Αρχές και Λειτουργία. 25
Κεφάλαιο 2: Γραμμική Άλγεβρα και Νευρωνικά Δίκτυα. 26
2.1 Διανύσματα Εισόδου και Εξόδου. 26
2.2 Πίνακες Βαρών και Γραμμικοί Μετασχηματισμοί. 28
2.3 Συναρτήσεις Ενεργοποίησης και Μη Γραμμικοί Μετασχηματισμοί 30
2.4 Συνάρτηση Σφάλματος. 32
2.5 Προώθηση Πληροφορίας. 33
Κεφάλαιο 3: Εκπαίδευση με Gradient Descent. 34
3.1 Πρόβλημα Βελτιστοποίησης στην Εκπαίδευση Νευρωνικών Δικτύων. 35
3.2: Η βασική ιδέα της Gradient Descent και Μαθηματική διατύπωση. 37
3.3: Learning rate και σύγκλιση. 40
3.4: Batch vs Stochastic vs Mini-batch Gradient Descent. 43
3.5: Από πού προκύπτει το gradient στα Νευρωνικά Δίκτυα. 46
3.6: Συνήθη προβλήματα στην πράξη. 48

Κεφάλαιο 4: Υλοποίηση Έξυπνου Βοηθού Μελέτης με χρήση Perceptron και Gradient Descent. 51
4.1 Σχεδιασμός και αρχιτεκτονική του έξυπνου βοηθού μελέτης. 52
Επιλογή του Perceptron και σύνδεση με τον Gradient Descent 52
Αρχιτεκτονική της εφαρμογής 53
4.2 Ανάπτυξη και εκπαίδευση του μοντέλου πρόβλεψης. 54
4.2.3 Εισαγωγή βιβλιοθηκών. 54
4.2.2 Φόρτωση του συνόλου δεδομένων. 55
4.2.3 Επιλογή χαρακτηριστικών. 56
4.2.4 Διαχωρισμός σε σύνολο εκπαίδευσης και ελέγχου. 57
4.2.5 Κανονικοποίηση (τυποποίηση z-score) 58
4.2.6 Δημιουργία του Perceptron και συνάρτηση πρόβλεψης. 59
4.2.7 Συνάρτηση κόστους. 60
4.2.8 Χειροκίνητη υλοποίηση του Gradient Descent. 60
4.2.9 Εκπαίδευση του μοντέλου. 64
4.2.10 Εμφάνιση του loss ανά epoch. 65
4.2.11 Αξιολόγηση του μοντέλου. 66
4.3 Εφαρμογή, αποτελέσματα και αξιολόγηση του έξυπνου βοηθού μελέτης. 68
Λογική των συστάσεων 68
Βιβλιογραφία 72

Κατάλογος Σχημάτων

ΣΧΗΜΑ 1: Το διάγραμμα απεικονίζει τη λογική και γνωστική ιεραρχία των βασικών εννοιών που συνδέονται με το πεδίο της Τεχνητής Νοημοσύνης. 11
ΣΧΗΜΑ 2: Κύριες Μορφές Μάθησης στη Μηχανική Μάθηση. 14
ΣΧΗΜΑ 3: Το σχήμα αποτυπώνει τη μονοεπίπεδη, γραμμική φύση του Perceptron. 21
ΣΧΗΜΑ 4: Διάγραμμα Πολυεπίπεδου Νευρωνικού Δικτύου (MLP). 25
ΣΧΗΜΑ 5: Διαδικασία ενεργοποίησης ενός επιπέδου νευρωνικού δικτύου. 30
ΣΧΗΜΑ 6: Εφαρμογή της συνάρτησης ενεργοποίησης στο διάνυσμα προενεργοποιήσεων. 32
ΣΧΗΜΑ 7: *Διάγραμμα ροής λειτουργίας του Έξυπνου Βοηθού Μελέτης, από την είσοδο του χρήστη έως τις τελικές συστάσεις.* 54
ΣΧΗΜΑ 8: *Μείωση της συνάρτησης κόστους ανά epoch, για το σύνολο εκπαίδευσης και ελέγχου.* 65
ΣΧΗΜΑ 9: . *Προβλεπόμενη έναντι πραγματικής επίδοσης στο σύνολο ελέγχου· η διακεκομμένη γραμμή αντιστοιχεί στην τέλεια πρόβλεψη.* 67

ΣΧΗΜΑ 10: *Εκπαιδευμένα βάρη ανά χαρακτηριστικό (σε τυποποιημένη κλίμακα).*.....70

Κατάλογος Πινάκων

Πίνακας 1: Επιρροή του ρυθμού μάθησης. 41
Πίνακας 2: Συγκριτικός πίνακας της κλασσικής (Batch) Gradient Descent και των παραλλαγών Stochastic και Mini-batch ως προς τον τρόπο υπολογισμού του gradient και τη συμπεριφορά σύγκλισης. 46
Πίνακας 3: Χαρακτηριστικά εισόδου του έξυπνου βοηθού μελέτης. 56
Πίνακας 4: Ερμηνεία των τιμών των χαρακτηριστικών.....64
Πίνακας 5: Τέσσερις δείκτες αξιολόγησης του μοντέλου.....66
Πίνακας 6:Αποτελέσματα ενδεικτικού παραδείγματος. 69

Κατάλογος Συντομογραφιών

Συντομογραφία	Πλήρης ονομασία
AI	Artificial Intelligence (Τεχνητή Νοημοσύνη)
ML	Machine Learning (Μηχανική Μάθηση)
DL	Deep Learning (Βαθιά Μάθηση)
ANN	Artificial Neural Network (Τεχνητό Νευρωνικό Δίκτυο)
MLP	Multilayer Perceptron (Πολυεπίπεδο Perceptron)
GD	Gradient Descent
BGD	Batch Gradient Descent
SGD	Stochastic Gradient Descent
MBGD	Mini-Batch Gradient Descent
ReLU	Rectified Linear Unit

Συντομογραφία	Πλήρης ονομασία
MSE	Mean Squared Error (Μέσο Τετραγωνικό Σφάλμα)
MAE	Mean Absolute Error (Μέσο Απόλυτο Σφάλμα)
RMSE	Root Mean Squared Error (Τετραγωνική Ρίζα του Μέσου Τετραγωνικού Σφάλματος)
LR	Learning Rate (Ρυθμός Μάθησης)

Οι αγγλικές συντομογραφίες διατηρούνται στην πρωτότυπη μορφή τους, καθώς αποτελούν διεθνώς καθιερωμένους όρους στον τομέα της Τεχνητής Νοημοσύνης και της Μηχανικής Μάθησης**.**

ΕΙΣΑΓΩΓΗ

Η ραγδαία ανάπτυξη της Τεχνητής Νοημοσύνης και της Μηχανικής Μάθησης έχει οδηγήσει στη δημιουργία μοντέλων που χρησιμοποιούνται σήμερα σε ένα ευρύ φάσμα εφαρμογών, όπως η αναγνώριση εικόνας, η επεξεργασία φυσικής γλώσσας, η πρόβλεψη δεδομένων και τα συστήματα υποστήριξης αποφάσεων. Σημαντικό μέρος της εξέλιξης αυτής οφείλεται στα Τεχνητά Νευρωνικά Δίκτυα, τα οποία έχουν τη δυνατότητα να μαθαίνουν από δεδομένα και να προσαρμόζονται σε πολύπλοκα προβλήματα.

Η αποτελεσματική εκπαίδευση ενός νευρωνικού δικτύου εξαρτάται σε μεγάλο βαθμό από τους αλγορίθμους βελτιστοποίησης που χρησιμοποιούνται για την προσαρμογή των παραμέτρων του. Ανάμεσα στις σημαντικότερες μεθόδους βρίσκεται η Gradient Descent, η οποία αποτελεί τη θεμελιώδη τεχνική ελαχιστοποίησης της συνάρτησης κόστους και χρησιμοποιείται ευρέως στην εκπαίδευση μοντέλων μηχανικής μάθησης.

Σκοπός της παρούσας διπλωματικής εργασίας είναι η θεωρητική μελέτη της μεθόδου Gradient Descent και η διερεύνηση του ρόλου της στην εκπαίδευση τεχνητών νευρωνικών δικτύων. Παράλληλα, επιδιώκεται η σύνδεση των μαθηματικών εννοιών της Γραμμικής Άλγεβρας με τη λειτουργία των νευρωνικών δικτύων, καθώς και η πρακτική εφαρμογή τους μέσω της ανάπτυξης ενός απλού μοντέλου Perceptron.

Στο πρακτικό μέρος της εργασίας υλοποιείται σε περιβάλλον Python ένας έξυπνος βοηθός μελέτης, ο οποίος βασίζεται σε μοντέλο Perceptron και εκπαιδεύεται με χειροκίνητη υλοποίηση του αλγορίθμου Gradient Descent, χωρίς χρήση έτοιμων αλγορίθμων εκπαίδευσης. Η εφαρμογή αξιοποιεί πραγματικό σύνολο δεδομένων για την πρόβλεψη της τελικής επίδοσης μαθητών, παρουσιάζοντας αναλυτικά όλα τα στάδια της διαδικασίας, από την προεπεξεργασία των δεδομένων έως την αξιολόγηση των αποτελεσμάτων.

Η εργασία οργανώνεται σε τέσσερα κεφάλαια. Το πρώτο κεφάλαιο παρουσιάζει τις βασικές έννοιες της Τεχνητής Νοημοσύνης, της Μηχανικής Μάθησης και των Τεχνητών Νευρωνικών Δικτύων. Το δεύτερο κεφάλαιο αναλύει τα βασικά εργαλεία της Γραμμικής Άλγεβρας που χρησιμοποιούνται στη λειτουργία των νευρωνικών δικτύων. Το τρίτο κεφάλαιο επικεντρώνεται στη μέθοδο Gradient Descent, στις παραλλαγές της και στα κυριότερα ζητήματα που εμφανίζονται κατά την εκπαίδευση των μοντέλων. Τέλος, το τέταρτο κεφάλαιο παρουσιάζει την υλοποίηση της εφαρμογής, την εκπαίδευση του μοντέλου, τα αποτελέσματα και την αξιολόγησή τους.

Κεφάλαιο 1: Εισαγωγή στα Νευρωνικά Δίκτυα.

Σκοπός του Κεφαλαίου

Σκοπός του πρώτου κεφαλαίου είναι να παρουσιάσει τη σχέση ανάμεσα στην Τεχνητή Νοημοσύνη**,** τη Μηχανική Μάθηση και τα Νευρωνικά Δίκτυα. Αρχικά, γίνεται αναφορά των βασικών εννοιών της Τεχνητής Νοημοσύνης και της Μηχανικής Μάθησης, οι οποίες αποτελούν το θεωρητικό υπόβαθρο για την κατανόηση των Νευρωνικών Δικτύων. Στο πλαίσιο αυτό, γίνεται μια ιστορική αναδρομή στην εξέλιξη της ιδέας των νευρωνικών δικτύων, με αναφορά στο μοντέλο του perceptron. Στη συνέχεια, περιγράφεται η βασική δομή ενός νευρωνικού δικτύου και η εξέλιξή του τις τελευταίες δεκαετίες, τόσο σε θεωρητικό όσο και σε εφαρμοσμένο επίπεδο.

Ενότητα 1.1: Η Τεχνητή Νοημοσύνη ως Απαρχή των Νευρωνικών Δικτύων.

Η Τεχνητή Νοημοσύνη αποτέλεσε έναν από τους σημαντικότερους κλάδους αρχικά της Πληροφορικής, αλλά εξελίχθηκε σε διεπιστημονικό πεδίο που

συνδυάζει μαθηματικά, στατιστική, νευροεπιστήμη, ψυχολογία, γλωσσολογία και μηχανική.

Η ανάπτυξη της ήρθε ως απάντηση στο ερώτημα *«Μπορούν οι μηχανές να σκεφτούν;»* που διατυπώθηκε πρώτη φορά το 1950, με τη δημοσίευση του έργου *"Computing Machinery and Intelligence",από τον* Alan Turing**.** Με σκοπό τη διερεύνηση του ερωτήματος αυτού, πρότεινε το γνωστό Turing Test**,** ως μέσο αξιολόγησης της ικανότητας μιας μηχανής να επιδεικνύει ανθρώπινη νοημοσύνη μέσω διαλόγου. Το έργο αυτό θεωρείται η θεωρητική αφετηρία της Τεχνητής Νοημοσύνης [1].

Λίγα χρόνια αργότερα, το 1955 με πρωτοβουλία των John McCarthy και των συνεργατών του Marvin Minsky**,** Claude Shannon και Nathaniel Rochester**,** συντάχθηκε ερευνητική πρόταση που χρησιμοποιεί για πρώτη φορά τον όρο '*Artificial Intelligence'*. Η πρόταση τέθηκε, το 1956, στο συνέδριο του "Dartmouth Summer Research Project on Artificial Intelligence", το οποίο πραγματοποιήθηκε στο Dartmouth College των Ηνωμένων Πολιτειών, αναδεικνύοντας την πρακτική έναρξη της. Η συνάντηση αυτή ένωσε ερευνητές από διαφορετικούς τομείς όπως μαθηματικά, λογική, ψυχολογία και υπολογιστικά συστήματα. Ενώ παράλληλα, διαμόρφωσε το θεωρητικό υπόβαθρο πάνω στο οποίο αναπτύχθηκε ολόκληρος ο κλάδος της Τεχνητής Νοημοσύνης τις επόμενες δεκαετίες. Τη πρώτη δεκαετία λειτούργησε ως μέσο επίλυσης μαθηματικών προβλημάτων και δημιουργίας προγραμμάτων που εκτελούσαν παιχνίδια στρατηγικής [2].

Το 1959 εισήχθη για πρώτη φορά ο όρος *Machine Learning* από τον Arthur Samuel της IBM**,** κατά την διάρκεια των ερευνών του πάνω σε προγράμματα που μπορούσαν να μαθαίνουν αυτόνομα μέσω εμπειρίας. Το έργο του *"Some Studies in Machine Learning Using the Game of Checkers"* θεωρείται το πρώτο πρακτικό παράδειγμα αλγοριθμικής μάθησης, καθώς το πρόγραμμα που ανέπτυξε μάθαινε να παίζει ντάμα βελτιώνοντας σταδιακά τις επιδόσεις του χωρίς πρόσθετο προγραμματισμό [3]. Η συμβολή του Samuel σηματοδότησε τη μετάβαση από τη συμβολική Τεχνητή Νοημοσύνη σε πιο προσαρμοστικά υπολογιστικά μοντέλα**,** θέτοντας τα θεμέλια της σύγχρονης Μηχανικής Μάθησης, ως υποκλάδου πλέον της τεχνητής νοημοσύνης. Η Μηχανική Μάθηση έχει ως στόχο την δημιουργία συστημάτων που μπορούν να μαθαίνουν από δεδομένα και παραδείγματα, ανακαλύπτοντας μοτίβα, ώστε να βελτιώνονται σταδιακά χωρίς να έχουν προγραμματιστεί ρητά.

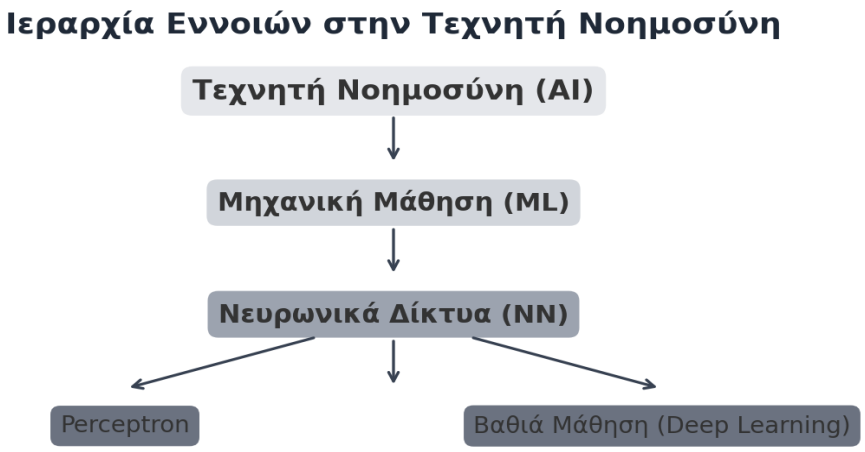
Η ανάγκη δημιουργίας τέτοιων συστημάτων οδήγησε στην ανάπτυξη των νευρωνικών δικτύων**,** τα οποία αποτελούν μία από τις σημαντικότερες κατηγορίες

αλγορίθμων της Μηχανικής Μάθησης και χρησιμοποιούνται ευρέως σε σύγχρονες εφαρμογές. Βασική ιδέα υπήρξε η προσομοίωση των βιολογικών κυττάρων σε τεχνητά νευρωνικά δίκτυα. Το πρώτο μοντέλο νευρωνικών δικτύων κάνει την εμφάνιση του, ήδη από το 1958, όπου ο Frank Rosenblatt δημιούργησε το Perceptron [4].

Παρ' όλη την άνθιση του κλάδου, τα τεχνικά και υπολογιστικά όρια της εποχής οδήγησαν σε περιόδους στασιμότητας, γνωστές ως "AI Winters"**.** Η εποχή αυτή εξελίσσεται την δεκαετία του 1970 και 1980, κατά τη διάρκεια της οποίας, ο επιχειρησιακός κόσμος αλλά και οι κυβερνήσεις φαίνεται να χάνουν τόσο το ενδιαφέρον τους όσο και την εμπιστοσύνη τους στην περαιτέρω εξέλιξη της τεχνητής νοημοσύνης, γεγονός που δικαιολογεί την ελάττωση της χρηματοδότησης προς τις έρευνες αυτής [5].

Η Τεχνητή Νοημοσύνη άρχισε να ανακάμπτει από την περίοδο ύφεσης (γνωστή ως *AI winter*) κατά τη διάρκεια της δεκαετία του 1980, καθώς επανήλθε η χρηματοδότηση και σημειώθηκε πρόοδος τόσο στο υλικό (hardware) όσο και στο λογισμικό (software)**.** Υπήρξε ανάπτυξη πιο αποδοτικών αλγορίθμων μάθησης, ενώ καθοριστικό ρόλο στην αναγέννηση της έπαιξε η διαθεσιμότητα των Big Data, καθώς κι η εξέλιξη της Μηχανικής Μάθησης (machine learning) [5].

Από το 1993 και ύστερα, η έρευνα στην Τεχνητή νοημοσύνη στράφηκε σε πιο εξειδικευμένα και στοχευμένα προβλήματα, τα οποία αντιμετωπίζονταν με αυστηρότερες επιστημονικές μεθόδους [5]. Στις αρχές του 21ου αιώνα, η εξέλιξη των Νευρωνικών Δικτύων και της Βαθιάς Μάθησης (Deep Learning)[6], έδωσε νέο ορίζοντα στις δυνατότητες της Τεχνητής Νοημοσύνης.



ΣΧΗΜΑ 1: Το διάγραμμα απεικονίζει τη λογική και γνωστική ιεραρχία των βασικών εννοιών που συνδέονται με το πεδίο της Τεχνητής Νοημοσύνης.

Πηγή: Ίδια επεξεργασία, βασισμένη στις πηγές [5], [7] και [9].

Ενότητα 1.2: Τι είναι η Μηχανική Μάθηση-Κλάδος της Τεχνητής Νοημοσύνης.

Σύμφωνα με τον Arthur Samuel (1959) έναν από τους πατέρες της μηχανικής μάθησης: *"Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed."*(η Μηχανική Μάθηση είναι το πεδίο που δίνει στους υπολογιστές τη δυνατότητα να μαθαίνουν χωρίς να έχουν προγραμματιστεί ρητά) [3]. Ένα από τα πρώτα πειραματικά παραδείγματα μηχανικής μάθησης που εφάρμοσε, παρουσιάστηκε στο άρθρο του *"Some Studies in Machine Learning Using the Game of Checkers"*, που δημοσιεύθηκε στο *IBM Journal of Research and Development* [3].

Στην εργασία αυτή, ο Samuel εφάρμοσε δύο διαφορετικές διαδικασίες μάθησης σε ένα πρόγραμμα που έπαιζε το παιχνίδι της ντάμας. Το πρόγραμμα, αφού έλαβε μόνο τους κανόνες του παιχνιδιού και ένα σύνολο παραμέτρων χωρίς προκαθορισμένα βάρη, κατόρθωσε μέσα σε λίγες ώρες να παίζει καλύτερα από τον

ίδιο τον δημιουργό του. Το συγκεκριμένο πείραμα έδειξε ότι ένας υπολογιστής μπορεί να βελτιώνει την απόδοσή του αξιοποιώντας την εμπειρία που αποκτά. Η εργασία του Samuel αποτέλεσε σημαντικό βήμα για την ανάπτυξη αλγορίθμων που μαθαίνουν από δεδομένα και επηρέασε την εξέλιξη της Μηχανικής Μάθησης τις επόμενες δεκαετίες [3].

Ένας πιο τυπικός ορισμός δόθηκε από τον Tom Mitchell, (1997): *Ένα πρόγραμμα μαθαίνει από εμπειρία Ε, σε σχέση με μια τάξη εργασιών Τ και ένα μέτρο απόδοσης Ρ, αν η απόδοσή του στις εργασίες της Τ, όπως μετράται από το Ρ, βελτιώνεται με την εμπειρία Ε* [7].

Οι αλγόριθμοι μηχανικής μάθησης αποτελούν τον πυρήνα της διαδικασίας εξόρυξης δεδομένων (data mining), καθώς επιτρέπουν την αυτόματη ανακάλυψη προτύπων, σχέσεων και κανόνων μέσα από μεγάλα και σύνθετα σύνολα δεδομένων [8].

Η εκμάθηση κανόνων μπορεί να επιτευχθεί με διάφορες προσεγγίσεις, όπως οι συμβολικοί αλγόριθμοι οι οποίοι περιγράφουν τη γνώση μέσω λογικών κανόνων τύπου “εάν-τότε”. Εναλλακτικά, μέσω αλγορίθμων δέντρων απόφασης, που αναπαριστούν τη διαδικασία λήψης αποφάσεων με ιεραρχική δομή. Σε περιπτώσεις όπου τα δεδομένα είναι εξαιρετικά μεγάλα, χρησιμοποιούνται εξειδικευμένοι αλγόριθμοι με έμφαση στην υπολογιστική αποδοτικότητα, προκειμένου να μειωθεί το κόστος εκπαίδευσης και να επιταχυνθεί η διαδικασία ανάλυσης [8].

Επιπρόσθετα, σύγχρονες τεχνικές, όπως τα νευρωνικά δίκτυα, οι πιθανοτικοί (Bayesian) αλγόριθμοι, καθώς και ο επαγωγικός λογικός προγραμματισμός, επεκτείνουν τη δυνατότητα κατανόησης και πρόβλεψης πολύπλοκων και μη γραμμικών φαινομένων [9]. Σύμφωνα, με τον *Mitchell (1997)*, οι αλγόριθμοι αυτοί βασίζονται σε στατιστικές αρχές μάθησης, οι οποίες επιτρέπουν στο σύστημα να γενικεύει την εμπειρία του σε νέες περιπτώσεις, μια διαδικασία που αποτελεί τον θεμελιώδη στόχο της Μηχανικής Μάθησης [7].

Παρ’ όλα αυτά, η επιτυχία της εξόρυξης δεδομένων δεν εξαρτάται μόνο από τον αλγόριθμο μάθησης. Εξίσου σημαντική είναι η σωστή προετοιμασία των δεδομένων, η οποία περιλαμβάνει τον καθαρισμό, τον μετασχηματισμό και την επιλογή των κατάλληλων χαρακτηριστικών. Στο τελικό στάδιο, η οπτικοποίηση των αποτελεσμάτων και η αξιολόγησή τους από ειδικούς συμβάλλουν στην εξαγωγή χρήσιμων συμπερασμάτων [8].

Συνεπώς, η Μηχανική Μάθηση λειτουργεί ως κεντρικό εργαλείο μέσα σε ένα ευρύτερο οικοσύστημα τεχνικών και διαδικασιών που μετατρέπει τα δεδομένα σε χρήσιμη γνώση και πρακτική αξία**.**

Η διαδικασία έχει τρία βασικά βήματα:

- Εκπαίδευση (Training):

  Ο αλγόριθμος εκπαιδεύεται χρησιμοποιώντας δεδομένα εισόδου και τις αντίστοιχες επιθυμητές εξόδους. Μαθαίνει τα πρότυπα που συνδέουν εισόδους και εξόδους.

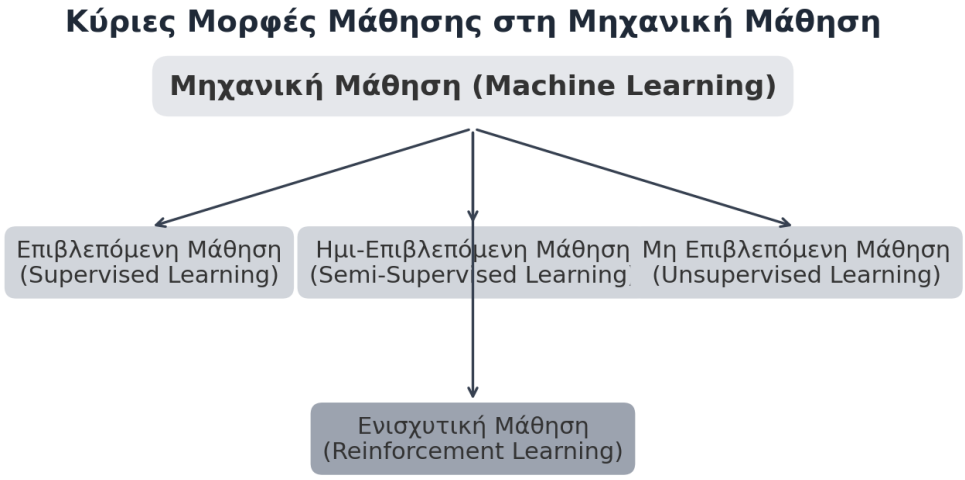
- Μάθηση Μοντέλου (Model Learning):

  Το σύστημα δημιουργεί ένα μαθηματικό μοντέλο που περιγράφει τη σχέση αυτή.

- Πρόβλεψη ή Απόφαση (Prediction):

  Όταν του δοθούν νέα δεδομένα, χρησιμοποιεί το μοντέλο για να προβλέψει ή να αποφασίσει.

Η Μηχανική Μάθηση χωρίζεται σε τρεις κύριες κατηγορίες, ανάλογα με το είδος μάθησης. Αρχικά, έχουμε την Επιβλεπόμενη (Supervised Learning) κατά την οποία ο αλγόριθμος εκπαιδεύεται με δεδομένα που έχουν γνωστές απαντήσεις. Για παράδειγμα, η αναγνώριση προσώπου, η πρόβλεψη των τιμών σε σπίτια.
Εν συνεχεία, παρουσιάζεται η Μη Επιβλεπόμενη (Unsupervised Learning). Εδώ, ο αλγόριθμος προσπαθεί να βρει πρότυπα ή ομάδες μέσα σε δεδομένα χωρίς γνωστές απαντήσεις. Χαρακτηριστικά παραδείγματα είναι η ανάλυση της αγοραστικής συμπεριφοράς, η ομαδοποίηση πελατών.
Τέλος, η Ενισχυτική (Reinforcement Learning) αποτελεί το είδος της μάθησης όπου ο αλγόριθμος μαθαίνει μέσω αλληλεπίδρασης με το περιβάλλον, λαμβάνοντας επιβραβεύσεις ή ποινές. Τα παιχνίδια στρατηγικής, η ρομποτική και τα αυτόνομα οχήματα είναι ορισμένα παραδείγματα που βρίσκει εφαρμογή το συγκεκριμένο είδος μάθησης.
Ως συνέπεια των βασικών αυτών κατηγοριών, δημιουργήθηκε και η Ημι-επιβλεπόμενη Μάθηση (Semi-Supervised Learning). Δεν είναι μία από τις αρχικές βασικές κατηγορίες, αλλά προέκυψε αργότερα, ως ενδιάμεση προσέγγιση ανάμεσα στη Επιβλεπόμενη (Supervised) και τη Μη Επιβλεπόμενη (Unsupervised) μάθηση. Συναντάται στην επεξεργασία ιατρικών εικόνων με λίγες ετικέτες (labels) [5], [7].



ΣΧΗΜΑ 2: Κύριες Μορφές Μάθησης στη Μηχανική Μάθηση.

Πηγή: Ίδια επεξεργασία, βασισμένη στις πηγές [5], [7] και [8].

Στη Μηχανική Μάθηση έχουν αναπτυχθεί πολλές διαφορετικές κατηγορίες αλγορίθμων, καθεμία από τις οποίες είναι κατάλληλη για διαφορετικά προβλήματα και τύπους δεδομένων. Ανάλογα με τον τρόπο με τον οποίο αναπαριστούν τη γνώση και πραγματοποιούν την εκπαίδευση, οι σημαντικότερες κατηγορίες μοντέλων συνοψίζονται στις ακόλουθες.

Γραμμικά Μοντέλα (Linear Models).
Τα γραμμικά μοντέλα αποτελούν την απλούστερη και περισσότερο θεμελιωμένη κατηγορία αλγορίθμων της Μηχανικής Μάθησης. Στόχος τους είναι η περιγραφή της σχέσης μεταξύ εισόδων και εξόδων μέσω μιας γραμμικής συνάρτησης. Στην κατηγορία αυτή ανήκουν η **Γραμμική Παλινδρόμηση (Linear Regression)**, η οποία χρησιμοποιείται για προβλήματα πρόβλεψης συνεχών τιμών, καθώς και η **Λογιστική Παλινδρόμηση (Logistic Regression)**, η οποία εφαρμόζεται κυρίως σε προβλήματα ταξινόμησης. Τα γραμμικά μοντέλα αποτελούν τη βάση για πολλές

σύγχρονες τεχνικές βελτιστοποίησης και θα αξιοποιηθούν στο πρακτικό μέρος της παρούσας εργασίας [7].

Μοντέλα Δέντρων Αποφάσεων (Tree-Based Models).
Τα μοντέλα αυτά οργανώνουν τη διαδικασία λήψης αποφάσεων σε μορφή δέντρου, χωρίζοντας σταδιακά τα δεδομένα σύμφωνα με συγκεκριμένα κριτήρια. Χρησιμοποιούνται ευρέως σε προβλήματα ταξινόμησης και παλινδρόμησης, ιδιαίτερα όταν τα δεδομένα είναι δομημένα. Χαρακτηριστικά παραδείγματα αποτελούν τα **Decision Trees** και οι επεκτάσεις τους, όπως τα **Random Forests**.

Πιθανοτικά Μοντέλα (Probabilistic Models).
Τα πιθανοτικά μοντέλα βασίζονται στις αρχές της θεωρίας πιθανοτήτων και της στατιστικής, εκτιμώντας την πιθανότητα εμφάνισης συγκεκριμένων γεγονότων ή κατηγοριών. Εφαρμόζονται σε προβλήματα όπως η ταξινόμηση κειμένων, η αναγνώριση προτύπων και η ανάλυση ακολουθιών δεδομένων. Ένα από τα πιο γνωστά παραδείγματα αποτελεί ο ταξινομητής Naive Bayes [5].

Μοντέλα Νευρωνικών Δικτύων (Neural Network Models).
Τα Τεχνητά Νευρωνικά Δίκτυα αποτελούν μία από τις σημαντικότερες κατηγορίες αλγορίθμων της σύγχρονης Μηχανικής Μάθησης. Εμπνέονται από τη λειτουργία των βιολογικών νευρώνων και έχουν τη δυνατότητα να προσεγγίζουν πολύπλοκες, μη γραμμικές σχέσεις μεταξύ των δεδομένων. Στην κατηγορία αυτή περιλαμβάνονται το Perceptron, τα Πολυεπίπεδα Perceptrons (MLP), καθώς και πιο εξειδικευμένες αρχιτεκτονικές, όπως τα Συνελικτικά Νευρωνικά Δίκτυα (CNN) και τα Αναδρομικά Νευρωνικά Δίκτυα (RNN). Η εκπαίδευσή τους βασίζεται σε αλγορίθμους βελτιστοποίησης, με σημαντικότερο τη Gradient Descent, η οποία αποτελεί το κύριο αντικείμενο της παρούσας εργασίας [9].

Μοντέλα Ομαδοποίησης και Μείωσης Διαστάσεων (Clustering and Dimensionality Reduction).
Η κατηγορία αυτή περιλαμβάνει τεχνικές που χρησιμοποιούνται κυρίως στη μη επιβλεπόμενη μάθηση, με σκοπό είτε την ομαδοποίηση παρόμοιων δεδομένων είτε τη μείωση της διάστασής τους, ώστε να διευκολύνεται η ανάλυση και η οπτικοποίηση. Χαρακτηριστικά παραδείγματα αποτελούν ο αλγόριθμος K-Means και η Ανάλυση Κυρίων Συνιστωσών (Principal Component Analysis, PCA) [8].

Συνοψίζοντας, η Μηχανική Μάθηση περιλαμβάνει ένα ευρύ σύνολο αλγορίθμων, οι οποίοι επιλέγονται ανάλογα με το πρόβλημα που καλούνται να επιλύσουν και τα χαρακτηριστικά των διαθέσιμων δεδομένων. Στην παρούσα διπλωματική εργασία το ενδιαφέρον επικεντρώνεται στα Τεχνητά Νευρωνικά Δίκτυα, καθώς η εκπαίδευσή τους πραγματοποιείται μέσω τεχνικών βελτιστοποίησης, με κυριότερη τη Gradient Descent, η οποία αναλύεται εκτενώς στα επόμενα κεφάλαια.

Ενότητα 1.3: Τι είναι τα Νευρωνικά Δίκτυα -Το Perceptron.

1.3.1 Από τα Βιολογικά στα Τεχνητά Νευρωνικά Δίκτυα.

Η λειτουργία του ανθρώπινου εγκέφαλου αποτέλεσε τον βασικό πυρήνα για την ιδέα και την υλοποίηση των Τεχνητών Νευρωνικών Δικτύων (Artificial Neural Networks, ANN). Η έμπνευση προήλθε από το πως οι βιολογικοί νευρώνες επικοινωνούν μεταξύ τους και επεξεργάζονται πληροφορίες, ενώ η διαδικασία της υλοποίησης εστιάζει στο πως να μιμηθούμε αυτή τη διαδικασία με μαθηματικό και υπολογιστικό τρόπο. Αρχή αποτέλεσε ο συνδυασμός της επιστήμης των μαθηματικών με τη νευροεπιστήμη, και η μελέτη ενός πολύπλοκου βιολογικού συστήματος του ανθρώπινου εγκέφαλου που περιλαμβάνει νευρώνες, οργανωμένους σε πολυεπίπεδα δίκτυα αλληλεπιδρώντων κυττάρων. Οι νευρώνες επικοινωνούν μεταξύ τους μέσω ηλεκτροχημικών σημάτων, σχηματίζοντας ένα εκτεταμένο σύστημα μετάδοσης και επεξεργασίας πληροφοριών [9].

Τα τρία βασικά δομικά στοιχεία ενός βιολογικού νευρώνα είναι:

- δενδρίτες, μέσω των οποίων λαμβάνει εισερχόμενα σήματα από άλλους νευρώνες,
- το κυτταρικό σώμα (soma), όπου πραγματοποιείται η ολοκλήρωση και επεξεργασία των σημάτων αυτών,
- τον άξονα (axon), μέσω του οποίου μεταδίδει την έξοδο σε άλλους νευρώνες ή σε άλλους τύπους κυττάρων [9].

Η λειτουργία ενός νευρώνα μπορεί να απλοποιηθεί σε μια διαδοχή τριών σταδίων:

- Λήψη και συγκέντρωση εισερχόμενων σημάτων.
- Εσωτερική επεξεργασία και υπολογισμός της συνολικής διέγερσης.
- Παραγωγή ενός εξερχόμενου σήματος, όταν η διέγερση υπερβεί συγκεκριμένο κατώφλι ενεργοποίησης [9].

Αν και βιολογικά πολύπλοκη, η διαδικασία επεξεργασίας της πληροφορίας, ωστόσο αποτέλεσε το θεμέλιο για την ανάπτυξη του τεχνητού νευρώνα**,** που έχει χαρακτηριστεί ως ένα αφαιρετικό μαθηματικό μοντέλο που επιχειρεί να μιμηθεί τον τρόπο με τον οποίο οι βιολογικοί νευρώνες συνδυάζουν και επεξεργάζονται τα ερεθίσματα. Το μοντέλο αυτό εισήχθη από τους McCulloch και Pitts το 1943 και αποτέλεσε τη βάση για τη διαμόρφωση των σύγχρονων Νευρωνικών Δικτύων και για την εξέλιξη της αντίστοιχης ερευνητικής περιοχής [10].

Ο τεχνητός νευρώνας (artificial neuron), αποτελεί το βασικό δομικό στοιχείο των Τεχνητών Νευρωνικών Δικτύων και σχεδιάστηκε με σκοπό να αναπαραστήσει τη διαδικασία λήψης, ολοκλήρωσης και μετάδοσης σημάτων, όπως αυτή παρατηρείται στα βιολογικά νευρικά συστήματα [9].

1.3.2 Η δομή ενός τεχνητού νευρώνα.

Σε υπολογιστικό επίπεδο, ο τεχνητός νευρώνας δέχεται n εισόδους, οι οποίες μπορεί να είναι οποιαδήποτε πληροφορία που περιγράφει το πρόβλημα π.χ. αριθμητικά δεδομένα (numeric inputs), όπως ηλικία ατόμου, η τιμή ενός προϊόντος, θερμοκρασία κ.α., κατηγορικά δεδομένα (categorical inputs), όπως φύλο, χρώμα κ.α., εικόνες (image inputs), τα pixels μιας εικόνας. Επίσης, μπορεί να δεχτεί εισόδους όπως ο ήχος (audio inputs), καθώς τα ακουστικά σήματα μετατρέπονται σε αριθμητικές τιμές, κείμενο (text inputs) που μετατρέπεται σε αριθμούς μέσω word embeddings κλπ. και χρονικές σειρές (time series inputs) όπως μετοχές, καρδιακοί παλμοί, καιρικά δεδομένα.

Οι είσοδοι είναι:

$$x_1, x_2, \dots, x_n,$$

καθεμία από τις οποίες συσχετίζεται με ένα βάρος

$$w_1, w_2, \dots, w_n,$$

που εκφράζει τη συνεισφορά ή τη σχετική σημασία της αντίστοιχης εισόδου στον τελικό υπολογισμό [9].

Τα βάρη είναι αριθμοί που καθορίζουν πόσο σημαντική είναι κάθε είσοδος για τον νευρώνα. Η σημασία μιας εισόδου σε ένα νευρωνικό δίκτυο καθορίζεται από το πόσο συμβάλλει στη μείωση του σφάλματος και στη βελτίωση της απόδοσης του

μοντέλου. Για παράδειγμα, για τη πρόβλεψη της τιμής ενός σπιτιού τα τετραγωνικά είναι σημαντική είσοδος, ενώ το χρώμα των τοίχων όχι. Στην ουσία, αν προσθέτοντας ένα χαρακτηριστικό η ακρίβεια ανεβαίνει τότε είναι σημαντικό ενώ αν δεν ανεβαίνει ή πέφτει τότε είναι άχρηστο ή θορυβώδες. Αν μια είσοδος είναι σημαντική το βάρος της πρέπει να είναι μεγάλο (π.χ. 2.0), ενώ αν είναι ασήμαντη, το βάρος γίνεται μικρό ή κοντά στο μηδέν. Η επιλογή τους δεν είναι αυθαίρετη ούτε χειροκίνητη, αλλά βασίζεται σε ορισμένους κανόνες. Πριν ξεκινήσει η εκπαίδευση, τα βάρη πρέπει να έχουν κάποιες αρχικές τιμές. Πριν από την εκπαίδευση, τα βάρη αρχικοποιούνται με μικρές τυχαίες τιμές. Κατά την εκπαίδευση, τα βάρη δεν μένουν στα ίδια νούμερα. Αλλάζουν συνεχώς μέσω της Gradient Descent:

$$w := w - \eta \frac{\partial J}{\partial w}$$

όπου:

- J: συνάρτηση κόστους
- η: learning rate
- $\frac{\partial J}{\partial w}$: πόσο επηρεάζει το βάρος το λάθος

Εν κατακλείδι,ο στόχος τους είναι να μειώσουν το λάθος του δικτύου και να κάνουν σωστές προβλέψεις [9].

Κάθε είσοδος x_i συνδέεται με τον αντίστοιχο συντελεστή βάρους w_i. Αυτό επεξηγεί ότι οι είσοδοι δεν πολλαπλασιάζονται όλες με τον ίδιο αριθμό, αλλά αντιστοιχίζονται σε ένα μοναδικό βάρος η καθεμία.
Η διαδικασία επεξεργασίας των εισόδων πραγματοποιείται μέσω μιας συνάρτησης συσσώρευσης (Aggregation Function), η οποία παίρνει τις εισόδους και τα βάρη και τα συνδυάζει σε ένα νούμερο. Στις περισσότερες περιπτώσεις είναι ένας γραμμικός συνδυασμός της μορφής:

$$z = \sum_{i=1}^{n} w_i x_i + b,$$

όπου b αποτελεί τον όρο μετατόπισης (bias), ο οποίος επιτρέπει την προσαρμογή της εξόδου ανεξάρτητα από τις τιμές των εισόδων [9]. Πιο αναλυτικά, μετατοπίζει τη συνάρτηση ενεργοποίησης, της δίνει τη δυνατότητα να μετακινηθεί δεξιά , αριστερά, πάνω και κάτω ανάλογα με το πρόβλημα. Είναι σαν να μετακινείς μια ευθεία πάνω στο γράφημα, καθώς λειτουργεί όπως ο σταθερός όρος (+b) στις εξισώσεις στην άλγεβρα:

$$y = ax + b$$

Το b είναι ο όρος που μετατοπίζει την ευθεία πάνω/κάτω. Ακριβώς το ίδιο κάνει και στο νευρώνα.

Χωρίς bias, η συνάρτηση ενεργοποίησης περνάει πάντα από την αρχή (0,0). Με bias, το δίκτυο μπορεί να μάθει διαφορετικές γραμμές διαχωρισμού, περισσότερα patterns, καθώς και πολύ πιο σύνθετες σχέσεις [9].

Το μέγεθος z που συμβολίζει το αποτέλεσμα της συνάρτησης συσσώρευσης, αποτελεί τη συνολική ενεργοποίηση του νευρώνα. Εν συνεχεία, το z διέρχεται από μια συνάρτηση ενεργοποίησης $f(\cdot)$ (Activation Function), η οποία το μετατρέπει σε τελική έξοδο του νευρώνα:

$$y = f(z).$$

Η συνάρτηση ενεργοποίησης μπορεί να λάβει διαφορετικές μορφές ανάλογα με το εκάστοτε πρόβλημα και η επιλογή της είναι καθοριστικής σημασίας, καθώς επηρεάζει τόσο την απόκριση του νευρώνα όσο και την ικανότητα του δικτύου να προσεγγίσει πολύπλοκες μη γραμμικές συναρτήσεις [9]. Οι επικρατέστερες μορφές συναρτήσεων ενεργοποίησης, καθεμία εκ των οποίων παρουσιάζει διαφορετικές ιδιότητες, είναι:

- Βηματική συνάρτηση (step function): χρησιμοποιήθηκε στα πρώιμα μοντέλα λογικής απόφασης και αποτελεί δυαδική συνάρτηση κατωφλίου.

- Tanh (υπερβολική εφαπτομένη):

$$f(z) = \tanh(z)$$

έχει εύρος τιμών $(-1,1)$ και συμβάλλει σε ταχύτερη σύγκλιση σε ορισμένα μοντέλα.

- ReLU (Rectified Linear Unit):

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

χρησιμοποιείται ευρέως στα σύγχρονα νευρωνικά δίκτυα, καθώς είναι υπολογιστικά απλή και συμβάλλει στην αποτελεσματικότερη εκπαίδευση βαθιών δικτύων.

Οι συναρτήσεις αυτές εισάγουν μη γραμμικότητα στο μοντέλο [9]. Η Μη γραμμικότητα ισοδυναμεί με την ικανότητα ενός μοντέλου να μην συμπεριφέρεται σαν ευθεία γραμμή. Του δίνει τη δυνατότητα να μπορεί να μάθει καμπύλες, πολύπλοκα σχήματα, μοτίβα με πολλές μεταβλητές ή προβλέψεις που αλλάζουν με πολύπλοκο τρόπο. Για το λόγο αυτό καθίσταται απαραίτητη προϋπόθεση για να καταφέρει ένα νευρωνικό δίκτυο να προσεγγίζει πολύπλοκες σχέσεις μεταξύ εισόδων και εξόδων. Χωρίς αυτήν, ακόμη και δίκτυα πολλαπλών επιπέδων θα είχαν την μορφή ενός απλού γραμμικού μετασχηματισμού, περιορίζοντας δραστικά την δυναμική τους [9].

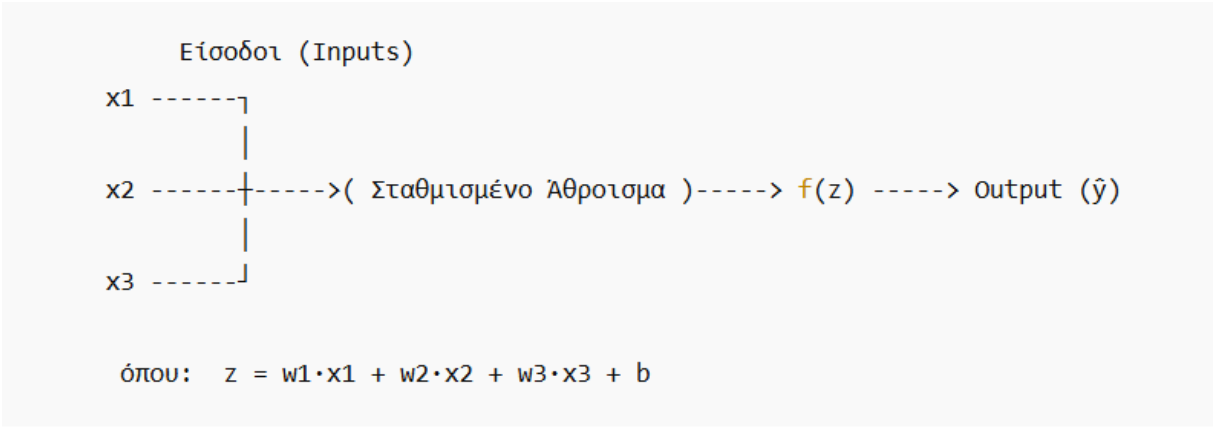
Το είδος της εξόδου εξαρτάται από το πρόβλημα και από την ενεργοποίηση που επιλέγεις. Ο νευρώνας μπορεί να δώσει για έξοδο δυαδική έξοδο (0 ή 1) για προβλήματα αποφάσεων, οποιοδήποτε πραγματικό αριθμό για προβλέψεις, πιθανότητες κυρίως σε προβλήματα πολυ-ταξινόμησης. Τέλος σε πιο σύνθετα μοντέλα δίνει ως έξοδο διάνυσμα [9].

Συνεπώς, ο τεχνητός νευρώνας αποτελεί το λειτουργικό υπόβαθρο των νευρωνικών δικτύων, ενώ οι παράμετροί του τα βάρη, bias και συνάρτηση ενεργοποίησης αποτελούν τα εργαλεία μέσω των οποίων το δίκτυο δύναται να εκπαιδευτεί και να προσαρμοστεί σε δεδομένα αυξανόμενης πολυπλοκότητας.

1.3.3 Βασικές λειτουργίες ενός Νευρωνικού Δικτύου.

Ένα Τεχνητό Νευρωνικό Δίκτυο (Artificial Neural Network , ANN) αποτελεί ένα πολυεπίπεδο υπολογιστικό μοντέλο, το οποίο δομείται από ένα σύνολο τεχνητών νευρώνων οργανωμένων σε διακριτά επίπεδα [9].

Με αφετηρία το Perceptron, ορίζεται η πιο χαρακτηριστική και ιστορικά σημαντική μορφή απλού νευρωνικού δικτύου, ενός μοντέλου με μόνο ένα επίπεδο, όπου οι είσοδοι συνδέονται απευθείας με τον νευρώνα εξόδου, χωρίς την ύπαρξη κρυφών επιπέδων [9]. Αποτελεί την τυπική υλοποίηση ενός single-layer neural network, το οποίο μπορεί να μάθει αποκλειστικά και μόνο γραμμικά διαχωρίσιμα προβλήματα. Επομένως, όταν αναφερόμαστε στα «απλά νευρωνικά δίκτυα» [4], στην πραγματικότητα περιγράφουμε την υπολογιστική αρχή που εισήγαγε το Perceptron, το οποίο χρησιμοποιείται ως το βασικό μοντέλο για την κατανόηση της λειτουργίας και των περιορισμών των αρχικών νευρωνικών δομών.



ΣΧΗΜΑ 3*: Το σχήμα αποτυπώνει τη μονοεπίπεδη, γραμμική φύση του Perceptron.*

Πηγή:Προσαρμογή από [9].

Από τα απλά νευρωνικά δίκτυα οδηγούμαστε στη δημιουργία πολυεπίπεδων νευρωνικών δικτύων, η αρχιτεκτονική των οποίων διαμορφώνεται από τρία κύρια είδη επιπέδων:

- Επίπεδο εισόδου (Input Layer), το οποίο δέχεται τα πρωτογενή δεδομένα και τα εισάγει στο δίκτυο χωρίς καμία μορφή επεξεργασίας [9].
- Κρυφά επίπεδα (Hidden Layers), όπου πραγματοποιείται η πλειονότητα των υπολογιστικών μετασχηματισμών μέσω συνδυασμού σταθμισμένων αθροισμάτων και μη γραμμικών συναρτήσεων ενεργοποίησης [9].

- Επίπεδο εξόδου (Output Layer), το οποίο παράγει την τελική απόκριση του δικτύου, προσαρμοσμένη στο εκάστοτε πρόβλημα (π.χ. ταξινόμηση, παλινδρόμηση, πιθανότητες κατηγοριών) [9].

Η λειτουργία ενός ANN διέπεται από δύο θεμελιώδεις διαδικασίες:

(α)Μπροστινή διάδοση (Forward Propagation).

Κατά τη διαδικασία της μπροστινής διάδοσης, τα πρωτογενή δεδομένα εισάγονται στο επίπεδο εισόδου. Έπειτα μεταφέρονται διαδοχικά μέσω των κρυφών επιπέδων έως ότου παραχθεί η τελική έξοδος. Σε κάθε νευρώνα, οι είσοδοι πολλαπλασιάζονται με τα αντίστοιχα βάρη, αθροίζονται μαζί με τον όρο μετατόπισης (bias) και εν συνεχεία διέρχονται από μη γραμμική συνάρτηση ενεργοποίησης [9].

(β) Εκπαίδευση και προσαρμογή βαρών (Training).

Η εκπαίδευση ενός νευρωνικού δικτύου (training phase) πραγματοποιείται με την επαναληπτική προσαρμογή των παραμέτρων του κυρίως των βαρών και του bias, με στόχο την ελαχιστοποίηση της διαφοράς μεταξύ της παραγόμενης εξόδου και της επιθυμητής τιμής. Είναι στην ουσία, η διαδικασία κατά την οποία το δίκτυο βελτιώνεται, αλλάζοντας τα βάρη του κάθε φορά που κάνει λάθος [9]. Η διαφορά αυτή παριστάνεται μέσω μιας συνάρτησης κόστους (loss function), η οποία ποσοτικοποιεί το σφάλμα του μοντέλου.

Στην περίπτωση της επιβλεπόμενης μάθησης, η διαδικασία εκπαίδευσης περιλαμβάνει:

1. υπολογισμό του σφάλματος,
2. διάδοση της πληροφορίας σφάλματος προς τα πίσω (μέσω κατάλληλου μηχανισμού όπως το backpropagation),
3. και τροποποίηση των βαρών με τρόπο που οδηγεί στη σταδιακή ελαχιστοποίηση της συνάρτησης κόστους.

Η προσαρμογή των βαρών πραγματοποιείται μέσω αλγορίθμων βελτιστοποίησης, εκ των οποίων η Gradient Descent αποτελεί τη θεμελιώδη μέθοδο [9]. Η τεχνική αυτή βρίσκει προς ποια κατεύθυνση αυξάνεται περισσότερο το λάθος και αλλάζει τα βάρη προς την αντίθετη κατεύθυνση, ώστε το λάθος να μικραίνει συνεχώς, μέχρι το μοντέλο να φτάσει στο καλύτερο δυνατό σημείο.

Η διαδικασία αυτή είναι επαναληπτική και συνεχίζεται μέχρι το δίκτυο να επιτύχει ικανοποιητικά επίπεδα απόδοσης, σύμφωνα με κάποιο προκαθορισμένο κριτήριο σύγκλισης.

1.3.4 Χαρακτηριστικά και Περιορισμοί των Απλών Νευρωνικών Δικτύων.

Τα απλά νευρωνικά δίκτυα παρουσιάζουν αρκετά πλεονεκτήματα κυρίως ως πρώτο βήμα κατανόησης των νευρωνικών δικτύων. Η κατανόησή τους είναι απαραίτητη, καθώς αποτυπώνουν τη βασική ιδέα της υπολογιστικής μάθησης: εισόδους, βάρη, γραμμικό συνδυασμό, ενεργοποίηση και προσαρμογή μέσω εκπαίδευσης. Επιπροσθέτως, διαθέτουν χαμηλό υπολογιστικό κόστος, γεγονός που τα κάνει κατάλληλα για μικρά και σχετικά απλά προβλήματα, ενώ παράλληλα είναι εύκολα στην υλοποίηση.

Βασικό χαρακτηριστικό τους, αλλά εν συνεχεία και περιορισμός τους είναι ότι μπορούν να λύσουν προβλήματα γραμμικά διαχωρίσιμα. Σε ένα τέτοιο πρόβλημα τα δεδομένα μπορούν να χωριστούν σε κατηγορίες με μια ευθεία γραμμή. Για παράδειγμα, ο διαχωρισμός σημείων σε δύο ομάδες όταν βρίσκονται εκατέρωθεν μιας ευθείας αποτελεί καθαρά γραμμικά διαχωρίσιμο πρόβλημα [4]. Σε αυτές τις περιπτώσεις, το Perceptron μπορεί να βρει μια κατάλληλη γραμμή απόφασης και να ταξινομήσει τα δεδομένα σωστά.

Ωστόσο, όπως προαναφέρθηκε η δυνατότητα αυτή είναι και ο κύριος περιορισμός του. Η απουσία κρυφών επιπέδων (hidden layers) το καθιστά ανίκανο να αντιληφθεί πιο σύνθετες, μη γραμμικές συσχετίσεις. Συνεπώς, οποιοδήποτε πρόβλημα απαιτεί μη γραμμική σχέση μεταξύ εισόδου και εξόδου, αδυνατεί να επιλυθεί από ένα απλό δίκτυο [9].

Στις περισσότερες σύγχρονες εφαρμογές, όπως η αναγνώριση εικόνων, η επεξεργασία φυσικής γλώσσας και η ανάλυση ηχητικών σημάτων, οι σχέσεις μεταξύ των δεδομένων είναι ιδιαίτερα πολύπλοκες και μη γραμμικές [6]. Για τον λόγο αυτό, τα απλά νευρωνικά δίκτυα δεν επαρκούν και αντικαθίστανται από πολυεπίπεδες αρχιτεκτονικές.

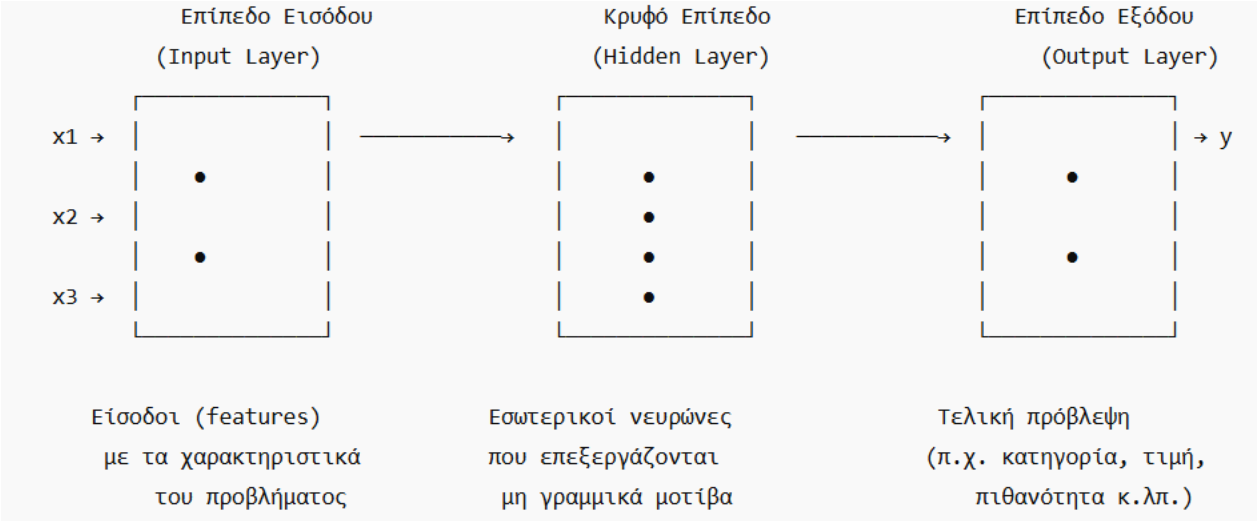
Παρά τους περιορισμούς αυτούς, τα απλά νευρωνικά δίκτυα αποτελούν το θεμέλιο λίθο για την ανάπτυξη πιο εξελιγμένων αρχιτεκτονικών. Η αναγνώριση των αδυναμιών τους οδήγησε στη δημιουργία των πολυεπίπεδων νευρωνικών δικτύων (Multilayer Perceptrons, MLP), τα οποία παρουσιάζονται στην επόμενη ενότητα.

1.3.5 Πολυεπίπεδα Νευρωνικά Δίκτυα (Multilayer Perceptrons , MLP).

Από τα απλά νευρωνικά δίκτυα οδηγούμαστε ως φυσική εξέλιξη στα Πολυεπίπεδα Νευρωνικά Δίκτυα (MLP). Η δημιουργία τους έρχεται ως συνέπεια στην πρόσθεση ενός ή περισσοτέρων κρυφών επιπέδων (hidden layers) ανάμεσα στο επίπεδο εισόδου και στο επίπεδο εξόδου. Κάθε κρυφό επίπεδο αποτελείται από νευρώνες που δέχονται τα αποτελέσματα του προηγούμενου επιπέδου, τα επεξεργάζονται μέσω μη γραμμικών συναρτήσεων ενεργοποίησης (όπως το ReLU ή το sigmoid) και τα προωθούν στο επόμενο επίπεδο. Έτσι, το δίκτυο μπορεί να απλοποιήσει μια δύσκολη σχέση σε πολλά απλούστερα βήματα, ενώ ταυτόχρονα αποκτά τη δυνατότητα να μαθαίνει πολύ πιο πολύπλοκες σχέσεις στα δεδομένα. Όσο περισσότερα επίπεδα διαθέτουν, τόσο πιο περίπλοκα μοτίβα μπορούν να εντοπίσουν χωρίς όμως αυτό να σημαίνει ότι περισσότερα επίπεδα οδηγούν πάντοτε σε καλύτερη απόδοση. Αυτός ο μηχανισμός επιτρέπει στα MLP να αντιμετωπίζουν προβλήματα που τα απλά δίκτυα δεν μπορούν [9].

Η διαδικασία κατά την οποία το δίκτυο υπολογίζει το σφάλμα της πρόβλεψης και το διαδίδει από το επίπεδο εξόδου προς τα προηγούμενα επίπεδα, ώστε να ενημερωθούν κατάλληλα τα βάρη, σηματοδοτεί την διαδικασία εκπαίδευσης του, που ονομάζεται backpropagation. Με αυτόν τον τρόπο, το MLP βελτιώνεται επαναληπτικά και μαθαίνει να παράγει πιο ακριβείς προβλέψεις [9].

Όταν ένα MLP περιλαμβάνει πολλά κρυφά επίπεδα, μετατρέπεται πρακτικά σε βαθύ δίκτυο, ικανό να λύσει σύνθετα προβλήματα όπως ανάλυση ήχου, επεξεργασία φυσικής γλώσσας και αναγνώριση εικόνας. Συνεπώς, τα πολυεπίπεδα νευρωνικά δίκτυα συνέβαλαν στην γέννηση του σύγχρονου Deep Learning και αποτελούν την βάση του [9].



ΣΧΗΜΑ 4: Διάγραμμα Πολυεπίπεδου Νευρωνικού Δικτύου (MLP).

Πηγή:Προσαρμογή από [9].

1.4 Deep Learning: Θεμελιώδεις Αρχές και Λειτουργία.

Η Βαθιά Μάθηση (Deep Learning) αποτελεί εξειδικευμένο κλάδο της Μηχανικής Μάθησης, ο οποίος βασίζεται στη χρήση πολυεπίπεδων τεχνητών νευρωνικών δικτύων για την εκμάθηση πολύπλοκων αναπαραστάσεων των δεδομένων [9].

Το Deep Learning έχει τη δυνατότητα να αναγνωρίζει μοτίβα με υψηλό βαθμό δυσκολίας, γεγονός που έρχεται σε αντίθεση με τα απλά νευρωνικά δίκτυα, τα οποία περιορίζονται σε απλές σχέσεις. Κατά τη διαδοχική επεξεργασία των δεδομένων, κάθε επίπεδο μαθαίνει να εξάγει βαθύτερα και πιο αφηρημένα χαρακτηριστικά, τα οποία τα επόμενα επίπεδα αξιοποιούν για να καταλήξουν σε ακριβέστερες προβλέψεις [9].

Η Gradient Descent καθώς και οι παραλλαγές της αποτελούν τους βασικούς αλγορίθμους βελτιστοποίησης που χρησιμοποιούνται για την εκπαίδευση τους. Το πεδίο του Deep Learning έχει γνωρίσει μεγάλη πρόοδο που οφείλεται κατά κύριο λόγο στην αύξηση της υπολογιστικής ισχύος, στη διαθεσιμότητα μεγάλων ποσοτήτων δεδομένων και στην ανάπτυξη νέων αρχιτεκτονικών (όπως τα Convolutional Neural Networks και τα Recurrent Neural Networks).

Ένα ακόμη πλεονέκτημα που καθιστά την Βαθιά Μάθηση ιδιαίτερα αποτελεσματική σε προβλήματα μεγάλης κλίμακας και υψηλής πολυπλοκότητας, όπου οι παραδοσιακές μέθοδοι αποτυγχάνουν είναι η αυτόματη εκμάθηση

κατάλληλων χαρακτηριστικών απευθείας από τα δεδομένα, μειώνοντας σημαντικά την ανάγκη χειροκίνητης εξαγωγής χαρακτηριστικών (feature engineering) [9].

Συνολικά, το Deep Learning έχει συμβάλει στην επιτυχή επίλυση πολύπλοκων προβλημάτων, με συνέπεια να εξελίσσει προοδευτικά την Τεχνητή Νοημοσύνη. Σήμερα αποτελεί το θεμέλιο πολλών σύγχρονων εφαρμογών, όπως η αναγνώριση εικόνας, η αυτόνομη οδήγηση, η μετάφραση φυσικής γλώσσας και τα συστήματα συστάσεων.

Κεφάλαιο 2: Γραμμική Άλγεβρα και Νευρωνικά Δίκτυα.

Σκοπός του Κεφαλαίου

Σκοπός του παρόντος κεφαλαίου είναι η κατανόηση της στενής σχέσης μεταξύ της Γραμμικής Άλγεβρας και των Τεχνητών Νευρωνικών Δικτύων. Για τον λόγο αυτό, παρουσιάζονται αναλυτικά τα βασικά μαθηματικά εργαλεία και έννοιες, όπως διανύσματα, πίνακες, γραμμικοί μετασχηματισμοί, συναρτήσεις ενεργοποίησης και συναρτήσεις σφάλματος, που χρησιμοποιούνται τόσο στη μοντελοποίηση όσο και εκπαίδευση των νευρωνικών δικτύων. Το κεφάλαιο αναδεικνύει τη σχέση αυτή ακολουθώντας τη ροή από τα δεδομένα εισόδου, στους γραμμικούς και μη γραμμικούς μετασχηματισμούς, τον υπολογισμό του σφάλματος κατά τη διαδικασία εκπαίδευσης, αναδεικνύοντας τον ρόλο της γραμμικής άλγεβρας σε κάθε στάδιο.

2.1 Διανύσματα Εισόδου και Εξόδου.

Στη γραμμική άλγεβρα, ένα διάνυσμα αποτελεί έναν οργανωμένο τρόπο για να περιγράψουμε, να επεξεργαστούμε και να αναλύσουμε πολλές πληροφορίες ταυτόχρονα με τη μορφή αριθμητικών τιμών, ως ένα ενιαίο αντικείμενο. Κάθε τιμή αντιστοιχεί σε μία συγκεκριμένη πληροφορία και όλες μαζί σχηματίζουν μια ενιαία μαθηματική αναπαράσταση. Τα διανύσματα μπορούν να γραφτούν ως γραμμές ή στήλες αριθμών [11].

Πολλά πραγματικά προβλήματα δεν περιγράφονται από έναν μόνο αριθμό, αλλά για να προσδιοριστούν με σαφήνεια χρειάζονται πολλές πληροφορίες ταυτόχρονα. Ορισμένα τέτοια προβλήματα για παράδειγμα θα μπορούσαν να αποτελούν τα ακόλουθα:

- Για να περιγράψουμε ένα σπίτι χρειαζόμαστε τα τετραγωνικά, τον αριθμό δωματίων, τον όροφο, την ηλικία $x = [120,3,2,15]$.

- Για να περιγράψουμε τη θέση και τη κατεύθυνση στον χώρο τόσο στη φυσική όσο και στη γεωμετρία. Όπως ένα σημείο στο επίπεδο χρειάζεται (x, y), στον χώρο χρειάζεται (x, y, z), πληροφορίες που πρέπει να είναι μαζί, γιατί όλες περιγράφουν το ίδιο αντικείμενο.
- Για να μπορούμε να κάνουμε μαθηματικές πράξεις πάνω σε πολλά δεδομένα ταυτόχρονα. Με τα διανύσματα μπορούμε να πολλαπλασιάζουμε πολλά δεδομένα μαζί, να εφαρμόζουμε τύπους σε ολόκληρα σύνολα δεδομένων [11].
- Για να απλοποιήσουμε τα μοντέλα, αντί να γράφουμε: $y = w_1x_1 + w_2x_2 + w_3x_3 + \cdots$ γράφουμε: $y = \mathrm{w}^T\mathrm{x}$ [9].
- Τέλος, για να μπορούν οι υπολογιστές να εργάζονται πιο αποδοτικά. Ιδιαίτερα τα νευρωνικά δίκτυα είναι σχεδιασμένα να επεξεργάζονται διανύσματα και πίνακες ώστε να κάνουν μαζικούς υπολογισμούς πολύ γρήγορα.

Στα νευρωνικά δίκτυα, η χρήση διανυσμάτων καθίσταται απαραίτητη, καθώς τα δεδομένα εισάγονται ως σύνολα τιμών που περιγράφουν διαφορετικά χαρακτηριστικά του ίδιου αντικειμένου και όχι ως μεμονωμένοι αριθμοί. Συνεπώς, κατά τη διαδικασία εισαγωγής των δεδομένων σε ένα νευρωνικό δίκτυο, αυτά οργανώνονται σε διανύσματα τα οποία ονομάζονται διανύσματα εισόδου και αποτελούν την τυπική μορφή με την οποία το δίκτυο λαμβάνει την πληροφορία [9].

Κάθε διάνυσμα εισόδου περιλαμβάνει όλες τις απαραίτητες πληροφορίες που χρειάζεται το δίκτυο για να επεξεργαστεί ένα συγκεκριμένο δείγμα. Για παράδειγμα, αν ένα αντικείμενο περιγράφεται από δύο ή τρία χαρακτηριστικά, τότε το διάνυσμα εισόδου περιλαμβάνει δύο ή τρεις αριθμητικές τιμές αντίστοιχα.

Με παρόμοιο τρόπο οργανώνεται και το αποτέλεσμα που παράγει το νευρωνικό δίκτυο. Εκφράζεται, εξίσου με τη μορφή διανύσματος και ονομάζεται διάνυσμα εξόδου. Ένα διάνυσμα εξόδου μπορεί να περιέχει μία ή περισσότερες τιμές, ανάλογα με το είδος του προβλήματος [9].

Συμπερασματικά, τα διανύσματα αποτελούν τον βασικό τρόπο με τον οποίο τα νευρωνικά δίκτυα δέχονται πληροφορία και παράγουν αποτελέσματα, ενώ παράλληλα μετατρέπουν πολυδιάστατα δεδομένα σε μαθηματικές αναπαραστάσεις κατάλληλες για υπολογιστική επεξεργασία και μάθηση.

2.2 Πίνακες Βαρών και Γραμμικοί Μετασχηματισμοί.

Στη γραμμική άλγεβρα, ένας πίνακας είναι ένα μαθηματικό εργαλείο που αποτελείται από αριθμητικά στοιχεία οργανωμένα σε γραμμές και στήλες. Κυρίως χρησιμοποιείται για την αναπαράσταση και την διαχείριση σχέσεων μεταξύ πολλών μεταβλητών ταυτόχρονα, ενώ παράλληλα αποτελούν βασικό εργαλείο για τον υπολογισμό γραμμικών μετασχηματισμών [9].

Εφόσον τα δεδομένα στα νευρωνικά δίκτυα παρουσιάζονται με τη μορφή των διανυσμάτων εισόδου, δημιουργείται η ανάγκη για μαθηματικούς μηχανισμούς που να επιτρέπουν τον συνδυασμό και τον μετασχηματισμό αυτών των διανυσμάτων. Οι πίνακες αναλαμβάνουν να καλύψουν τον ρόλο αυτό, καθώς εφαρμόζονται στα διανύσματα εισόδου και καθορίζουν τον τρόπο με τον οποίο η πληροφορία μετασχηματίζεται και μεταφέρεται στα επόμενα επίπεδα του δικτύου. Συνεπώς, τα διανύσματα και οι πίνακες έχουν διαφορετικό αλλά συμπληρωματικό ρόλο [9].

Με τη βοήθεια των πινάκων επιτρέπεται η εφαρμογή πράξεων πάνω σε διανύσματα. Ειδικότερα, ο πολλαπλασιασμός πίνακα με διάνυσμα οδηγεί σε ένα νέο διάνυσμα, το οποίο αποτελεί μετασχηματισμένη μορφή του αρχικού [11]. Στα νευρωνικά δίκτυα, η αρχή αυτή αξιοποιείται μέσω των πινάκων βαρών, οι οποίοι εφαρμόζονται στα διανύσματα εισόδου για την παραγωγή των ενδιάμεσων και τελικών εξόδων του δικτύου [9].

Η διαδικασία ακολουθεί την εξής πορεία, αρχικά ξεκινάμε από το διάνυσμα εισόδου: $\mathbf{x} = [x_1, x_2, \dots, x_n]^\top$ το οποίο και το πολλαπλασιάζουμε με τον πίνακα βαρών W: Wx. Κάθε γραμμή του πίνακα W αντιστοιχεί σε έναν νευρώνα του επόμενου επιπέδου, ενώ κάθε στήλη του πίνακα αντιστοιχεί σε ένα συγκεκριμένο χαρακτηριστικό της εισόδου [9].

Συνεπώς, η βασική μαθηματική πράξη που πραγματοποιείται σε κάθε επίπεδο του δικτύου είναι ο πολλαπλασιασμός του πίνακα βαρών W με το διάνυσμα εισόδου x. Η πράξη αυτή αποτελεί έναν γραμμικό μετασχηματισμό, μέσω του οποίου το διάνυσμα εισόδου μετατρέπεται σε ένα νέο διάνυσμα, το οποίο αντιπροσωπεύει τις προενεργοποιήσεις των νευρώνων του επόμενου επιπέδου [9].

Στη συνέχεια, προσθέτουμε μια σταθερά, το λεγόμενο bias, για να δώσουμε στον νευρώνα τη δυνατότητα να ενεργοποιηθεί ακόμα κι αν οι είσοδοι είναι μηδέν. Μετά την πρόσθεση του bias προκύπτει το $\mathrm{z} = \mathrm{Wx} + \mathrm{b}$ όπου παίρνουμε: $\mathrm{z} = [z_1, z_2, \dots, z_m]^\top$. Έτσι, το bias επιτρέπει στο δίκτυο να ξεκινήσει να λειτουργεί

σωστά και να μάθει αποτελεσματικά από τα δεδομένα, δηλαδή να ενεργοποιηθεί [9].

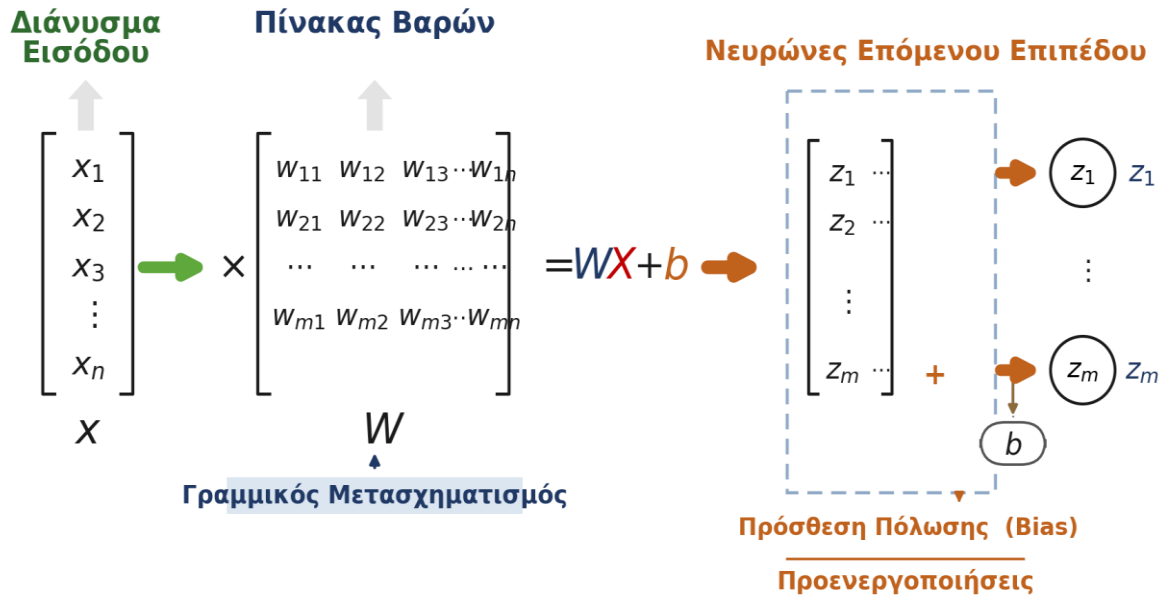
Κάθε στοιχείο z_i είναι η συνολική ενεργοποίηση ενός συγκεκριμένου νευρώνα του επόμενου επιπέδου [9]. Αν η τιμή είναι υψηλή, σημαίνει ότι η συγκεκριμένη είσοδος, μαζί με το βάρος που της αντιστοιχεί, έπαιξε σημαντικό ρόλο στο τελικό αποτέλεσμα για αυτόν τον νευρώνα.

Δηλαδή, αυτό το χαρακτηριστικό που καταχωρήθηκε ως είσοδος (όπως τα τετραγωνικά του σπιτιού, η τιμή κτλ.) θεωρήθηκε σημαντικό από το δίκτυο για τον συγκεκριμένο υπολογισμό. Αντίθετα, αν η τιμή είναι χαμηλή, τότε αυτό το χαρακτηριστικό δεν συνέβαλε και τόσο πολύ στην ενεργοποίηση του νευρώνα και άρα δεν το θεώρησε σημαντικό σε αυτό το στάδιο.

Συνεπώς αν η τιμή z_i είναι υψηλή, τότε μετά την ενεργοποίηση ο νευρώνας θα δώσει μια ισχυρή έξοδο. Αν είναι χαμηλή, μπορεί να μην ενεργοποιηθεί σχεδόν καθόλου [9].

Έτσι λοιπόν, όταν εμφανιστεί ένα υψηλό z_i, καταλαβαίνουμε ότι η είσοδος και το βάρος έδειξαν πως αυτό το χαρακτηριστικό είναι σημαντικό για τον συγκεκριμένο υπολογισμό που κάνει ο νευρώνας.

Κάθε στοιχείο του διανύσματος z αντιστοιχεί σε έναν νευρώνα, και το σύνολο των τιμών του z μας δείχνει πόσο έτοιμοι είναι όλοι οι νευρώνες αυτού του επιπέδου πριν ενεργοποιηθούν. Στη συνέχεια αυτές οι τιμές περνάνε από τις συναρτήσεις ενεργοποίησης και μετατρέπονται σε τελικές εξόδους που θα γίνουν οι είσοδοι για το επόμενο επίπεδο του δικτύου.



ΣΧΗΜΑ 5: Διαδικασία ενεργοποίησης ενός επιπέδου νευρωνικού δικτύου.

Πηγή:Ίδια επεξεργασία, βασισμένη στην πηγή [9].

Συνεπώς, στα νευρωνικά δίκτυα τα διανύσματα λειτουργούν ως φορείς πληροφορίας, ενώ οι πίνακες λειτουργούν ως μηχανισμοί επεξεργασίας της πληροφορίας. Ο γραμμικός μετασχηματισμός που προκύπτει από τη μεταξύ τους αλληλεπίδραση αποτελεί το θεμέλιο πάνω στο οποίο στηρίζεται η περαιτέρω επεξεργασία μέσω των συναρτήσεων ενεργοποίησης.

2.3 Συναρτήσεις Ενεργοποίησης και Μη Γραμμικοί Μετασχηματισμοί

Στα μαθηματικά, μια συνάρτηση ορίζεται ως ένας κανόνας που αντιστοιχίζει κάθε στοιχείο ενός συνόλου εισόδου σε ένα μοναδικό στοιχείο ενός συνόλου εξόδου. Με άλλα λόγια, μια συνάρτηση περιγράφει πώς μια τιμή μετασχηματίζεται σε μια άλλη μέσω ενός συγκεκριμένου κανόνα. Οι συναρτήσεις αποτελούν βασικό εργαλείο για τη μοντελοποίηση σχέσεων μεταξύ μεταβλητών [11].

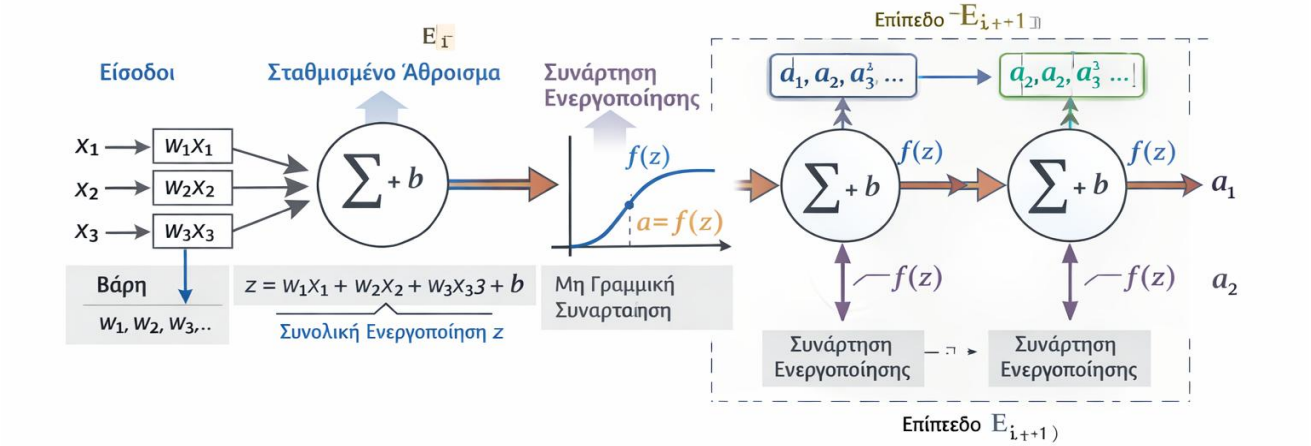
Στα νευρωνικά δίκτυα, μετά τον γραμμικό μετασχηματισμό, το αποτέλεσμα περνά από μια συνάρτηση ενεργοποίησης. Το διάνυσμα a που προκύπτει μετά την

εφαρμογή της συνάρτησης ενεργοποίησης αποτελεί την έξοδο του επιπέδου και χρησιμοποιείται ως είσοδος για το επόμενο επίπεδο του νευρωνικού δικτύου, όπου η ίδια διαδικασία επαναλαμβάνεται. Ο ρόλος της δεν είναι απλώς να παράγει μια αριθμητική τιμή, αλλά να καθορίζει αν και σε ποιον βαθμό ο νευρώνας θα ενεργοποιηθεί [9].

Αν η τιμή είναι πολύ χαμηλή, τότε όταν εφαρμόσουμε τη συνάρτηση ενεργοποίησης, ο νευρώνας μπορεί ουσιαστικά να μην ενεργοποιηθεί καθόλου ή να δώσει μια πολύ ασθενή έξοδο. Αυτό σημαίνει ότι πρακτικά η πληροφορία από αυτή την είσοδο δεν θα περάσει παρακάτω στο δίκτυο με κάποιο σημαντικό τρόπο.

Η βασική σημασία της συνάρτησης ενεργοποίησης ωστόσο έγκειται στο γεγονός ότι εισάγει μη γραμμικότητα στο μοντέλο. Αν το νευρωνικό δίκτυο περιείχε μόνο γραμμικούς μετασχηματισμούς, τότε ακόμη και με πολλά επίπεδα θα ισοδυναμούσε με έναν απλό γραμμικό μετασχηματισμό και δεν θα μπορούσε να μοντελοποιήσει σύνθετες σχέσεις στα δεδομένα [9].

Οι μη γραμμικοί μετασχηματισμοί που προκύπτουν από τις συναρτήσεις ενεργοποίησης επιτρέπουν στο νευρωνικό δίκτυο να προσεγγίζει πολύπλοκες και μη γραμμικές συναρτήσεις. Με τον τρόπο αυτό, το δίκτυο αποκτά τη δυνατότητα να μαθαίνει σύνθετα μοτίβα και να εφαρμόζεται σε πραγματικά προβλήματα, όπου οι σχέσεις μεταξύ των δεδομένων δεν είναι απλές ή γραμμικές [9].



ΣΧΗΜΑ 6: Εφαρμογή της συνάρτησης ενεργοποίησης στο διάνυσμα προενεργοποιήσεων.

Πηγή:Ίδια επεξεργασία, βασισμένη στην πηγή [9].

2.4 Συνάρτηση Σφάλματος.

Η συνάρτηση σφάλματος λειτουργεί στα νευρωνικά δίκτυα ως ένα βασικό μαθηματικό εργαλείο με το οποίο γίνεται ποσοτικοποίηση της απόκλισης μεταξύ της εξόδου που παράγει το δίκτυο και της επιθυμητής ή πραγματικής τιμής. Πιο αναλυτικά, η συνάρτηση σφάλματος αποτυπώνει αριθμητικά το πόσο καλή ή κακή είναι η πρόβλεψη του μοντέλου για ένα δεδομένο δείγμα [12].

Ως προς την μαθηματική της έκφραση η συνάρτηση σφάλματος ορίζεται ως μια συνάρτηση που δέχεται ως είσοδο την προβλεπόμενη έξοδο του δικτύου και την πραγματική τιμή και επιστρέφει ως αποτέλεσμα μια μη αρνητική πραγματική τιμή. Η μη αρνητική τιμή που προκύπτει εκφράζει το μέγεθος του σφάλματος και εξαρτάται από το είδος του προβλήματος [12].

Στα προβλήματα παλινδρόμησης, η συνάρτηση σφάλματος που χρησιμοποιείται κατά κύριο λόγο είναι το Μέσο Τετραγωνικό Σφάλμα, το οποίο δίνεται από τη σχέση:

$$L(y, \hat{y}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

όπου y_i είναι η πραγματική τιμή και $\hat{y}_i$ η προβλεπόμενη τιμή για το i-οστό δείγμα [12]. Όλες οι συναρτήσεις σφάλματος παρουσιάζουν ένα κοινό χαρακτηριστικό, ότι επιτρέπουν τη σύγκριση μεταξύ διαφορετικών μοντέλων και ρυθμίσεων, παρέχοντας ένα ενιαίο κριτήριο αξιολόγησης της απόδοσής τους. Η συνάρτηση σφάλματος συνδέεται άμεσα με τη διαδικασία εκπαίδευσης και τη βελτιστοποίηση των παραμέτρων του νευρωνικού δικτύου [9].

Σημειώνεται ότι η συνάρτηση L(y,ŷ) όπως ορίστηκε παραπάνω, υπολογισμένη ως μέσος όρος πάνω σε όλο το σύνολο δεδομένων, αντιστοιχεί εννοιολογικά στη συνάρτηση κόστους J(θ) που ορίζεται αναλυτικά στο Κεφάλαιο 3, ενώ το επιμέρους σφάλμα για ένα μόνο δείγμα αντιστοιχεί στη συνάρτηση σφάλματος (loss function) ℓ(·). Στην παρούσα εργασία, οι όροι «συνάρτηση σφάλματος» και «συνάρτηση κόστους» χρησιμοποιούνται με αυτή την αντιστοιχία σε όλα τα κεφάλαια, ώστε το L του παρόντος κεφαλαίου να ταυτίζεται με το J(θ) του Κεφαλαίου 3.

2.5 Προώθηση Πληροφορίας.

Η προώθηση πληροφορίας (forward propagation) περιγράφει τη μαθηματική διαδικασία μέσω της οποίας ένα νευρωνικό δίκτυο μετατρέπει το διάνυσμα εισόδου σε έξοδο, μέσω διαδοχικών γραμμικών και μη γραμμικών μετασχηματισμών. Κατά τη διαδικασία αυτή, η πληροφορία κινείται μόνο προς μία κατεύθυνση, από το επίπεδο εισόδου προς τα κρυφά επίπεδα και τελικά προς το επίπεδο εξόδου, χωρίς να επιστρέφει σε προηγούμενο επίπεδο και χωρίς να πραγματοποιείται καμία προσαρμογή των παραμέτρων του δικτύου [9].

Η περιγραφή της διαδικασίας που ακολουθείται δίνεται ως εξής : Έστω ένα πολυεπίπεδο νευρωνικό δίκτυο αποτελούμενο από L επίπεδα. Το διάνυσμα εισόδου του δικτύου συμβολίζεται ως $a^{(0)}$και αντιστοιχεί στα χαρακτηριστικά που περιγράφουν ένα δεδομένο δείγμα. Για κάθε επίπεδο $l = 1,2, \dots, L$, η λειτουργία του δικτύου περιγράφεται από δύο διαδοχικά στάδια: τον γραμμικό μετασχηματισμό και την εφαρμογή της συνάρτησης ενεργοποίησης [9].

Στο πρώτο στάδιο, υπολογίζεται το διάνυσμα προενεργοποιήσεων $\mathrm{z}^{(l)}$, το οποίο προκύπτει από τον γραμμικό μετασχηματισμό:

$$\mathrm{z}^{(l)} = \mathrm{W}^{(l)}\mathrm{a}^{(l-1)} + \mathrm{b}^{(l)}$$

όπου $\mathrm{W}^{(l)}$είναι ο πίνακας βαρών του επιπέδου l, $\mathrm{b}^{(l)}$το διάνυσμα πόλωσης (bias) και $\mathrm{a}^{(l-1)}$το διάνυσμα ενεργοποιήσεων του προηγούμενου επιπέδου. Κάθε στοιχείο του διανύσματος $\mathrm{z}^{(l)}$αντιστοιχεί στη συνολική (καθαρή) ενεργοποίηση ενός νευρώνα του επιπέδου πριν την εφαρμογή μη γραμμικότητας [9].

Στο δεύτερο στάδιο, εφαρμόζεται η συνάρτηση ενεργοποίησης $f(\cdot)$ στο διάνυσμα προενεργοποιήσεων, παράγοντας το διάνυσμα ενεργοποιήσεων:

$$\mathrm{a}^{(l)} = f\left(\mathrm{z}^{(l)}\right)$$

Το διάνυσμα $\mathrm{a}^{(l)}$αποτελεί την έξοδο του επιπέδου l και χρησιμοποιείται ως είσοδος για το επόμενο επίπεδο του δικτύου. Κάθε συνιστώσα $a_i^{(l)}$εκφράζει τον βαθμό ενεργοποίησης του αντίστοιχου νευρώνα και καθορίζει τη συμβολή του στη συνέχεια της επεξεργασίας [9].

Η παραπάνω διαδικασία επαναλαμβάνεται διαδοχικά για όλα τα επίπεδα του δικτύου. Στο τελευταίο επίπεδο, το διάνυσμα ενεργοποιήσεων $\mathrm{a}^{(L)}$αποτελεί την τελική έξοδο του νευρωνικού δικτύου. Η μορφή και το νόημα της εξόδου αυτής εξαρτώνται από το είδος του προβλήματος που εξετάζεται, όπως παλινδρόμηση ή ταξινόμηση, καθώς και από τη συνάρτηση ενεργοποίησης που χρησιμοποιείται στο επίπεδο εξόδου [9].

Συνεπώς, η προώθηση πληροφορίας ως διαδικασία αποτελεί τη βάση για την κατανόηση της λειτουργίας του νευρωνικού δικτύου και προηγείται της εκπαίδευσης και της βελτιστοποίησης των παραμέτρων του.

Κεφάλαιο 3: Εκπαίδευση με Gradient Descent.

Σκοπός του Κεφαλαίου.

Σκοπός του παρόντος κεφαλαίου είναι η ανάδειξη του ρόλου της μεθόδου Καθοδικής Κλίσης (Gradient Descent) κατά τη διαδικασία της εκπαίδευσης Νευρωνικών Δικτύων. Ειδικότερα, το κεφάλαιο περιέχει τη διατύπωση του

προβλήματος βελτιστοποίησης, τη βασική ιδέα και τη μαθηματική διατύπωση της μεθόδου Gradient Descent. Επίσης, γίνεται ανάλυση του ρυθμού μάθησης και των συνθηκών σύγκλισης, καθώς παρουσίαση των κύριων παραλλαγών της μεθόδου (Batch, Stochastic και Mini-batch Gradient Descent). Τέλος, εξετάζεται η σύνδεση της μεθόδου με τον αλγόριθμο οπισθοδιάδοσης (Backpropagation) του σφάλματος και αναφέρονται συνοπτικά βασικά ζητήματα και περιορισμοί που προκύπτουν κατά τη διαδικασία της εκπαίδευσης.

3.1 Πρόβλημα Βελτιστοποίησης στην Εκπαίδευση Νευρωνικών Δικτύων.

Ένα Νευρωνικό Δίκτυο είναι στην ουσία ένα παραμετρικό υπολογιστικό μοντέλο, το οποίο μετασχηματίζει τα δεδομένα εισόδου σε εξόδους μέσω μιας ακολουθίας γραμμικών συνδυασμών και μη γραμμικών συναρτήσεων ενεργοποίησης. Τα βάρη και τα bias, αποτελούν τις παραμέτρους του και καθορίζουν αποκλειστικά τη συμπεριφορά του δικτύου [9]. Οι παράμετροι αυτοί ρυθμίζονται κατά τη διαδικασία της εκπαίδευσης [13].

Τόσο τα βάρη που εκφράζουν τη σχετική σημασία των εισόδων κάθε νευρώνα, όσο και τα bias που επιτρέπουν τη μετατόπιση της ενεργοποίησης, καθιστούν το μοντέλο ικανό να προσαρμόζεται σε δεδομένα που δεν διέρχονται αναγκαστικά από την αρχή των αξόνων. Η επιλογή των αρχικών τιμών των παραμέτρων αυτών γίνεται συνήθως τυχαία και, κατά συνέπεια, το δίκτυο στην αρχική του κατάσταση δεν αναπαριστά ικανοποιητικά τη σχέση μεταξύ εισόδων και εξόδων [9].

Η εκπαίδευση ενός Νευρωνικού Δικτύου έχει ως στόχο τη σταδιακή προσαρμογή των βαρών και των bias, προκειμένου η έξοδος του δικτύου να προσεγγίζει όσο το δυνατόν καλύτερα τις προσδοκόμενες τιμές για ένα δεδομένο σύνολο εκπαίδευσης.

Η απόκλιση μεταξύ της προβλεπόμενης και της πραγματικής εξόδου ποσοτικοποιείται μέσω μιας loss function $\ell(\cdot)$. Με τον όρο loss function $\ell(\cdot)$ εννοούμε τη μέτρηση του σφάλματος σε επίπεδο μεμονωμένου δείγματος, ενώ με τον όρο cost function J(θ) γίνεται αναφορά στη συνολική συνάρτηση κόστους, η οποία προκύπτει ως ο μέσος όρος των loss functions στο σύνολο εκπαίδευσης και ελαχιστοποιείται κατά τη διαδικασία της εκπαίδευσης. Πιο συγκεκριμένα, η cost function μετρά πόσο μη λειτουργικό είναι το δίκτυο για συγκεκριμένες τιμές παραμέτρων θ. Όσο μικρότερο το $J(\theta)$ τόσο καλύτερο μοντέλο [9], [12].

Έστω σύνολο εκπαίδευσης

$$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N},$$

όπου x_i είναι τα δεδομένα εισόδου και y_i οι αντίστοιχες επιθυμητές έξοδοι. Το Νευρωνικό Δίκτυο ορίζεται ως παραμετρική συνάρτηση $f(x; \theta)$, με θ να περιλαμβάνει όλα τα βάρη και τα bias του δικτύου. Η συνολική συνάρτηση κόστους ορίζεται ως ο μέσος όρος των απωλειών στο σύνολο εκπαίδευσης [12]:

$$J(\theta) = \frac{1}{N} \sum_{i=1}^{N} \ell(f(x_i; \theta), y_i).$$

Το πρόβλημα της εκπαίδευσης ανάγεται, συνεπώς, στην επίλυση ενός προβλήματος βελτιστοποίησης. Γενικά, ως πρόβλημα βελτιστοποίησης ορίζεται η διαδικασία προσδιορισμού των τιμών ενός ή περισσότερων μεταβλητών απόφασης, έτσι ώστε να ελαχιστοποιείται ή να μεγιστοποιείται μια πραγματική συνάρτηση, η οποία ονομάζεται αντικειμενική συνάρτηση (objective function) [9].

Στο πλαίσιο της εκπαίδευσης Νευρωνικών Δικτύων, οι μεταβλητές απόφασης είναι οι παράμετροι του μοντέλου (βάρη και bias), ενώ η αντικειμενική συνάρτηση ταυτίζεται με τη συνάρτηση κόστους $J(\theta)$, η οποία ποσοτικοποιεί το συνολικό σφάλμα του δικτύου στο σύνολο εκπαίδευσης. Το αντίστοιχο πρόβλημα βελτιστοποίησης διατυπώνεται ως:

$$\theta^* = \arg\min_{\theta} J(\theta),$$

στο οποίο ζητείται ο προσδιορισμός των παραμέτρων θ^*που ελαχιστοποιούν τη συνάρτηση κόστους [9], [12].

Το $\min J(\theta)$ παρουσιάζει ποια είναι η ελάχιστη τιμή της συνάρτησης κόστους, χωρίς όμως να μας φανερώνει ποια θτην πέτυχε. Για τον λόγο αυτό γίνεται χρήση του arg min (argument of the minimum) με το όποιο γίνεται ορατό πλέον ποια τιμή του θ κάνει τη συνάρτηση $J(\theta)$ ελάχιστη. Δηλαδή, το min επιστρέφει αριθμό, ενώ το arg min επιστρέφει τις παραμέτρους.

Το συγκεκριμένο πρόβλημα είναι γενικά μη γραμμικό και έχει υψηλή διάσταση, καθώς ο αριθμός των παραμέτρων αυξάνεται με το πλήθος των επιπέδων και των νευρώνων του δικτύου. Αυτό έχει ως αποτέλεσμα, η αναλυτική εύρεση της βέλτιστης λύσης να είναι ανέφικτη και να καθίσταται απαραίτητη η χρήση επαναληπτικών μεθόδων βελτιστοποίησης, όπως η μέθοδος Καθοδικής Κλίσης (Gradient Descent).

3.2: Η βασική ιδέα της Gradient Descent και Μαθηματική διατύπωση.

Η μέθοδος Καθοδικής Κλίσης (Gradient Descent) αποτελεί έναν επαναληπτικό αλγόριθμο βελτιστοποίησης, ο οποίος χρησιμοποιείται για την ελαχιστοποίηση μιας πραγματικής συνάρτησης πολλών μεταβλητών [9].

Κατά τη διαδικασία εκπαίδευσης ενός Νευρωνικού Δικτύου, ο στόχος είναι η προσαρμογή των παραμέτρων του μοντέλου, έτσι ώστε η έξοδος του δικτύου να προσεγγίζει όσο το δυνατόν καλύτερα τις επιθυμητές τιμές. Συνεπώς, η Gradient Descent εφαρμόζεται για την ελαχιστοποίηση της συνάρτησης κόστους $J(\theta)$ ως προς το διάνυσμα παραμέτρων θ, το οποίο περιλαμβάνει όλα τα βάρη και τα bias του δικτύου. Κάθε διαφορετικός συνδυασμός τιμών των παραμέτρων αυτών αντιστοιχεί σε μια διαφορετική συμπεριφορά του δικτύου και, κατ' επέκταση, σε μια διαφορετική τιμή της συνάρτησης κόστους $J(\theta)$, η οποία ποσοτικοποιεί το συνολικό σφάλμα του μοντέλου [9], [13].

Οι παράμετροι του δικτύου μπορούν να θεωρηθούν ως συνιστώσες ενός διανύσματος

$$\theta = (w_1, w_2, \dots, w_n, b_1, b_2, \dots),$$

όπου κάθε παράμετρος αντιστοιχεί σε έναν άξονα ενός αφηρημένου, πολυδιάστατου χώρου, γνωστού ως χώρου παραμέτρων. Αν για παράδειγμα, έχουμε ένα βάρος τότε προκύπτει ένας άξονας και κάθε σημείο πάνω στη γραμμή ισούται με μία τιμή του βάρους. Αν έχουμε δύο βάρη w_1, w_2 τότε προκύπτουν δύο άξονες ένας ο οριζόντιος για το w_1 και ένας ο κάθετος για το w_2.Κάθε σημείο στο επίπεδο (w_1, w_2) είναι ένας συγκεκριμένος συνδυασμός βαρών. Με παρόμοιο τρόπο αν έχω τρία βάρη w_1, w_2, w_3 τότε υπάρχουν τρεις άξονες και ο χώρος γίνεται τρισδιάστατος με κάθε σημείο (w_1, w_2, w_3) να είναι ένα σημείο στο 3D χώρο. Αν

όμως, έχουμε 4 ή περισσότερα βάρη επειδή κάθε βάρος ισούται με έναν άξονα δημιουργείται ο χώρος υψηλής διάστασης (high-dimensional space) [9].

Ένα σημείο σε αυτόν τον χώρο αντιπροσωπεύει έναν πλήρη και συγκεκριμένο ορισμό του Νευρωνικού Δικτύου. Αν και ο χώρος αυτός δεν είναι δυνατόν να απεικονιστεί γραφικά όταν ο αριθμός των παραμέτρων είναι μεγάλος, η μαθηματική του ύπαρξη επιτρέπει την ανάλυση και τη βελτιστοποίηση της συμπεριφοράς του μοντέλου [9].

Η παράγωγος μιας συνάρτησης μιας μεταβλητής εκφράζει τον ρυθμό μεταβολής της ως προς τη μεταβλητή, ενώ στις συναρτήσεις πολλών μεταβλητών χρησιμοποιούνται οι μερικές παράγωγοι, οι οποίες δείχνουν πώς μεταβάλλεται η συνάρτηση ως προς μία μεταβλητή, όταν όλες οι υπόλοιπες παραμένουν σταθερές. Η Gradient Descent σε πολλούς άξονες δεν μελετά τον χώρο**,** απλώς υπολογίζει τις μερικές παραγώγους $\frac{\partial J}{\partial w_1}$, $\frac{\partial J}{\partial w_2}$, ... *ώστε* για κάθε άξονα (βάρος) να ενημερώνεται για το πόσο πρέπει να κινηθεί [12].

Η μερική παράγωγος $\frac{\partial J}{\partial w}$ ανάλογα την τιμή της παρουσιάζει τι θα γίνει αν αυξηθεί λίγο το βάρος w, δηλαδή αν το κόστος J αυξάνεται ή μειώνεται και πόσο.

Περίπτωση 1**:** Αν $\frac{\partial J}{\partial w} > 0$ σημαίνει ότι αν αυξηθεί το w, το κόστος αυξάνεται. Άρα για να μειωθεί το κόστος, πρέπει να μειωθεί το w και να κινηθεί αντίθετα από αυτό που δείχνει η παράγωγος.

Περίπτωση 2: Αν $\frac{\partial J}{\partial w} < 0$ σημαίνει ότι αν αυξηθεί το w, το κόστος μειώνεται. Άρα πρέπει να αυξήσεις το w και να κινηθεί αντίθετα από την κατεύθυνση αύξησης του κόστους.

Αυτό συνοψίζεται στον κανόνα:

$$w_{\text{new}} = w_{\text{old}} - \eta \frac{\partial J}{\partial w}$$
$$b_{\text{new}} = b_{\text{old}} - \eta \frac{\partial J}{\partial b}$$

όπου:

- $\eta > 0$= ρυθμός μάθησης (learning rate)
- το $-$ εξασφαλίζει ότι κινούμαστε προς την αντίθετη κατεύθυνση και βοηθάει τον αλγόριθμο να καταλάβει ότι αν μια παράμετρος αυξάνει το κόστος, πρέπει να μειωθεί, ενώ αν μια παράμετρος το μειώνει, να αυξηθεί.

Αυτό γίνεται για κάθε βάρος και κάθε bias ξεχωριστά.

Ο συνδυασμός όλων των μερικών παραγώγων της συνάρτησης κόστους ως προς τις παραμέτρους του μοντέλου σχηματίζει το διάνυσμα κλίσης (gradient), το οποίο αποτελεί στην ουσία ένα βέλος που υποδεικνύει κατεύθυνση μεταβολής στον χώρο των παραμέτρων και συμβολίζεται ως $\nabla J(\theta)$ [9].

Για την περίπτωση πολυδιάστατης συνάρτησης κόστους $J(\theta)$, όπου

$\theta = (\theta_1, \theta_2, \dots, \theta_n)$,

το gradient ορίζεται ως:

$$\nabla J(\theta) = \left[\frac{\partial J}{\partial \theta_1}, \frac{\partial J}{\partial \theta_2}, \dots, \frac{\partial J}{\partial \theta_n}\right]^T$$

όπου κάθε θ_i αντιστοιχεί σε μία παράμετρο του δικτύου.

Η Gradient Descent ενημερώνει τις παραμέτρους του μοντέλου μετακινώντας το διάνυσμα θ προς την αντίθετη κατεύθυνση του gradient, σύμφωνα με τον γενικό κανόνα:

$$\theta_{new} = \theta_{old} - \eta \frac{\partial J}{\partial \theta_{old}},$$

για $\theta \in \{\theta_1, \theta_2, \dots, \theta_n\}$,

όπου $\eta > 0$ και kο δείκτης επανάληψης. Ο ρυθμός μάθησης η καθορίζει το μέγεθος του βήματος σε κάθε επανάληψη της διαδικασίας.

Το gradient υποδεικνύει την κατεύθυνση της μέγιστης τοπικής αύξησης της συνάρτησης κόστους στον χώρο των παραμέτρων και δείχνει προς ποια αλλαγή των βαρών και των bias το σφάλμα του μοντέλου αυξάνεται ταχύτερα. Συνεπώς,

για τη μείωση της τιμής της συνάρτησης κόστους, οι παράμετροι του δικτύου πρέπει να μεταβάλλονται προς την αντίθετη κατεύθυνση από αυτήν που υποδεικνύει το gradient [9].

Η μέθοδος Καθοδικής Κλίσης (Gradient Descent) βασίζεται ακριβώς σε αυτήν την αρχή. Σε κάθε επαναληπτικό βήμα, οι παράμετροι του μοντέλου ενημερώνονται οδηγώντας το Νευρωνικό Δίκτυο σε διαδοχικές καταστάσεις με ολοένα και μικρότερο σφάλμα. Η επαναλαμβανόμενη αυτή διαδικασία επιτρέπει την προσεγγιστική εύρεση ενός σημείου ελαχίστου της συνάρτησης κόστους και, κατά συνέπεια, την αποτελεσματική εκπαίδευση του μοντέλου [9], [13].

3.3: Learning rate και σύγκλιση.

Η συμπεριφορά και η αποτελεσματικότητα της μεθόδου Gradient Descent εξαρτώνται σε μεγάλο βαθμό από την επιλογή του ρυθμού μάθησης (learning rate), καθώς και από τα κριτήρια με τα οποία καθορίζεται η σύγκλιση της διαδικασίας εκπαίδευσης. Με τον όρο συμπεριφορά της Gradient Descent, εννοούμε πώς κινείται ο αλγόριθμος κατά την εκπαίδευση. Η συμπεριφορά περιγράφει τι κάνει ο αλγόριθμος στην πράξη, για παράδειγμα:

- μειώνεται το κόστος ομαλά ή με σκαμπανεβάσματα
- πλησιάζει σταθερά το ελάχιστο ή ταλαντώνεται
- συγκλίνει γρήγορα ή πολύ αργά
- σταθεροποιείται ή αποκλίνει

Ενώ, με τον όρο αποτελεσματικότητα αναφερόμαστε στο πόσο καλά εκπαιδεύεται το μοντέλο, πόσο χρόνο χρειάζεται και πόσες επαναλήψεις κατά την εκπαίδευση του [13]. Το learning rate η επηρεάζει άμεσα και τα δύο.

Μικρό η	Μεγάλο η
○ πολύ μικρά βήματα	○ μεγάλα βήματα
○ αργή μείωση κόστους	○ ταλαντώσεις γύρω από το ελάχιστο
○ σταθερή αλλά αναποτελεσματική εκπαίδευση	○ πιθανή απόκλιση

Πίνακας 1: Επιρροή του ρυθμού μάθησης.

Πηγή:Ίδια επεξεργασία, βασισμένη στις πηγές [9] και [11].

Αν και ο κανόνας ενημέρωσης των παραμέτρων είναι απλός στη μορφή του, η πρακτική εφαρμογή του επηρεάζεται άμεσα από το μέγεθος του βήματος που πραγματοποιείται σε κάθε επανάληψη [9], [13].

Ουσιαστικά, το η ελέγχει πόσο μεγάλο βήμα γίνεται στον χώρο των παραμέτρων προς την κατεύθυνση μείωσης της συνάρτησης κόστους. Για τον λόγο αυτό, το learning rate αποτελεί υπερπαράμετρο της διαδικασίας εκπαίδευσης, η τιμή της οποίας δεν προκύπτει από τα δεδομένα αλλά επιλέγεται από τον χρήστη. Η αρχική του τιμή επιλέγεται εμπειρικά, καθώς δεν προσδιορίζεται αυτόματα από τα δεδομένα [13].

Η επιλογή της τιμής του ρυθμού μάθησης επηρεάζει άμεσα τη σύγκλιση της μεθόδου. Όταν το η είναι πολύ μικρό, οδηγεί σε βραδεία σύγκλιση και αυξημένο υπολογιστικό κόστος. Στο πλαίσιο της εκπαίδευσης Νευρωνικών Δικτύων, το υπολογιστικό κόστος αφορά κυρίως τον χρόνο εκτέλεσης (CPU/GPU time), την χρήση μνήμης (RAM / GPU memory) και την κατανάλωση ενέργειας (σε μεγάλες εκπαιδεύσεις) [13].

Αντίθετα, όταν το η είναι πολύ μεγάλο προκαλεί ταλαντώσεις ή ακόμη και απόκλιση της μεθόδου.

Η έννοια της σύγκλισης αναφέρεται στη συμπεριφορά της ακολουθίας των παραμέτρων $\{\theta_k\}$ καθώς ο αριθμός των επαναλήψεων αυξάνεται. Η Gradient Descent θεωρείται ότι συγκλίνει όταν οι παράμετροι του μοντέλου ή η τιμή της συνάρτησης κόστους σταθεροποιούνται και δεν παρουσιάζουν πλέον σημαντικές μεταβολές [9].

Σημαντικό είναι να σημειωθεί ότι η σύγκλιση μπορεί να εξεταστεί τόσο ως προς τις τιμές των παραμέτρων όσο και ως προς την τιμή της συνάρτησης κόστους. Για παράδειγμα, υπάρχει σύγκλιση ως προς τις παραμέτρους όταν τα βάρη και τα bias σχεδόν δεν αλλάζουν πια από βήμα σε βήμα.

Δηλαδή:

- στην επανάληψη k:

$$w_k = 1.234$$

- στην επανάληψη $k+1$:

$$w_{k+1} = 1.234001$$

Η αλλαγή είναι αμελητέα. Άρα οι παράμετροι έχουν συγκλίνει.

Ενώ για παράδειγμα, αναφερόμαστε ότι υπάρχει σύγκλιση ως προς το κόστος όταν η τιμή της συνάρτησης κόστους σχεδόν δεν μειώνεται πια.

Δηλαδή:

- στην επανάληψη k:
$$J_k = 0.1532$$
- στην επανάληψη $k + 1$:
$$J_{k+1} = 0.1531$$

Η βελτίωση είναι ελάχιστη. Άρα το κόστος έχει συγκλίνει.

Σε ορισμένες περιπτώσεις, οι παράμετροι ενδέχεται να συνεχίζουν να μεταβάλλονται ελαφρώς, ενώ η συνάρτηση κόστους παραμένει ουσιαστικά αμετάβλητη. Για τον λόγο αυτό, χρησιμοποιούνται συγκεκριμένα κριτήρια τερματισμού της εκπαίδευσης [13].

Συνήθη κριτήρια τερματισμού της Gradient Descent περιλαμβάνουν τον καθορισμό ενός μέγιστου αριθμού επαναλήψεων ή εποχών (epochs), τη διακοπή της διαδικασίας όταν η μεταβολή της συνάρτησης κόστους μεταξύ διαδοχικών επαναλήψεων γίνει μικρότερη από ένα προκαθορισμένο κατώφλι όπως όταν το $|J_{k+1} - J_k|$<ε, καθώς και την περίπτωση όπου το μέτρο του gradient προσεγγίζει το μηδέν ‖∇J(θ)‖≈0. Τα κριτήρια αυτά επιτρέπουν την πρακτική εφαρμογή της μεθόδου χωρίς την απαίτηση πλήρους μαθηματικής σύγκλισης. Τους κανόνες αυτούς τους χρησιμοποιούμε για να αποφασίσουμε πότε σταματά η μέθοδος, επειδή το μοντέλο δεν βελτιώνεται πλέον ουσιαστικά [9], [13].

Συνοψίζοντας, ο ρυθμός μάθησης και τα κριτήρια σύγκλισης συντελούν σε μεγάλο βαθμό στην αποτελεσματική εφαρμογή της μεθόδου. Η κατάλληλη επιλογή τους εξασφαλίζει σταθερή και αποδοτική εκπαίδευση του Νευρωνικού Δικτύου, ενώ ταυτόχρονα επηρεάζει άμεσα τόσο την ταχύτητα σύγκλισης όσο και την τελική ποιότητα του εκπαιδευμένου μοντέλου.

3.4: Batch vs Stochastic vs Mini-batch Gradient Descent.

Η μέθοδος της Gradient Descent έχει οριστεί ως μια μορφή εκπαίδευσης κατά την οποία η εκπαίδευση του μοντέλου πραγματοποιείται μέσω της ελαχιστοποίησης της συνάρτησης κόστους ως προς τις παραμέτρους του.

Στην πράξη, το Νευρωνικό Δίκτυο εκτελείται για όλα τα δείγματα του συνόλου εκπαίδευσης, υπολογίζεται η συνάρτηση κόστους για κάθε δείγμα και στη συνέχεια το gradient προκύπτει ως ο μέσος όρος των μερικών παραγώγων της συνάρτησης κόστους ως προς τις παραμέτρους του μοντέλου, για κάθε επαναληπτικό βήμα της διαδικασίας εκπαίδευσης. Όμως η ενημέρωση των βαρών και των bias πραγματοποιείται μόνο αφού ολοκληρωθεί ο υπολογισμός αυτός [9].

Η προσέγγιση αυτή οδηγεί σε σταθερή και ομαλή πορεία σύγκλισης, καθώς το υπολογιζόμενο gradient αποτελεί ακριβή εκτίμηση της κατεύθυνσης μείωσης της συνάρτησης κόστους. Παρόλα αυτά, όταν το πλήθος των δεδομένων είναι μεγάλο, η ανάγκη επεξεργασίας ολόκληρου του συνόλου εκπαίδευσης σε κάθε επανάληψη συνεπάγεται αυξημένο υπολογιστικό κόστος, γεγονός που καθιστά τη βασική αυτή μορφή της Gradient Descent λιγότερο πρακτική για μεγάλα προβλήματα [14].

Για τον λόγο αυτό έχουν προταθεί διαφορετικές παραλλαγές της μεθόδου. Η κλασσική Gradient Descent αντιστοιχεί στην Batch Gradient Descent, ενώ οι Stochastic και Mini-batch Gradient Descent αποτελούν παραλλαγές της. Οι διαφορές μεταξύ των μεθόδων Batch, Stochastic και Mini-batch Gradient Descent εντοπίζονται κατά κύριο λόγο στο πόσα δείγματα χρησιμοποιούνται για τον υπολογισμό του gradient σε κάθε επαναληπτικό βήμα [14].

<u>Batch Gradient Descent</u>

Στην περίπτωση της Batch Gradient Descent, το gradient της συνάρτησης κόστους υπολογίζεται χρησιμοποιώντας ολόκληρο το σύνολο εκπαίδευσης:

$$\nabla J(\theta) = \frac{1}{N}\sum_{i=1}^{N} \nabla_\theta \, \ell(f(x_i;\theta), y_i).$$

Ο κανόνας ενημέρωσης των παραμέτρων εφαρμόζεται αφού ολοκληρωθεί ο υπολογισμός του gradient για όλα τα δείγματα.

Στην μέθοδο αυτή παρουσιάζεται σταθερή και ομαλή σύγκλιση, εφόσον το gradient αποτελεί ακριβή εκτίμηση της κατεύθυνσης μείωσης της συνάρτησης κόστους. Εξαιτίας όμως, του μεγάλου υπολογιστικού της κόστους και της απαίτησης μεγάλης μνήμης, ιδιαίτερα όταν το πλήθος των δεδομένων είναι μεγάλο, καθίσταται λιγότερο πρακτική για μεγάλα σύνολα εκπαίδευσης [14].

Stochastic Gradient Descent

Στη Stochastic Gradient Descent (SGD), το gradient υπολογίζεται χρησιμοποιώντας ένα μόνο δείγμα από το σύνολο εκπαίδευσης σε κάθε επανάληψη:

$$\nabla J(\theta) \approx \nabla_\theta \ell(f(x_i;\theta), y_i).$$

Οι παράμετροι ενημερώνονται μετά από κάθε δείγμα. Το γεγονός αυτό οδηγεί σε πολύ συχνές ενημερώσεις [14].

Συνεπώς, είναι προφανές ότι η μέθοδος αυτή μειώνει σημαντικά το υπολογιστικό κόστος ανά επανάληψη και επιτρέπει ταχύτερη πρόοδο στα αρχικά στάδια της εκπαίδευσης. Ωστόσο, λόγω της στοχαστικής φύσης του gradient, η πορεία σύγκλισης παρουσιάζει έντονες διακυμάνσεις και δεν είναι ομαλή. Παρόλα αυτά, ο θόρυβος που εισάγεται μπορεί να βοηθήσει το μοντέλο να αποφύγει τοπικά ελάχιστα.

Με τον όρο θόρυβος (noise) εννοούμε ότι το gradient που υπολογίζουμε δεν είναι ακριβής εκτίμηση της πραγματικής κατεύθυνσης μείωσης της συνάρτησης κόστους, αλλά μια προσεγγιστική και μεταβαλλόμενη εκτίμηση. Συνεπώς, εξαιτίας της κατεύθυνσης του gradient το οποίο αλλάζει, στη μέθοδο αυτή, από update σε update, δηλαδή δεν είναι σταθερή συνεπάγεται κι ότι η πορεία προς το ελάχιστο παρουσιάζει συνεχείς διακυμάνσεις. Άρα, έχουμε αστάθεια στο gradient.

Αντίθετα, στο Batch Gradient Descent που γίνεται χρήση όλων των δεδομένων έχουμε ακριβή μέση κατεύθυνση και συνεπώς, σχεδόν καθόλου θόρυβο.

Mini-batch Gradient Descent

Η Mini-batch Gradient Descent αποτελεί έναν συμβιβασμό μεταξύ των δύο προηγούμενων μεθόδων. Σε αυτή την περίπτωση, το gradient υπολογίζεται

χρησιμοποιώντας ένα μικρό υποσύνολο (mini-batch) του συνόλου εκπαίδευσης, μεγέθους m, όπου:

$$1 < m < N.$$

Το gradient προσεγγίζεται ως:

$$\nabla J(\theta) \approx \frac{1}{m} \sum_{i \in B} \nabla_\theta \, \ell(f(x_i; \theta), y_i),$$

όπου Bείναι το σύνολο των δειγμάτων του mini-batch.

Η μέθοδος αυτή συνδυάζει την αποδοτικότητας της SGD και της σταθερότητας της Batch Gradient Descent. Επιπροσθέτως, επιτρέπει την αποδοτική αξιοποίηση παράλληλων υπολογιστικών αρχιτεκτονικών, γεγονός που την καθιστά την επικρατέστερη επιλογή στη σύγχρονη εκπαίδευση Νευρωνικών Δικτύων. Τυπικές τιμές μεγέθους mini-batch κυμαίνονται συνήθως μεταξύ 16 και 128 δειγμάτων. Η Mini-batch Gradient Descent παρουσιάζει μέτριο θόρυβο, γεγονός που δικαιολογείται από το μικρό σύνολο δείγματος που χρησιμοποιεί [14].

Συνοψίζοντας, οι τρεις παραλλαγές της Gradient Descent διαφέρουν ως προς τον υπολογιστικό φόρτο, τη σταθερότητα της σύγκλισης και την πρακτική εφαρμοσιμότητά τους. Η επιλογή της κατάλληλης μεθόδου εξαρτάται από το μέγεθος του συνόλου δεδομένων και τους διαθέσιμους υπολογιστικούς πόρους, με τη Mini-batch Gradient Descent να αποτελεί τη συνηθέστερη επιλογή στην πράξη.

Χαρακτηριστικό	Κλασσική Gradient Descent (Batch)	Stochastic Gradient Descent (SGD)	Mini-batch Gradient Descent
Σύνολο δεδομένων για τον υπολογισμό του gradient	Ολόκληρο το σύνολο εκπαίδευσης	Ένα μόνο δείγμα	Υποσύνολο (mini-batch) του συνόλου εκπαίδευσης

Αριθμός ενημερώσεων παραμέτρων	Μία ενημέρωση ανά epoch	Μία ενημέρωση ανά δείγμα	Μία ενημέρωση ανά mini-batch
Υπολογιστικό κόστος ανά ενημέρωση	Υψηλό	Πολύ χαμηλό	Μέτριο
Σταθερότητα σύγκλισης	Πολύ σταθερή και ομαλή	Ασταθής, με έντονες διακυμάνσεις	Σχετικά σταθερή
Ταχύτητα αρχικής σύγκλισης	Αργή	Γρήγορη	Γρήγορη
Θόρυβος στο gradient	Καθόλου	Υψηλός	Μέτριος
Απαίτηση μνήμης	Υψηλή	Πολύ χαμηλή	Μέτρια
Καταλληλότητα για μεγάλα σύνολα δεδομένων	Χαμηλή	Υψηλή	Πολύ υψηλή
Πρακτική χρήση στη σύγχρονη εκπαίδευση	Σπάνια	Περιορισμένη	Πολύ συχνή (επικρατέστερη)

Πίνακας 2: Συγκριτικός πίνακας της κλασσικής (Batch) Gradient Descent και των παραλλαγών Stochastic και Mini-batch ως προς τον τρόπο υπολογισμού του gradient και τη συμπεριφορά σύγκλισης.

Πηγή: Ίδια επεξεργασία, βασισμένη στις πηγές [9] και [11].

3.5: Από πού προκύπτει το gradient στα Νευρωνικά Δίκτυα.

Κατά τη περιγραφή της διαδικασίας εκπαίδευσης των Νευρωνικών Δικτύων το gradient εμφανίστηκε ως το βασικό μέγεθος το οποίο μέσω της μεθόδου Gradient Descent καθοδηγεί την ενημέρωση των παραμέτρων του μοντέλου. Ωστόσο, μέχρι αυτό το σημείο, το gradient θεωρήθηκε δεδομένο. Τόσο ο υπολογισμός του gradient στα Νευρωνικά Δίκτυα όσο και η μαθηματική του προέλευση είναι τομείς που συνδέονται άμεσα με την συνάρτηση κόστους [9].

Η συνάρτηση κόστους ενός Νευρωνικού Δικτύου εξαρτάται από τις παραμέτρους του, δηλαδή τα βάρη και τα bias όλων των επιπέδων. Η εξάρτηση αυτή δεν είναι

άμεση, καθώς η συνάρτηση κόστους προκύπτει μέσω διαδοχικών μετασχηματισμών των δεδομένων από τα επιμέρους επίπεδα του δικτύου και, ως εκ τούτου, αποτελεί σύνθεση πολλών συναρτήσεων που εξαρτώνται από όλες τις παραμέτρους του μοντέλου. Πιο συγκεκριμένα, με τον όρο σύνθετη συνάρτηση εννοούμε μια συνάρτηση που δεν παίρνει απευθείας τις εισόδους, αλλά παίρνει ως είσοδο το αποτέλεσμα άλλων συναρτήσεων. Κατά συνέπεια, η συνάρτηση κόστους αποτελεί σύνθετη συνάρτηση πολλών μεταβλητών [9], [12].

Λόγω αυτής της σύνθετης δομής, ο υπολογισμός της παραγώγου της συνάρτησης κόστους ως προς μια παράμετρο δεν μπορεί να γίνει άμεσα. Εξαιτίας αυτού, χρησιμοποιείται ο κανόνας αλυσίδας του διαφορικού λογισμού, ο οποίος επιτρέπει τον υπολογισμό της παραγώγου μιας σύνθετης συνάρτησης μέσω των παραγώγων των επιμέρους συναρτήσεων που τη συνθέτουν [9]

Η αλγοριθμική υλοποίηση του κανόνα αλυσίδας στα Νευρωνικά Δίκτυα πραγματοποιείται μέσω της διαδικασίας backpropagation. Αρχικά, το σφάλμα υπολογίζεται στην έξοδο του δικτύου, μέσω της παραγώγου της συνάρτησης κόστους ως προς την τελική έξοδο. Στη συνέχεια, το σφάλμα αυτό διαδίδεται προς τα πίσω, επίπεδο προς επίπεδο, επιτρέποντας τον υπολογισμό των μερικών παραγώγων της συνάρτησης κόστους ως προς τα βάρη και τα bias κάθε επιπέδου [9].

Το αποτέλεσμα της διαδικασίας αυτής είναι το gradient της συνάρτησης κόστους, το οποίο είναι διάνυσμα που περιέχει μία μερική παράγωγο για κάθε παράμετρο του Νευρωνικού Δικτύου. Κάθε συνιστώσα του gradient εκφράζει τον ρυθμό μεταβολής της συνάρτησης κόστους ως προς τη συγκεκριμένη παράμετρο και παρέχει την απαραίτητη πληροφορία για την κατεύθυνση και το μέγεθος της ενημέρωσής της κατά τη διαδικασία της Gradient Descent [9].

Για να γίνει σαφής η προέλευση του gradient στα Νευρωνικά Δίκτυα, εξετάζεται ένα απλοποιημένο παράδειγμα της διαδικασίας εκπαίδευσης. Έστω ότι το Νευρωνικό Δίκτυο, για ένα δεδομένο δείγμα εισόδου, παράγει μια έξοδο $\hat{y}$, η οποία συγκρίνεται με την επιθυμητή έξοδο y. Η διαφορά μεταξύ της προβλεπόμενης και της πραγματικής τιμής ποσοτικοποιείται μέσω της συνάρτησης κόστους.

Θεωρείται συνάρτηση κόστους το μέσο τετραγωνικό σφάλμα (Mean Squared Error), το οποίο ορίζεται ως:

$$J = \frac{1}{2}(y - \hat{y})^2$$

Η συνάρτηση κόστους εξαρτάται άμεσα από την έξοδο του δικτύου $\hat{y}$, η οποία με τη σειρά της εξαρτάται από τις παραμέτρους του μοντέλου, δηλαδή τα βάρη και τα bias όλων των επιπέδων. Συνεπώς, η εξάρτηση της συνάρτησης κόστους από τις παραμέτρους είναι έμμεση και προκύπτει μέσω μιας ακολουθίας ενδιάμεσων υπολογισμών [9], [12].

Συμβολικά, η εξάρτηση αυτή μπορεί να παρασταθεί ως:

$$J \to \hat{y} \to a^{(L)} \to z^{(L)} \to a^{(L-1)} \to \cdots \to w$$

Εξαιτίας της σύνθετης δομής της, ο υπολογισμός της παραγώγου της συνάρτησης κόστους ως προς ένα βάρος w υπολογίζεται ως:

$$\frac{\partial J}{\partial w} = \frac{\partial J}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial a} \cdot \frac{\partial a}{\partial z} \cdot \frac{\partial z}{\partial w}$$

Η παραπάνω σχέση δείχνει ότι η επίδραση του βάρους w στη συνάρτηση κόστους προκύπτει ως γινόμενο των επιμέρους παραγώγων που συνδέουν το βάρος με την τελική έξοδο του δικτύου. Η διαδικασία αυτή ξεκινά από το σφάλμα στην έξοδο και διαδίδεται προς τα πίσω, επίπεδο προς επίπεδο, επιτρέποντας τον υπολογισμό των μερικών παραγώγων ως προς όλες τις παραμέτρους του μοντέλου [9].

Συνεπώς, το gradient στα Νευρωνικά Δίκτυα προκύπτει άμεσα από τη μαθηματική δομή του δικτύου και τον κανόνα αλυσίδας. Η διαδικασία backpropagation καθιστά εφικτό τον αποδοτικό υπολογισμό του gradient, επιτρέποντας την εκπαίδευση πολυεπίπεδων και μη γραμμικών μοντέλων μέσω μεθόδων βελτιστοποίησης βασισμένων στο gradient.

3.6: Συνήθη προβλήματα στην πράξη.

Κατά τη διαδικασία της εκπαίδευσης των νευρωνικών δικτύων με αλγορίθμους Gradient Descent, εμφανίζονται συχνά προβλήματα τα οποία δυσκολεύουν και σε ορισμένες περιπτώσεις εμποδίζουν τη σύγκλιση του μοντέλου σε ικανοποιητικές λύσεις. Τα προβλήματα αυτά εντοπίζονται κυρίως τόσο στη μορφή της συνάρτησης κόστους όσο και στις επιλογές υπερπαραμέτρων και την ποιότητα των δεδομένων [9], [14].

Με τον όρο υπερπαράμετροι ορίζονται οι παράμετροι που καθορίζουν τη δομή και τη διαδικασία εκπαίδευσης ενός μοντέλου μηχανικής μάθησης και επιλέγονται πριν από την εκπαίδευση, σε αντίθεση με τις παραμέτρους του μοντέλου που προσαρμόζονται αυτόματα μέσω των δεδομένων.

Τα συνήθη προβλήματα παρουσιάζονται ως εξής:

Τοπικά ελάχιστα και σημεία σέλας

Η συνάρτηση κόστους των νευρωνικών δικτύων είναι συνήθως μη κυρτή, με αποτέλεσμα να περιέχει πολλαπλά τοπικά ελάχιστα και σημεία σέλας (saddle points). Τα σημεία σέλας (saddle points) είναι σημεία της συνάρτησης κόστους όπου η κλίση (gradient) είναι μηδενική ή πολύ κοντά στο μηδέν και δεν αποτελούν ούτε τοπικό ελάχιστο ούτε τοπικό μέγιστο. Δηλαδή, η συνάρτηση μειώνεται προς κάποιες κατευθύνσεις και αυξάνεται προς άλλες γύρω από το ίδιο σημείο [9].

Ο Gradient Descent μπορεί να παγιδευτεί σε τέτοιες περιοχές, επιβραδύνοντας ή σταματώντας την εκπαίδευση. Ιδιαίτερα τα σημεία σέλας είναι ιδιαίτερα έντονο πρόβλημα σε νευρωνικά δίκτυα υψηλής διάστασης, καθώς είναι πολύ πιο συχνά από τα τοπικά ελάχιστα. Συνεπώς, το gradient μπορεί να είναι πολύ μικρό χωρίς το σημείο να αντιστοιχεί σε βέλτιστη λύση.

Vanishing και Exploding Gradients

Στην εκπαίδευση βαθιών νευρωνικών δικτύων ένα από τα πιο γνωστά προβλήματα είναι το φαινόμενο των vanishing και exploding gradients.
Στην περίπτωση των vanishing gradients, οι τιμές των παραγώγων γίνονται εξαιρετικά μικρές καθώς το σφάλμα διαδίδεται προς τα πίσω στα αρχικά επίπεδα του δικτύου, με αποτέλεσμα οι αντίστοιχες παράμετροι να ενημερώνονται ελάχιστα. Αντίθετα, στα exploding gradients οι παράγωγοι λαμβάνουν πολύ μεγάλες τιμές, προκαλώντας αστάθεια και απότομες μεταβολές των βαρών. Τα φαινόμενα αυτά επηρεάζουν άμεσα τη σύγκλιση και την απόδοση του μοντέλου [9].

Ακατάλληλη επιλογή learning rate

Κρίσιμος παράγοντας για την αποδοτική λειτουργία του Gradient Descent είναι η επιλογή του κατάλληλου learning rate. Ένα πολύ μεγάλο learning rate μπορεί να προκαλέσει ταλαντώσεις γύρω από το ελάχιστο ή την απόκλιση της συνάρτησης κόστους, ενώ ένα πολύ μικρό learning rate συντελεί στην αργή σύγκλιση και στο

αυξημένο υπολογιστικό κόστος της μεθόδου. Για τον λόγο αυτό, η εύρεση κατάλληλης τιμής απαιτεί πειραματισμό ή τη χρήση τεχνικών προσαρμοστικού learning rate [13], [14].

Υπερπροσαρμογή (Overfitting)

Κατά την εκπαίδευση, το μοντέλο ενδέχεται να προσαρμοστεί υπερβολικά στα δεδομένα εκπαίδευσης, μαθαίνοντας ακόμη και τον θόρυβο που αυτά περιέχουν ή τις τυχαίες ιδιαιτερότητές τους, αντί να μάθει το γενικό μοτίβο του προβλήματος. Το φαινόμενο της υπερπροσαρμογής οδηγεί σε χαμηλό σφάλμα εκπαίδευσης αλλά το μοντέλο αδυνατεί να αναπτύξει την ικανότητα του να δουλεύει σωστά σε νέα δεδομένα, άγνωστα για αυτό. Αν και το πρόβλημα αυτό δεν οφείλεται άμεσα στον Gradient Descent, σχετίζεται με τη διαδικασία εκπαίδευσης και την ενημέρωση των παραμέτρων [12].

Κακή κλιμάκωση και ποιότητα δεδομένων

Η έλλειψη κανονικοποίησης (normalization) των εισόδων μπορεί να επηρεάσει αρνητικά τη συμπεριφορά του Gradient Descent, οδηγώντας σε αργή ή ασταθή σύγκλιση. Με την κανονικοποίηση φέρνουμε όλα τα χαρακτηριστικά σε παρόμοια κλίμακα, π.χ. στο διάστημα [0,1] ή με μέσο όρο 0 και διασπορά 1. Επιπλέον, δεδομένα με θόρυβο ή ελλιπείς τιμές δυσκολεύουν την εκπαίδευση και ενδέχεται να οδηγήσουν σε μη αξιόπιστα μοντέλα [9], [12].

Υπολογιστικό κόστος

Ο υπολογιστικός φόρτος του Gradient Descent μπορεί να γίνει ιδιαίτερα υψηλός σε μεγάλα σύνολα δεδομένων και βαθιά δίκτυα . Η επιλογή batch size, η πολυπλοκότητα του μοντέλου και ο αριθμός των επαναλήψεων (epochs) επηρεάζουν άμεσα τον χρόνο εκπαίδευσης και τους απαιτούμενους πόρους [14].

Σύνοψη της διαδικασίας εκπαίδευσης ενός νευρωνικού δικτύου.

Η διαδικασία εκπαίδευσης ενός Νευρωνικού Δικτύου αρχίζει με την εισαγωγή των δεδομένων εισόδου υπό μορφή διανύσματος. Αυτά πολλαπλασιάζονται με τα αντίστοιχα βάρη και στη συνέχεια αθροίζονται μαζί με τον όρο bias. Το άθροισμα αυτό σχηματίζει το διάνυσμα z, το οποίο σε επόμενο στάδιο διέρχεται από τη συνάρτηση ενεργοποίησης, παράγοντας το διάνυσμα ενεργοποιήσεων a. Το

διάνυσμα a αποτελεί την είσοδο του επόμενου επιπέδου και η διαδικασία επαναλαμβάνεται έως ότου παραχθεί η τελική έξοδος του δικτύου.

Εν συνεχεία, γίνεται σύγκριση της τελικής εξόδου με την επιθυμητή έξοδο μέσω της συνάρτησης κόστους, με τη βοήθεια της οποίας ποσοτικοποιείται το σφάλμα του μοντέλου. Η εκπαίδευση του δικτύου πραγματοποιείται μέσω της Gradient Descent, κατά την οποία υπολογίζονται οι μερικές παράγωγοι της συνάρτησης κόστους ως προς όλες τις παραμέτρους του δικτύου. Μέσω της διαδικασίας backpropagation , το gradient της συνάρτησης κόστους διαδίδεται προς τα πίσω, επιτρέποντας την ενημέρωση των βαρών και των bias σε όλα τα επίπεδα του δικτύου προς την κατεύθυνση μείωσης του κόστους [9].

Η διαδικασία επαναλαμβάνεται έως ότου η μεταβολή της συνάρτησης κόστους από επανάληψη σε επανάληψη γίνει αμελητέα, οπότε η εκπαίδευση τερματίζεται, καθώς η περαιτέρω συνέχιση δεν οδηγεί σε ουσιαστική βελτίωση του μοντέλου. Η εκπαίδευση ενός Νευρωνικού Δικτύου διατυπώνεται ως πρόβλημα ελαχιστοποίησης της συνάρτησης κόστους ως προς τις παραμέτρους του μοντέλου, με στόχο την εύρεση του $\theta^{\backslash *} = \arg\min_\theta J(\theta)$, το οποίο προσεγγίζεται αριθμητικά μέσω της μεθόδου Gradient Descent.

Κεφάλαιο 4: Υλοποίηση Έξυπνου Βοηθού Μελέτης με χρήση Perceptron και Gradient Descent.

Σκοπός του Κεφαλαίου

Το παρόν κεφάλαιο αποτελεί το πρακτικό μέρος της εργασίας και επιδιώκει να μεταφέρει στο επίπεδο της υλοποίησης το θεωρητικό υπόβαθρο που αναπτύχθηκε στα προηγούμενα κεφάλαια. Συγκεκριμένα, υλοποιείται σε γλώσσα Python, εντός περιβάλλοντος Jupyter Notebook, ένα απλό νευρωνικό δίκτυο τύπου μονοεπίπεδου αισθητήρα (Single Layer Perceptron), το οποίο εκπαιδεύεται αποκλειστικά μέσω χειροκίνητης υλοποίησης του αλγορίθμου Καθοδικής Κλίσης (Gradient Descent). Η εφαρμογή δεν αποτελεί αυτοσκοπό· λειτουργεί ως μελέτη περίπτωσης (case study) για την επίδειξη, βήμα προς βήμα, της διαδικασίας εκπαίδευσης ενός νευρωνικού μοντέλου, ώστε να καταστούν εμφανείς οι έννοιες της αρχικοποίησης των βαρών, του υπολογισμού της πρόβλεψης, της συνάρτησης κόστους, του gradient, της ενημέρωσης των παραμέτρων και της σταδιακής μείωσης του σφάλματος.

4.1 Σχεδιασμός και αρχιτεκτονική του έξυπνου βοηθού μελέτης.

Η μετάβαση από τη θεωρία στην πράξη απαιτεί έναν συγκεκριμένο και σαφώς οριοθετημένο πρακτικό προβληματισμό, ο οποίος να επιτρέπει την εφαρμογή των βασικών αρχών της εκπαίδευσης νευρωνικών δικτύων χωρίς υπερβολική πολυπλοκότητα. Για τον σκοπό αυτό σχεδιάστηκε ένας Έξυπνος Βοηθός Μελέτης (Smart Study Assistant), μια εφαρμογή που εκτιμά την αναμενόμενη ακαδημαϊκή επίδοση ενός μαθητή με βάση λίγα, ερμηνεύσιμα χαρακτηριστικά και, στη συνέχεια, παρέχει εξατομικευμένες συστάσεις μελέτης.

Η ανάγκη που καλύπτει η εφαρμογή είναι διττή. Από εκπαιδευτική σκοπιά, η πρόβλεψη της επίδοσης και η έγκαιρη καθοδήγηση του μαθητή αποτελούν αναγνωρισμένο πεδίο εφαρμογής της εξόρυξης εκπαιδευτικών δεδομένων. Από τη σκοπιά της παρούσας εργασίας, το σενάριο αυτό προσφέρει ένα διαφανές πλαίσιο μέσα στο οποίο κάθε επιμέρους βήμα του Gradient Descent μπορεί να παρατηρηθεί και να ερμηνευθεί. Η επιλογή ενός εκπαιδευτικού σεναρίου, αντί ενός αφηρημένου μαθηματικού προβλήματος, επιτρέπει επίσης τη διαισθητική κατανόηση των αποτελεσμάτων, καθώς τόσο οι είσοδοι όσο και η έξοδος έχουν άμεσο νόημα για τον αναγνώστη [9], [12].

Στόχος της εφαρμογής είναι, δεδομένων των ωρών μελέτης, της προηγούμενης επίδοσης, του αριθμού των προηγούμενων αποτυχιών και των απουσιών ενός μαθητή, να εκτιμηθεί ο αναμενόμενος τελικός του βαθμός και, με βάση αυτή την εκτίμηση, να προταθεί κατάλληλο επίπεδο ασκήσεων, να αξιολογηθεί αν η τρέχουσα προετοιμασία επαρκεί για έναν επιθυμητό στόχο και να διατυπωθεί πρόταση ως προς την ένταση της μελέτης.

Επιλογή του Perceptron και σύνδεση με τον Gradient Descent

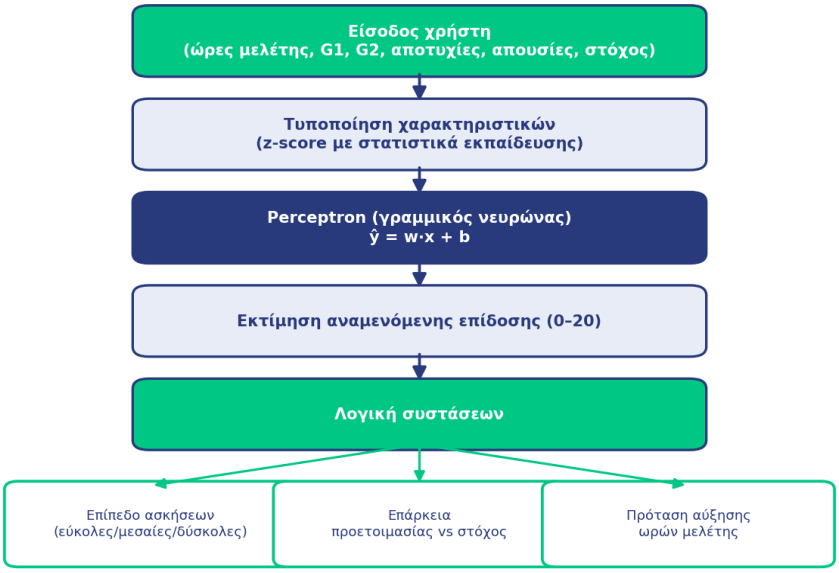
Ως μοντέλο επιλέχθηκε ο μονοεπίπεδος αισθητήρας, καθώς αποτελεί την απλούστερη μορφή νευρωνικού δικτύου και αναπαριστά με σαφήνεια τη θεμελιώδη υπολογιστική αρχή «είσοδοι - βάρη - σταθμισμένο άθροισμα - ενεργοποίηση». Επειδή το πρόβλημα είναι παλινδρόμησης, δηλαδή η εκτίμηση μιας συνεχούς

αριθμητικής τιμής (του βαθμού), ο νευρώνας χρησιμοποιεί γραμμική (ταυτοτική) συνάρτηση ενεργοποίησης. Με τη διαμόρφωση αυτή, ο μονοεπίπεδος νευρώνας που εκπαιδεύεται με Gradient Descent και κριτήριο το μέσο τετραγωνικό σφάλμα ταυτίζεται μαθηματικά με τον προσαρμοστικό γραμμικό νευρώνα (Adaline) των Widrow και Hoff (1960), ο οποίος αποτέλεσε την πρώτη ιστορικά εφαρμογή της εκπαίδευσης ενός νευρώνα μέσω βαθμωτής μείωσης του σφάλματος.

Η σύνδεση με τον Gradient Descent είναι άμεση: τα βάρη και το bias του νευρώνα αρχικοποιούνται και, σε κάθε επανάληψη, ενημερώνονται προς την αντίθετη κατεύθυνση από εκείνη που υποδεικνύει το gradient της συνάρτησης κόστους, μέχρι το σφάλμα να σταθεροποιηθεί. Έτσι, η εφαρμογή λειτουργεί ως ζωντανή απεικόνιση της θεωρίας του τρίτου κεφαλαίου [9], [12].

Αρχιτεκτονική της εφαρμογής

Η λειτουργία της εφαρμογής οργανώνεται σε διαδοχικά στάδια. Τα δεδομένα του χρήστη τυποποιούνται με τα στατιστικά του συνόλου εκπαίδευσης, τροφοδοτούνται στον εκπαιδευμένο νευρώνα, ο οποίος παράγει την εκτιμώμενη επίδοση, και τέλος η εκτίμηση αυτή μετατρέπεται σε πρακτικές συστάσεις μέσω μιας λογικής αποφάσεων.



ΣΧΗΜΑ 7: ***Διάγραμμα ροής λειτουργίας του Έξυπνου Βοηθού Μελέτης, από την είσοδο του χρήστη έως τις τελικές συστάσεις.***

Πηγή: Ίδια επεξεργασία.

4.2 Ανάπτυξη και εκπαίδευση του μοντέλου πρόβλεψης.

Στην ενότητα αυτή παρουσιάζεται σταδιακά ο κώδικας της υλοποίησης. Για κάθε τμήμα προηγείται σύντομη θεωρητική περιγραφή και ακολουθεί ανάλυση της λειτουργίας του. Ο πλήρης, εκτελέσιμος κώδικας περιλαμβάνεται στο συνοδευτικό αρχείο Jupyter Notebook.

4.2.3 Εισαγωγή βιβλιοθηκών.

Χρησιμοποιούνται η NumPy για τις διανυσματικές πράξεις (διανύσματα, πίνακες, πράξεις γραμμικής άλγεβρας, υπολογισμό μέσου όρου, υπολογισμό τυπικής απόκλισης, Gradient Descent, υπολογισμό του perceptron), η Pandas για τη διαχείριση του συνόλου δεδομένων. Επειδή το dataset βρίσκεται σε αρχείο CSV, η Pandas: ανοίγει το dataset, το μετατρέπει σε πίνακα (DataFrame) και επιτρέπει να

επιλέξουμε μόνο τις στήλες που θέλουμε. Εν συνεχεία, η Matplotlib χρησιμοποιείται για τα γραφήματα, καθώς με τη βοήθεια της παρουσιάζεται η μείωση της συνάρτησης κόστους, η πορεία της εκπαίδευσης και τα αποτελέσματα του μοντέλου. Από τη βιβλιοθήκη scikit-learn αξιοποιείται μόνο η βοηθητική συνάρτηση διαχωρισμού των δεδομένων (το μοντέλο μαθαίνει μόνο από το training set και στο τέλος ελέγχεται αν μπορεί να προβλέψει σωστά σε μαθητές που δεν έχει ξαναδεί)· η εκπαίδευση του μοντέλου υλοποιείται εξ ολοκλήρου χειροκίνητα. Τέλος, η random είναι υποενότητα της βιβλιοθήκης NumPy και η seed() είναι συνάρτησή της. Με την εντολή np.random.seed(42) ορίζεται ένας σταθερός τυχαίος σπόρος, ώστε κάθε φορά που εκτελείται το πρόγραμμα τα αρχικά βάρη να είναι τα ίδια, ο διαχωρισμός train/test να είναι ο ίδιος με στόχο να παράγει τα ίδια τυχαία αποτελέσματα και η διαδικασία να είναι αναπαραγώγιμη. Δηλαδή αν ένας άλλος ερευνητής τρέξει τον ίδιο κώδικα με το ίδιο dataset, θα πάρει τα ίδια αποτελέσματα [15], [16], [17], [18].

```
import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split

np.random.seed(42)  # αναπαραγωγιμότητα
```

4.2.2 Φόρτωση του συνόλου δεδομένων.

Χρησιμοποιείται το σύνολο δεδομένων Student Performance, το οποίο συγκεντρώθηκε από τους Cortez και Silva (2008) και διατίθεται δημόσια μέσω του UCI Machine Learning Repository (Cortez, 2008). Περιέχει στοιχεία μαθητών δευτεροβάθμιας εκπαίδευσης δύο πορτογαλικών σχολείων και περιλαμβάνει δημογραφικά, κοινωνικά και ακαδημαϊκά χαρακτηριστικά. Από το σύνολο των δεδομένων αξιοποιείται το αρχείο student-mat.csv (μάθημα Μαθηματικών), το οποίο περιέχει 395 εγγραφές και 33 χαρακτηριστικά. Η παράμετρος sep=";" δηλώνει ότι στο συγκεκριμένο αρχείο CSV οι στήλες διαχωρίζονται με ελληνικό ερωτηματικό (;), έτσι η Pandas διαβάζει σωστά κάθε χαρακτηριστικό ως ξεχωριστή στήλη. Η Pandas ανοίγει το αρχείο student-mat.csv και το αποθηκεύει σε μία μεταβλητή που λέγεται df, το οποίο σημαίνει DataFrame, δηλαδή ένας πίνακας

δεδομένων. Η ιδιότητα shape επιστρέφει τις διαστάσεις του DataFrame, δηλαδή το χρησιμοποιούμε για να επιβεβαιώσουμε ότι το dataset φορτώθηκε σωστά και ότι περιέχει τον αναμενόμενο αριθμό δειγμάτων και χαρακτηριστικών πριν ξεκινήσει η επεξεργασία των δεδομένων [17].

```
df = pd.read_csv("student-mat.csv", sep=";")

print(df.shape)   # (395, 33)
```

4.2.3 Επιλογή χαρακτηριστικών.

Σύμφωνα με την αρχή της απλότητας, δηλαδή αποφυγή άσχετων πληροφοριών, επιλογή μόνο των πιο σημαντικών, ώστε το μοντέλο να είναι απλό και εύκολα ερμηνεύσιμο διατηρούνται μόνο πέντε χαρακτηριστικά εισόδου από τα 33 διαθέσιμα και μία μεταβλητή-στόχος. Η επιλογή αιτιολογείται από την παιδαγωγική τους σημασία και από τα ευρήματα της σχετικής βιβλιογραφίας, όπου επισημαίνεται η ισχυρή προγνωστική αξία των ενδιάμεσων βαθμών G1 και G2 για τον τελικό βαθμό G3.

Χαρακτηριστικό	Περιγραφή	Αιτιολόγηση
studytime	Εβδομαδιαίες ώρες μελέτης (1–4)	Άμεσος δείκτης προσπάθειας
G1	Βαθμός 1ης περιόδου (0–20)	Προηγούμενη επίδοση
G2	Βαθμός 2ης περιόδου (0–20)	Πρόσφατη προηγούμενη επίδοση
failures	Προηγούμενες αποτυχίες (0–4)	Δείκτης δυσκολιών
absences	Αριθμός απουσιών	Δείκτης εμπλοκής/συνέπειας
G3 (στόχος)	Τελικός βαθμός (0–20)	Μεταβλητή προς πρόβλεψη

Πίνακας 3: Χαρακτηριστικά εισόδου του έξυπνου βοηθού μελέτης.

Πηγή:Ίδια επεξεργασία, βασισμένη στο σύνολο δεδομένων Student Performance Dataset.

Στον πίνακα παρουσιάζονται τα χαρακτηριστικά που διατηρήθηκαν και τον λόγο της άμεσης επιρροής τους στο τελικό αποτέλεσμα. Το studytime είναι άμεσος δείκτης προσπάθειας, καθώς όσο περισσότερο διαβάζει ένας μαθητής, τόσο πιθανότερο είναι να έχει καλύτερη επίδοση. Το G1 (Προηγούμενη επίδοση) είναι ο βαθμός του πρώτου τετραμήνου και αποτελεί ισχυρή ένδειξη για τον τελικό βαθμό. Αντίστοιχα, το G2 (Πρόσφατη προηγούμενη επίδοση) είναι ο βαθμός που είναι ακόμη πιο κοντά στον τελικό βαθμό, επομένως αποτελεί ακόμη καλύτερο προγνωστικό δείκτη. Τα failures (Δείκτης δυσκολιών) αποτελούν προηγούμενες αποτυχίες και συνήθως συνδέονται με χαμηλότερη τελική επίδοση.Τα absences (Δείκτης συνέπειας), δηλαδή οι απουσίες συνήθως σημαίνουν μικρότερη συμμετοχή στο μάθημα και τέλος το G3 που δεν είναι είσοδος αλλά η μεταβλητή που πρέπει να προβλεφθεί [9].

4.2.4 Διαχωρισμός σε σύνολο εκπαίδευσης και ελέγχου.

Τα 395 δείγματα χωρίζονται σε 80% εκπαίδευση (316 δείγματα) και 20% έλεγχο (79 δείγματα). Ο διαχωρισμός επιτρέπει την αξιολόγηση της ικανότητας γενίκευσης του μοντέλου σε δεδομένα που δεν χρησιμοποιήθηκαν κατά τη διαδικασία της εκπαίδευσης. Η συνάρτηση train_test_split() λαμβάνει ως είσοδο τα χαρακτηριστικά X και τη μεταβλητή-στόχο y και τα διαχωρίζει αυτόματα σε τέσσερα σύνολα: X_{train}, X_{test}, y_{train} και y_{test}, όπου το X_train περιέχει τις εισόδους των 316 μαθητών, το y_train περιέχει τους πραγματικούς G3 αυτών των μαθητών, η X_test περιέχει τις εισόδους των 79 μαθητών και η y_test τους πραγματικούς G3 των 79 μαθητών. Η παράμετρος test_size=0.2 ορίζει ότι το 20% των δειγμάτων θα χρησιμοποιηθεί ως σύνολο ελέγχου (test set), ενώ το υπόλοιπο 80% ως σύνολο εκπαίδευσης (training set). Η παράμετρος random_state=42 δεν καθορίζει το ποσοστό του διαχωρισμού, αλλά εξασφαλίζει ότι ο τυχαίος διαχωρισμός θα είναι πάντα ο ίδιος σε κάθε εκτέλεση του προγράμματος, ώστε τα αποτελέσματα να είναι αναπαραγώγιμα, δηλαδή οι ίδιοι 316 μαθητές θα βρίσκονται πάντα στο training set, και οι ίδιοι 79 μαθητές θα βρίσκονται πάντα στο test set [16].

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

4.2.5 Κανονικοποίηση (τυποποίηση z-score).

Στη συνέχεια εφαρμόζεται τυποποίηση των χαρακτηριστικών με τη μέθοδο Z-score, καθώς τα επιλεγμένα χαρακτηριστικά έχουν διαφορετικές κλίμακες τιμών (για παράδειγμα οι ώρες μελέτης κυμαίνονται από 1 έως 4, οι βαθμοί από 0 έως 20 και οι απουσίες μπορούν να λάβουν πολύ μεγαλύτερες τιμές). Αν δεν προηγηθεί αυτή η διαδικασία, χαρακτηριστικά με μεγαλύτερες αριθμητικές τιμές θα επηρεάζουν δυσανάλογα την εκπαίδευση του perceptron και τη σύγκλιση του αλγορίθμου Gradient Descent. Για τον λόγο αυτό, υπολογίζονται αρχικά, αποκλειστικά από το σύνολο εκπαίδευσης, ο μέσος όρος (μ) και η τυπική απόκλιση (σ) κάθε χαρακτηριστικού. Έπειτα, κάθε τιμή μετασχηματίζεται σύμφωνα με τον τύπο $z = \frac{x-\mu}{\sigma}$ όπου x είναι η αρχική τιμή ενός χαρακτηριστικού για έναν συγκεκριμένο μαθητή (για παράδειγμα οι ώρες μελέτης ή ο βαθμός G1), μ είναι ο μέσος όρος του ίδιου χαρακτηριστικού στο σύνολο εκπαίδευσης και σ είναι η αντίστοιχη τυπική απόκλιση που μετρά πόσο αποκλίνουν οι τιμές των μαθητών από αυτόν τον μέσο όρο. Με την εφαρμογή του μετασχηματισμού Z-score, οι τιμές κάθε χαρακτηριστικού μετασχηματίζονται έτσι ώστε το σύνολο των τιμών του συγκεκριμένου χαρακτηριστικού να έχει μέσο όρο 0 και τυπική απόκλιση 1. Με τον τρόπο αυτό όλα τα χαρακτηριστικά εκφράζονται στην ίδια στατιστική κλίμακα και μπορούν να συγκριθούν ισότιμα κατά την εκπαίδευση του μοντέλου

```
mu = X_train.mean(axis=0)

sigma = X_train.std(axis=0)

X_train_std = (X_train - mu) / sigma

X_test_std  = (X_test  - mu) / sigma
```

- Στο πρώτο βήμα υπολογίζεται ο μέσος όρος (μ) κάθε χαρακτηριστικού χρησιμοποιώντας αποκλειστικά τα δεδομένα του συνόλου εκπαίδευσης (Training Set). Η συνάρτηση mean(axis=0) υπολογίζει τον μέσο όρο ανά στήλη, δηλαδή για κάθε χαρακτηριστικό ξεχωριστά (studytime, G1, G2, failures και absences).
- Στη συνέχεια υπολογίζεται η τυπική απόκλιση (σ) κάθε χαρακτηριστικού, επίσης μόνο από το σύνολο εκπαίδευσης. Η συνάρτηση std(axis=0)

υπολογίζει την τυπική απόκλιση ανά στήλη, εκφράζοντας πόσο αποκλίνουν οι τιμές κάθε χαρακτηριστικού από τον αντίστοιχο μέσο όρο τους.

- Για κάθε τιμή εφαρμόζεται ο τύπος $z = \frac{x-\mu}{\sigma}$.
- Τέλος, εφαρμόζεται η ίδια διαδικασία τυποποίησης και στο σύνολο ελέγχου (Test Set). Για λόγους επιστημονικής ορθότητας, δεν υπολογίζονται νέοι μέσοι όροι και νέες τυπικές αποκλίσεις από τα δεδομένα ελέγχου. Αντίθετα, χρησιμοποιούνται οι ίδιες τιμές μ και σπου υπολογίστηκαν από το σύνολο εκπαίδευσης, ώστε να αποφευχθεί η διαρροή πληροφορίας (data leakage) και να διασφαλιστεί αντικειμενική αξιολόγηση του μοντέλου [16].

4.2.6 Δημιουργία του Perceptron και συνάρτηση πρόβλεψης.

Ο μονοεπίπεδος νευρώνας υπολογίζει το σταθμισμένο άθροισμα των εισόδων και προσθέτει τον όρο μετατόπισης (bias). Με γραμμική ενεργοποίηση, η πρόβλεψη δίνεται από τη σχέση ŷ = w·x + b.

```
def predict(X, w, b):
    return X @ w + b
```

- Με την εντολή def predict(X, w, b): ορίζεται η συνάρτηση πρόβλεψης του perceptron. Η συνάρτηση δέχεται ως είσοδο το διάνυσμα των χαρακτηριστικών του μαθητή (X), το διάνυσμα των βαρών (w) και τον όρο μετατόπισης (b). Σκοπός της είναι να υπολογίσει την προβλεπόμενη έξοδο του γραμμικού νευρώνα χρησιμοποιώντας τις παραμέτρους που προέκυψαν από τη διαδικασία εκπαίδευσης.
- Η εντολή return X @ w + b υπολογίζει την έξοδο του γραμμικού νευρώνα. Αρχικά πραγματοποιείται το εσωτερικό γινόμενο μεταξύ του διανύσματος χαρακτηριστικών (X) και του διανύσματος βαρών (w), δηλαδή κάθε χαρακτηριστικό πολλαπλασιάζεται με το αντίστοιχο βάρος του και τα επιμέρους γινόμενα αθροίζονται. Στη συνέχεια προστίθεται ο όρος μετατόπισης (bias), ώστε να παραχθεί η τελική πρόβλεψη (ŷ) του perceptron, σύμφωνα με τη σχέση $\hat{y} = w \cdot x + b$, όπου $\hat{y}$ ο προβλεπόμενος τελικός βαθμός του μαθητή ($\widehat{G3}$) [4], [9].

4.2.7 Συνάρτηση κόστους.

Ως κριτήριο σφάλματος χρησιμοποιείται το μέσο τετραγωνικό σφάλμα,

$J = (1/2N)\cdot\Sigma(\hat{y} - y)^2$, κατάλληλο για προβλήματα παλινδρόμησης. Ο συντελεστής 1/2 διευκολύνει την παραγώγιση χωρίς να επηρεάζει τη θέση του ελαχίστου.

```
def cost_mse(X, y, w, b):
    error = predict(X, w, b) - y
    return (error @ error) / (2 * len(y))
```

- Με την εντολή def cost_mse(X, y, w, b): ορίζεται η συνάρτηση υπολογισμού του μέσου τετραγωνικού σφάλματος. Η συνάρτηση δέχεται ως είσοδο τα χαρακτηριστικά των μαθητών (X), τους πραγματικούς τελικούς βαθμούς (y), καθώς και τα τρέχοντα βάρη (w) και το bias (b) του perceptron.
- Η εντολή error = predict(X, w, b) - y υπολογίζει το σφάλμα πρόβλεψης για κάθε μαθητή. Αρχικά καλείται η συνάρτηση predict(), η οποία υπολογίζει τους προβλεπόμενους τελικούς βαθμούς ($\hat{y}$). Στη συνέχεια αφαιρούνται οι πραγματικοί βαθμοί (y), ώστε να προκύψει η απόκλιση μεταξύ πρόβλεψης και πραγματικής τιμής για κάθε δείγμα.
- Με την εντολή return (error @ error) / (2 * len(y)) υπολογίζεται το συνολικό κόστος του μοντέλου. Αρχικά υπολογίζεται το άθροισμα των τετραγώνων όλων των σφαλμάτων (error @ error) και στη συνέχεια το αποτέλεσμα διαιρείται με το $2N$, όπου N είναι το πλήθος των μαθητών του συνόλου εκπαίδευσης (len(y)). Έτσι προκύπτει μία μόνο τιμή, η συνάρτηση κόστους J για τη συγκεκριμένη επανάληψη. Επομένως η cost_mse() δεν επιστρέφει 316 τιμές, αλλά μία μόνο τιμή, που εκφράζει πόσο καλά ή πόσο άσχημα αποδίδει συνολικά το μοντέλο στην τρέχουσα επανάληψη. Αυτή η τιμή είναι που προσπαθεί να ελαχιστοποιήσει ο Gradient Descent με την ενημέρωση των βαρών και των bias [12].

4.2.8 Χειροκίνητη υλοποίηση του Gradient Descent.

Αφού υπολογιστεί η συνάρτηση κόστους J, πρέπει να ενημερωθούν τα βάρη και το bias κατάλληλα ώστε να μειωθεί. Το gradient της συνάρτησης κόστους ως προς τα

βάρη είναι $(1/N)\cdot X^T(\hat{y} - y)$ και ως προς το bias ο μέσος όρος του σφάλματος. Σε κάθε epoch οι παράμετροι ενημερώνονται κατά $-\eta \cdot$gradient. Πρόκειται για Batch Gradient Descent, καθώς το gradient υπολογίζεται σε ολόκληρο το σύνολο εκπαίδευσης. Στον κώδικα διακρίνονται καθαρά τα πέντε βήματα: πρόβλεψη, σφάλμα, υπολογισμός gradient, ενημέρωση παραμέτρων και καταγραφή του κόστους. Η διαδικασία αυτή επαναλαμβάνεται σε κάθε epoch μέχρι να ολοκληρωθεί η εκπαίδευση και να επιτευχθεί η σύγκλιση του μοντέλου.

```
def train_gradient_descent(X, y, lr=0.1, epochs=400):
    n, d = X.shape
    w = np.zeros(d)        # αρχικοποίηση βαρών
    b = 0.0                # αρχικοποίηση bias
    history = []
    for epoch in range(epochs):
        y_hat = predict(X, w, b)          # 1. πρόβλεψη
        error = y_hat - y                 # 2. σφάλμα
        grad_w = (X.T @ error) / n        # 3. gradient (w)
        grad_b = error.mean()             #    gradient (b)
        w -= lr * grad_w                  # 4. ενημέρωση w
        b -= lr * grad_b                  #    ενημέρωση b
        history.append(cost_mse(X, y, w, b))# 5. καταγραφή loss
    return w, b, history
```

- Ορίζεται η συνάρτηση train_gradient_descent, η οποία υλοποιεί την εκπαίδευση του perceptron με χρήση του αλγορίθμου Gradient Descent. Η συνάρτηση δέχεται ως είσοδο τον πίνακα χαρακτηριστικών X, το διάνυσμα πραγματικών τιμών y, τον ρυθμό μάθησης (learning rate) και τον αριθμό

των επαναλήψεων (epochs), ενώ στο τέλος επιστρέφει τα τελικά βάρη, το bias και το ιστορικό της συνάρτησης κόστους.

- Η εντολή n, d = X.shape υπολογίζει τις διαστάσεις του πίνακα εισόδου X, όπου το n αντιστοιχεί στο πλήθος των δειγμάτων (μαθητών) και το d στο πλήθος των χαρακτηριστικών εισόδου που χρησιμοποιούνται για την εκπαίδευση του μοντέλου. Οι πληροφορίες αυτές χρησιμοποιούνται τόσο για την αρχικοποίηση των παραμέτρων όσο και για τον υπολογισμό των gradients.
- Αρχικοποιείται το διάνυσμα των βαρών με μηδενικές τιμές (w = np.zeros(d)). Το μέγεθος του διανύσματος είναι ίσο με το πλήθος των χαρακτηριστικών εισόδου, ώστε κάθε χαρακτηριστικό να αντιστοιχεί σε ένα βάρος. Η επιλογή μηδενικών αρχικών τιμών είναι κατάλληλη στη συγκεκριμένη εφαρμογή, καθώς χρησιμοποιείται ένα απλό perceptron με μία μόνο έξοδο.
- Αρχικοποιείται ο όρος μετατόπισης (bias) με τιμή μηδέν (b = 0.0). Το bias αποτελεί ανεξάρτητη παράμετρο του μοντέλου και κατά την εκπαίδευση ενημερώνεται μαζί με τα βάρη, συμβάλλοντας στη βελτίωση της τελικής πρόβλεψης.
- Δημιουργείται μία κενή λίστα (history = []) για την αποθήκευση της τιμής της συνάρτησης κόστους σε κάθε εποχή. Το ιστορικό αυτό χρησιμοποιείται αργότερα για την απεικόνιση της μεταβολής του loss και τον έλεγχο της σύγκλισης του αλγορίθμου.
- Με την εντολή for epoch in range (epochs) ξεκινά η επαναληπτική διαδικασία εκπαίδευσης. Κάθε επανάληψη αντιστοιχεί σε μία εποχή (epoch), κατά την οποία χρησιμοποιούνται όλα τα δείγματα του συνόλου εκπαίδευσης, καθώς υλοποιείται η Batch Gradient Descent.
- Υπολογίζονται οι προβλεπόμενες τιμές (y_hat = predict(X, w, b)) εφαρμόζοντας τον γραμμικό μετασχηματισμό Wx+bσε όλα τα δείγματα του συνόλου εκπαίδευσης. Οι τιμές αυτές αποτελούν την έξοδο του μοντέλου πριν από οποιαδήποτε ενημέρωση των παραμέτρων.
- Υπολογίζεται το διάνυσμα σφαλμάτων (error = y_hat - y), το οποίο προκύπτει από τη διαφορά μεταξύ των προβλεπόμενων και των

πραγματικών τιμών. Κάθε στοιχείο του διανύσματος αντιστοιχεί στο σφάλμα πρόβλεψης ενός συγκεκριμένου μαθητή. Θετικές τιμές δηλώνουν υπερεκτίμηση της πρόβλεψης, ενώ αρνητικές υποεκτίμηση.

- Υπολογίζεται το gradient της συνάρτησης κόστους ως προς τα βάρη (grad_w = (X.T @ error) / n). Το gradient είναι διάνυσμα που περιέχει μία τιμή για κάθε βάρος του μοντέλου και δείχνει τόσο το μέγεθος όσο και την κατεύθυνση της μεταβολής που απαιτείται, ώστε να μειωθεί η συνάρτηση κόστους. Ο πολλαπλασιασμός με τον μετασχηματισμένο πίνακα X^Tεπιτρέπει τον ταυτόχρονο υπολογισμό των gradients για όλα τα βάρη, ενώ η διαίρεση με το n υπολογίζει τον μέσο όρο για όλα τα δείγματα του συνόλου εκπαίδευσης.
- Υπολογίζεται το gradient ως προς το bias (grad_b = error.mean()), το οποίο ισούται με τον μέσο όρο των σφαλμάτων όλων των δειγμάτων. Επειδή υπάρχει μόνο μία παράμετρος bias, αρκεί μία μόνο τιμή gradient για την ενημέρωσή του.
- Ενημερώνονται τα βάρη σύμφωνα με τον κανόνα της Gradient Descent (w -= lr * grad_w). Από κάθε βάρος αφαιρείται το γινόμενο του learning rate με το αντίστοιχο gradient. Αν το gradient είναι θετικό, το βάρος μειώνεται, ενώ αν είναι αρνητικό, το βάρος αυξάνεται. Με τον τρόπο αυτό τα βάρη μεταβάλλονται προς την κατεύθυνση που οδηγεί στη μείωση της συνάρτησης κόστους.
- Με ανάλογο τρόπο ενημερώνεται και το bias (b -= lr * grad_b), εφαρμόζοντας τον ίδιο κανόνα ενημέρωσης. Η ταυτόχρονη προσαρμογή των βαρών και του bias επιτρέπει στο μοντέλο να βελτιώνει σταδιακά τις προβλέψεις του σε κάθε εποχή.
- Υπολογίζεται και αποθηκεύεται η νέα τιμή της συνάρτησης κόστους (history.append(cost_mse(X, y, w, b))). Η αποθήκευση αυτή επιτρέπει την παρακολούθηση της εξέλιξης του loss κατά τη διάρκεια της εκπαίδευσης και χρησιμοποιείται αργότερα για την κατασκευή του αντίστοιχου διαγράμματος.
- Με την ολοκλήρωση όλων των εποχών, η συνάρτηση επιστρέφει τα τελικά βάρη, το bias και το ιστορικό της συνάρτησης κόστους. Οι παράμετροι

αυτές αποτελούν το εκπαιδευμένο μοντέλο και χρησιμοποιούνται στη συνέχεια για την πραγματοποίηση προβλέψεων σε νέα δεδομένα [9], [13].

4.2.9 Εκπαίδευση του μοντέλου.

Η εκπαίδευση εκτελείται με ρυθμό μάθησης η = 0,1 για 400 epochs. Το κόστος μειώνεται από 50,65 (αρχικά) σε 1,70 (τελικά) στο σύνολο εκπαίδευσης, γεγονός που επιβεβαιώνει τη σύγκλιση. Τα εκπαιδευμένα βάρη (επί τυποποιημένων χαρακτηριστικών) και το bias παρουσιάζονται στον Πίνακα που ακολουθεί.

Παράμετρος	Τιμή	Ερμηνεία
w(studytime)	−0,060	Αμελητέα συνεισφορά
w(G1)	+0,466	Θετική συνεισφορά
w(G2)	+3,672	Κυρίαρχος προγνωστικός παράγοντας
w(failures)	−0,335	Αρνητική συνεισφορά
w(absences)	+0,330	Μικρή θετική συνεισφορά
bias (b)	+10,326	≈ μέσος βαθμός του δείγματος

Πίνακας 4: Ερμηνεία των τιμών των χαρακτηριστικών.

Πηγή: Ίδια επεξεργασία.

Το πρόσημο του βάρους δείχνει την κατεύθυνση της επίδρασης: θετικό βάρος σημαίνει ότι η αύξηση του αντίστοιχου χαρακτηριστικού οδηγεί σε αύξηση της προβλεπόμενης τιμής, ενώ αρνητικό βάρος σημαίνει ότι η αύξηση του χαρακτηριστικού οδηγεί σε μείωση της πρόβλεψης. Η ένταση της επίδρασης δεν καθορίζεται από το πρόσημο αλλά από την απόλυτη τιμή του βάρους· όσο μεγαλύτερη είναι η απόλυτη τιμή, τόσο μεγαλύτερη είναι η συμβολή του συγκεκριμένου χαρακτηριστικού στην τελική πρόβλεψη του μοντέλου.

Παρατηρείται ότι ο βαθμός της δεύτερης περιόδου (G2) κυριαρχεί, αποτέλεσμα αναμενόμενο δεδομένης της ισχυρής συσχέτισής του με τον τελικό βαθμό. Το bias προσεγγίζει τον μέσο βαθμό του δείγματος, όπως είναι αναμενόμενο όταν τα χαρακτηριστικά είναι τυποποιημένα [18].

4.2.10 Εμφάνιση του loss ανά epoch.

Στο Σχήμα 4.2 παρουσιάζεται η μεταβολή της συνάρτησης κόστους (loss) κατά τη διάρκεια της εκπαίδευσης του perceptron. Η καμπύλη κόστους μειώνεται απότομα στα πρώτα epochs γεγονός που δείχνει ότι το μοντέλο μαθαίνει γρήγορα και προσαρμόζει αποτελεσματικά τα βάρη και το bias. Στη συνέχεια σταθεροποιείται, χωρίς ταλαντώσεις, υποδεικνύοντας κατάλληλη επιλογή ρυθμού μάθησης και ομαλή σύγκλιση. Επιπλέον, οι καμπύλες του συνόλου εκπαίδευσης (train) και του συνόλου ελέγχου (test) παραμένουν πολύ κοντά μεταξύ τους σε όλη τη διάρκεια της εκπαίδευσης, χωρίς να εμφανίζουν σημαντική απόκλιση. Γεγονός που αποτελεί ισχυρή ένδειξη ότι το μοντέλο παρέχει ικανοποιητικά αποτελέσματα ακόμη και στα δεδομένα που δεν χρησιμοποιήθηκαν για την εκπαίδευση και δεν παρουσιάζει σοβαρά φαινόμενα υπερπροσαρμογής (overfitting) [12], [16].

ΣΧΗΜΑ 8: ***Μείωση της συνάρτησης κόστους ανά epoch, για το σύνολο εκπαίδευσης και ελέγχου.***

Πηγή: Ίδια επεξεργασία.

4.2.11 Αξιολόγηση του μοντέλου.

Η αξιολόγηση στηρίζεται σε τέσσερις δείκτες: μέσο τετραγωνικό σφάλμα (MSE), ρίζα του μέσου τετραγωνικού σφάλματος (RMSE), μέσο απόλυτο σφάλμα (MAE) και συντελεστή προσδιορισμού (R^2). Το MSE μετρά το μέσο τετραγωνικό λάθος της προβλεπόμενης τιμής από την πραγματική, ενώ το RMSE είναι η τετραγωνική ρίζα του MSE, φέρνοντας το σφάλμα στην ίδια κλίμακα με τα δεδομένα. Όσο μικρότερες είναι οι τιμές των δύο αυτών δεικτών, τόσο ακριβέστερες είναι οι προβλέψεις του μοντέλου. Το MAE υπολογίζει πόσο απέχουν οι προβλέψεις από τις πραγματικές τιμές σε απόλυτες μονάδες, και μετά βρίσκει τον μέσο όρο αυτών των αποκλίσεων, ενώ το R^2 δείχνει το ποσοστό της διακύμανσης των δεδομένων που εξηγεί το μοντέλο.Τα αποτελέσματα συνοψίζονται στον Πίνακα 4.3.

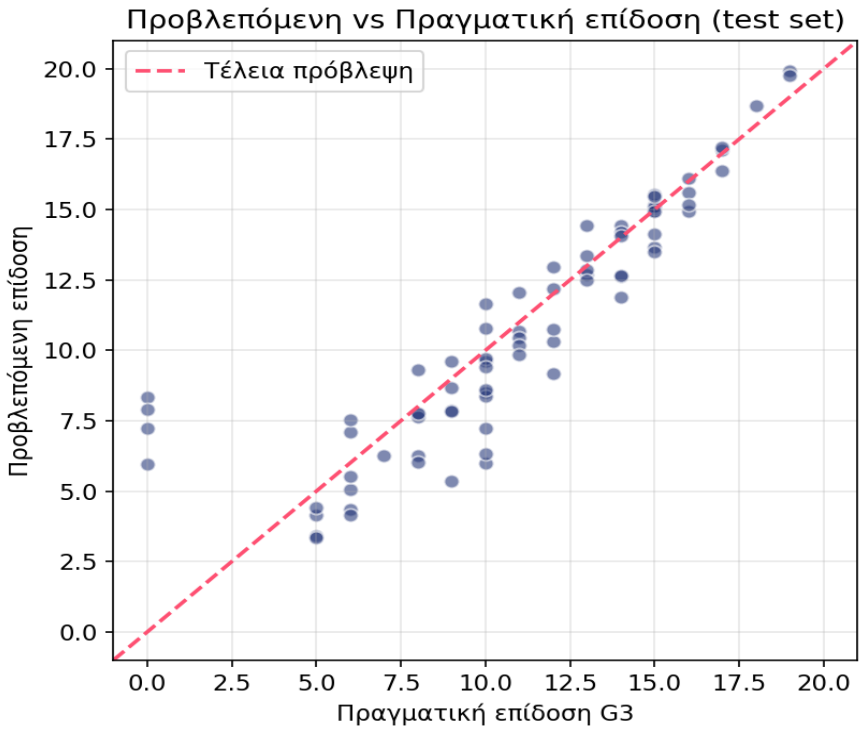
Δείκτης	Σύνολο εκπαίδευσης	Σύνολο ελέγχου
MSE	3,402	4,466
RMSE	1,844	2,113
MAE	1,127	1,340
R^2	0,838	0,782

Πίνακας 5: Τέσσερις δείκτες αξιολόγησης του μοντέλου.

Πηγή: Ίδια επεξεργασία.

Στο σύνολο ελέγχου το μοντέλο επιτυγχάνει RMSE ≈ 2,11 βαθμών (σε κλίμακα 0–20) που σημαίνει ότι, κατά μέσο όρο, οι προβλέψεις του απέχουν κατά 2 μονάδες από τις πραγματικές τιμές και $R^2 = 0,782$, ερμηνεύοντας περίπου το 78% της διακύμανσης του τελικού βαθμού. Για σύγκριση, ένα τετριμμένο μοντέλο που προβλέπει πάντα τον μέσο όρο πετυχαίνει RMSE ≈ 4,55· επομένως ο εκπαιδευμένος νευρώνας περίπου υποδιπλασιάζει το σφάλμα, γεγονός που τεκμηριώνει την αξία της εκπαίδευσης μέσω Gradient Descent. Το διάγραμμα διασποράς προβλεπόμενης–

πραγματικής επίδοσης (Σχήμα 4.3) δείχνει τα σημεία συγκεντρωμένα γύρω από τη διαγώνιο της τέλειας πρόβλεψης.



ΣΧΗΜΑ 9: . ***Προβλεπόμενη έναντι πραγματικής επίδοσης στο σύνολο ελέγχου· η διακεκομμένη γραμμή αντιστοιχεί στην τέλεια πρόβλεψη.***

Πηγή: Ίδια επεξεργασία.

Πιο αναλυτικά με τη βοήθεια του διαγράμματος γίνεται η σύγκριση μεταξύ της πραγματικής και της προβλεπόμενης βαθμολογίας των μαθητών στο σύνολο ελέγχου (79 μαθητές). Στον οριζόντιο άξονα απεικονίζεται η πραγματική επίδοση (G3), ενώ στον κατακόρυφο η προβλεπόμενη επίδοση. Κάθε σημείο αντιστοιχεί σε έναν μαθητή ,ενώ η διακεκομμένη κόκκινη γραμμή αντιπροσωπεύει την ιδανική περίπτωση, κατά την οποία η προβλεπόμενη τιμή είναι ίση με την πραγματική. Όσο πιο κοντά βρίσκονται τα σημεία στη διαγώνιο αυτή, τόσο μικρότερο είναι το σφάλμα πρόβλεψης και τόσο μεγαλύτερη είναι η ακρίβεια του μοντέλου. Σημεία που βρίσκονται πάνω από τη διαγώνιο υποδηλώνουν ότι το μοντέλο υπερεκτίμησε τη βαθμολογία του μαθητή, ενώ σημεία κάτω από αυτήν δείχνουν υποεκτίμηση της πραγματικής επίδοσης. Παρατηρείται ότι η πλειονότητα των σημείων συγκεντρώνεται κοντά στη διαγώνιο, γεγονός που υποδηλώνει ότι το perceptron πραγματοποιεί γενικά ακριβείς προβλέψεις. Αν και υπάρχουν ορισμένες

μεγαλύτερες αποκλίσεις για λίγους μαθητές, αυτές αποτελούν μεμονωμένες περιπτώσεις και δεν επηρεάζουν τη συνολική εικόνα. Η οπτική αυτή απεικόνιση επιβεβαιώνει τα αποτελέσματα των δεικτών αξιολόγησης (RMSE και R^2), δείχνοντας ότι το μοντέλο έχει μάθει ικανοποιητικά τη σχέση μεταξύ των χαρακτηριστικών εισόδου και της τελικής βαθμολογίας και παρουσιάζει καλή ικανότητα γενίκευσης σε νέα δεδομένα.

4.3 Εφαρμογή, αποτελέσματα και αξιολόγηση του έξυπνου βοηθού μελέτης.

Με το μοντέλο εκπαιδευμένο, η τελική εφαρμογή δέχεται τα δεδομένα ενός μαθητή, τα τυποποιεί με τα στατιστικά εκπαίδευσης, εκτιμά την αναμενόμενη επίδοση και παράγει συστάσεις. Η εκτίμηση περιορίζεται στο διάστημα 0–20, ώστε να παραμένει εντός της έγκυρης κλίμακας βαθμολογίας.

Λογική των συστάσεων

- Επίπεδο ασκήσεων: για εκτιμώμενη επίδοση κάτω του 10 προτείνονται εύκολες ασκήσεις (εδραίωση βάσεων), για 10 έως 15 μεσαίες και για 15 και άνω δύσκολες ασκήσεις εμβάθυνσης.
- Επάρκεια προετοιμασίας: η εκτιμώμενη επίδοση συγκρίνεται με τον επιθυμητό στόχο του χρήστη και κρίνεται αν η τρέχουσα προετοιμασία επαρκεί.
- Πρόταση μελέτης: όταν η εκτίμηση υπολείπεται του στόχου, διατυπώνεται σύσταση αύξησης των ωρών μελέτης και της εξάσκησης και μείωσης των απουσιών.

```
def smart_study_assistant(studytime, G1, G2, failures,
                          absences, target_grade):
  x = np.array([studytime, G1, G2, failures, absences], float)
  pred = float(predict((x - mu)/sigma, w, b))
  pred = min(20.0, max(0.0, pred))
  ...  # επίπεδο ασκήσεων, επάρκεια, πρόταση
```

- Δημιουργεί μια συνάρτηση που δέχεται τα στοιχεία του μαθητή και τον επιθυμητό στόχο, ώστε να προβλέψει τον τελικό βαθμό και να δώσει εξατομικευμένες συστάσεις.

- Δημιουργεί έναν αριθμητικό πίνακα (διάνυσμα) που θα χρησιμοποιηθεί ως είσοδος του perceptron. Με την εντολή float μετατρέπει όλες τις τιμές του πίνακα σε δεκαδικούς αριθμούς για μεγαλύτερη ακρίβεια στους υπολογισμούς.
- Κανονικοποιεί τα δεδομένα ώστε να έχουν την ίδια μορφή με εκείνη που χρησιμοποιήθηκε κατά την εκπαίδευση του μοντέλου.
- Εξασφαλίζει ότι η πρόβλεψη δεν μπορεί να πάρει τιμή μικρότερη από το 0 και μεγαλύτερη από 20.

Παραδείγματα εκτέλεσης

Δοκιμάστηκαν τρία αντιπροσωπευτικά προφίλ μαθητών. Τα αποτελέσματα παρουσιάζονται στον Πίνακα 4.4, σύμφωνα με τον οποίο όσο χαμηλότερη η προηγούμενη επίδοση, τόσο χαμηλότερη η εκτίμηση και απλούστερες οι προτεινόμενες ασκήσεις.

Προφίλ	Είσοδοι (studytime, G1, G2, failures, absences· στόχος)	Εκτίμηση	Ασκήσεις	Επάρκεια
Αδύναμος	1, 7, 6, 2, 12 · στόχος 12	4,8/20	Εύκολες	Όχι
Μέσος	2, 11, 11, 0, 4 · στόχος 14	10,8/20	Μεσαίες	Όχι
Δυνατός	4, 16, 17, 0, 2 · στόχος 16	17,1/20	Δύσκολες	Ναι

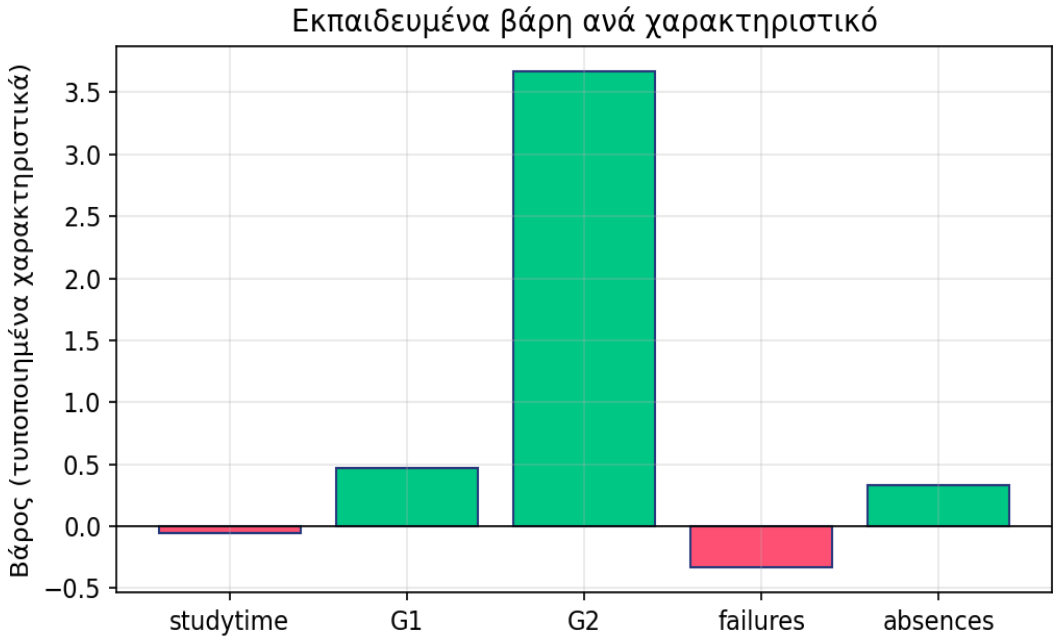
Πίνακας 6:Αποτελέσματα ενδεικτικού παραδείγματος.

Πηγή: Ίδια επεξεργασία.

Σχολιασμός των αποτελεσμάτων

Η ανάλυση των εκπαιδευμένων βαρών (Σχήμα 4.4) επιβεβαιώνει ότι η πρόσφατη προηγούμενη επίδοση (G2) και, δευτερευόντως, η G1 είναι οι κυρίαρχοι προγνωστικοί παράγοντες, ενώ οι οι προηγούμενες αποτυχίες επηρεάζουν αρνητικά την πρόβλεψη του μοντέλου, καθώς όσο αυξάνονται τόσο μειώνεται η εκτιμώμενη

τελική βαθμολογία. Συνολικά, η εφαρμογή επιδεικνύει με σαφήνεια ολόκληρη την αλυσίδα της εκπαίδευσης ενός νευρωνικού μοντέλου μέσω Gradient Descent, από την αρχικοποίηση των παραμέτρων έως την παραγωγή χρήσιμων, ερμηνεύσιμων προβλέψεων.



ΣΧΗΜΑ 10: ***Εκπαιδευμένα βάρη ανά χαρακτηριστικό (σε τυποποιημένη κλίμακα).***

Πηγή:Ίδια επεξεργασία.

Περιορισμοί της εφαρμογής

- Γραμμικότητα: ο μονοεπίπεδος νευρώνας μοντελοποιεί μόνο γραμμικές σχέσεις και όχι στις σύνθετες, μη γραμμικές αλληλεπιδράσεις μεταξύ των χαρακτηριστικών.
- Αμελητέα συνεισφορά της μελέτης: στο συγκεκριμένο σύνολο δεδομένων, παρουσία των βαθμών G1 και G2, το χαρακτηριστικό studytime απέκτησε αμελητέο (ελαφρώς αρνητικό) βάρος. Αυτό σημαίνει ότι η σύσταση για αύξηση των ωρών μελέτης διατυπώνεται ως γενική παιδαγωγική καθοδήγηση και δεν θεμελιώνεται ποσοτικά ή αιτιακά από το μοντέλο, καθώς οι προηγούμενοι βαθμοί απορροφούν το μεγαλύτερο μέρος του προγνωστικού σήματος.
- Ισχυρή συσχέτιση στόχου: η υψηλή συσχέτιση των G1, G2 με τον G3 καθιστά την πρόβλεψη ευκολότερη αλλά λιγότερο ενημερωτική για τους υπόλοιπους

παράγοντες. Πιο αναλυτικά, το perceptron μπορεί να προβλέψει εύκολα τον G3 βασιζόμενο κυρίως στους δύο προηγούμενους βαθμούς γεγονός που αυξάνει την ακρίβεια της πρόβλεψης, αλλά μειώνει τη συμμετοχή των υπόλοιπων χαρακτηριστικών, όπως οι ώρες μελέτης, οι απουσίες και οι προηγούμενες αποτυχίες, επειδή περιέχουν λιγότερη νέα πληροφορία σε σχέση με τους ήδη γνωστούς βαθμούς.

- Μέγεθος και προέλευση δεδομένων: το δείγμα των 395 μαθητών προέρχεται από δύο σχολεία μίας χώρας, γεγονός που περιορίζει τη γενίκευση των ευρημάτων.

Πιθανές μελλοντικές βελτιώσεις

- Επέκταση σε πολυεπίπεδο δίκτυο (MLP) με μη γραμμικές ενεργοποιήσεις, ώστε να αποτυπωθούν σύνθετες σχέσεις.
- Ενσωμάτωση πραγματικών δεικτών εξάσκησης (αριθμός λυμένων ασκήσεων, ποσοστό επιτυχίας) μέσω εμπλουτισμού των δεδομένων.
- Χρήση παραλλαγών του Gradient Descent (Stochastic, Mini-batch) και τεχνικών κανονικοποίησης/τακτικοποίησης για βελτίωση της γενίκευσης.
- Διασταυρούμενη επικύρωση (cross-validation) για πιο αξιόπιστη εκτίμηση της απόδοσης.

Σύνοψη του κεφαλαίου

Στο κεφάλαιο αυτό υλοποιήθηκε σε Python ένας μονοεπίπεδος νευρώνας ο οποίος εκπαιδεύτηκε με χειροκίνητη εφαρμογή του Gradient Descent σε πραγματικά εκπαιδευτικά δεδομένα. Παρουσιάστηκε ολόκληρη η διαδικασία από την επιλογή και προεπεξεργασία των χαρακτηριστικών έως την εκπαίδευση, την αξιολόγηση και την ενσωμάτωση του μοντέλου σε μια λειτουργική εφαρμογή συστάσεων. Με τον τρόπο αυτό επιβεβαιώθηκε στην πράξη η θεωρία των προηγούμενων κεφαλαίων, καθώς φάνηκε πώς η σταδιακή ελαχιστοποίηση της συνάρτησης κόστους οδηγεί στη βελτίωση των προβλέψεων. Παρά τους περιορισμούς του απλού μοντέλου, η εφαρμογή κατέδειξε ότι ακόμη και ένας μονοεπίπεδος νευρώνας μπορεί να αποτελέσει χρήσιμο προγνωστικό εργαλείο, όταν εκπαιδευτεί κατάλληλα με τη μέθοδο Gradient Descent.

Βιβλιογραφία

[1] Turing, Alan M., "Computing Machinery and Intelligence," *Mind*, vol. 59, no. 236, pp. 433–460, 1950, doi: 10.1093/mind/LIX.236.433.

[2] McCarthy, John, Minsky, Marvin L., Rochester, Nathaniel, and Shannon, Claude E., "A Proposal for the Dartmouth Summer Research Project on Artificial Intelligence," Dartmouth College, Hanover, New Hampshire, USA, 1955.

[3] A. L. Samuel, "Some Studies in Machine Learning Using the Game of Checkers," *IBM J. Res. Dev.*, vol. 3, no. 3, pp. 210–229, Jul. 1959, doi: 10.1147/rd.33.0210.

[4] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain.," *Psychol. Rev.*, vol. 65, no. 6, pp. 386–408, 1958, doi: 10.1037/h0042519.

[5] S. J. Russell and P. Norvig, *Artificial intelligence: a modern approach*, Fourth Edition. in Pearson Series in Artificial Intelligence. Hoboken, NJ: Pearson, 2021.

[6] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015, doi: 10.1038/nature14539.

[7] T. M. Mitchell, *Machine learning*, Nachdr. in McGraw-Hill series in Computer Science. New York: McGraw-Hill, 2013.

[8] J. Han, M. Kamber, and J. Pei, *Data mining: concepts and techniques*, 3rd ed. in Morgan Kaufmann series in data management systems. Amsterdam Boston: Elsevier/Morgan Kaufmann, 2012.

[9] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. in Adaptive computation and machine learning. Cambridge, Mass: The MIT press, 2016.

[10] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bull. Math. Biophys.*, vol. 5, no. 4, pp. 115–133, Dec. 1943, doi: 10.1007/BF02478259.

[11] G. Strang, *Introduction to linear algebra*, 5th edition. Wellesley: Cambridge press, 2016.

[12] C. M. Bishop, *Pattern recognition and machine learning*. in Information science and statistics. New York: Springer, 2006.

[13] S. S. Haykin and S. S. Haykin, *Neural networks and learning machines*, 3rd ed. New York: Prentice Hall, 2009.

[14] S. Ruder, "An overview of gradient descent optimization algorithms," Jun. 15, 2017, *arXiv*: arXiv:1609.04747. doi: 10.48550/arXiv.1609.04747.

[15] C. R. Harris *et al.*, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020, doi: 10.1038/s41586-020-2649-2.

[16] Pedregosa, Fabian, "Scikit-learn: Machine Learning in Python," vol. 12, Journal of Machine Learning Research, 2011.

[17] McKinney, Wes, "Data Structures for Statistical Computing in Python."

[18] J. D. Hunter, "Matplotlib: A 2D Graphics Environment," *Comput. Sci. Eng.*, vol. 9, no. 3, pp. 90–95, 2007, doi: 10.1109/MCSE.2007.55.