ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ

ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ
ΤΜΗΜΑ ΨΗΦΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Μηχανική Μάθηση και Εφαρμογές με Python

Παναγιώτης Γ. Σταθόπουλος

Επιβλέπων Καθηγητής:
Μιχαήλ Φιλιππάκης, Αναπληρωτής Καθηγητής

ΠΕΙΡΑΙΑΣ

ΟΚΤΩΒΡΙΟΣ 2021

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Μηχανική Μάθηση και Εφαρμογές με Python

Παναγιώτης Σταθόπουλος
Α.Μ.: Ε14176

ΠΕΡΙΛΗΨΗ

Σκοπός της παρούσας διατριβής είναι να παρουσιαστούν, να εξεταστούν και να συγκριθούν ως προς την αποτελεσματικότητα διάφοροι Αλγόριθμοι Ταξινόμησης που χρησιμοποιούνται ως Μοντέλα στην Μηχανική Μάθηση με την βοήθεια της γλώσσας προγραμματισμού Python. Το πρόβλημα που θα εξετάσουμε έχει να κάνει με την επιβίωση ή όχι ενός επιβάτη του Τιτανικού.

Το πρώτο κεφάλαιο της εργασίας αναφέρεται στο ιστορικό υπόβαθρο και σε εισαγωγικούς ορισμούς.

Στο δεύτερο κεφάλαιο θα μελετήσουμε την Κατηγοριοποίηση των Αλγορίθμων αυτών. Θα δούμε τις κύριες κατηγορίες και θα παρατεθεί εκτενής θεωρητική ανάλυση.

Εν συνεχεία, το επόμενο κεφάλαιο αναφέρεται στα Είδη της Μηχανικής Μάθησης, όπου κύριο κριτήριο επιλογής είναι η μορφολογία του Συνόλου Δεδομένων, καθώς και το τελικό αποτέλεσμα που αναζητούμε.

Στο τέταρτο κεφάλαιο, αναγράφεται το θεωρητικό υπόβαθρο των Μοντέλων – Αλγόριθμων πρόβλεψης, όπου και είναι το «κέντρο αποφάσεων» της Μηχανικής Μάθησης. Εκεί παρουσιάζονται μερικοί από τους δημοφιλέστερους εξ' αυτών. Ένα Μοντέλο προπονείται και έπειτα αφού αποκτήσει την κατάλληλη εμπειρία, μέσω της «κριτικής του σκέψης» για το εκάστοτε θέμα αποφασίζει και μας δίνει ένα αποτέλεσμα.

Έπειτα, στο πέμπτο κεφάλαιο θα δούμε μερικές από τις επικρατέστερες Μετρικές Αξιολόγησης των αλγορίθμων αυτών, διότι σκοπός των αλγορίθμων τελικά είναι να παράγουν όσο το δυνατόν το πιο αξιόπιστο αποτέλεσμα, δηλαδή να έχουν υψηλή Ακρίβεια.

Τελικά, θα παρουσιαστεί μια υλοποίηση Μηχανικής Μάθησης σε Python για το προαναφερθέν πρόβλημα του Τιτανικού. Θα γίνει εκτενείς ανάλυση παρουσίαση και μορφοποίηση των δεδομένων ώστε να χρησιμοποιηθεί πλήθος Μοντέλων αποτελεσματικά και τελικά να εξετάσουμε τις επιδόσεις τους βασισμένοι στην Ακρίβεια.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: Μηχανική μάθηση

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: Ανάλυση Δεδομένων, Αλγόριθμοι Ταξινόμησης, Μηχανική Μάθηση, Μοντέλα – Αλγόριθμοι Πρόβλεψης, Σύγκριση

ABSTRACT

The purpose of this thesis is to present, examine and compare in terms of efficiency various Classification Algorithms used as Models in Machine Learning with the help of Python programming language. The problem we will consider has to do with the survival or not of a Titanic passenger.

The first chapter of the paper refers to the historical background and introductory definitions.

In the second chapter we will study the Categorization of these Algorithms. We will look at the main categories and provide an extensive theoretical analysis.

Furthermore, the next chapter refers to the Types of Machine Learning, where the main selection criterion is the morphology of the Dataset, as well as the final result we are looking for.

In the fourth chapter, the theoretical background of the Models – Prediction Algorithms is written, where the “decision center” of Machine Learning is. There are presented some of the most popular ones. A Model is trained and then after gaining the appropriate experience, through its “critical thinking” for each case decides and gives us a result.

Then, in the fifth chapter we will see some of the most prevalent Evaluation Metrics of these algorithms, because the purpose of the algorithms is ultimately to produce the most reliable result possible, that is to have high Accuracy.

Finally, a Python Machine Learning implementation for the aforementioned Titanic problem will be presented. Extensive analysis, presentation and formatting of the data will be done in order to use a multitude of Models effectively and finally to examine their performance based on Accuracy.

SUBJECT AREA: Machine Learning

KEYWORDS: Data Analysis, Classification Algorithms, Machine Learning, Models – Prediction Algorithms, Comparison

Στην οικογένειά μου...

ΕΥΧΑΡΙΣΤΙΕΣ

Στο σημείο αυτό θα ήθελα να ευχαριστήσω όλους εκείνους τους ανθρώπους που συνεισέφεραν ώστε να πραγματοποιηθεί η παρούσα διατριβή.

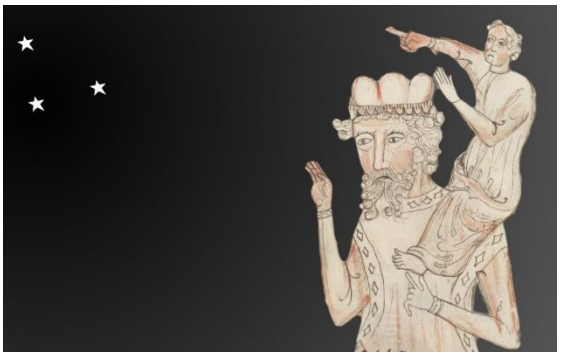
Αρχικά, να ευχαριστήσω τον επιβλέποντα, αναπληρωτή καθηγητή κ. Μ.Φιλιππάκη για την εμπιστοσύνη που μου έδειξε αναθέτοντάς μου την εργασία αυτή, καθώς επίσης και για την ευχέρεια που μου έδωσε ώστε να δομήσω το περιεχόμενο σύμφωνα με τις επιθυμίες μου. Η καθοδήγηση και η βοήθεια του στάθηκαν ως βασικοί πυλώνες καθ' όλη την διάρκεια της εκπόνησης όπως και στην επιτυχή διεκπεραίωση αυτής.

Παράλληλα, θα ήθελα να ευχαριστήσω και όλα τα υπόλοιπα Μέλη Διδακτικού και Ερευνητικού Προσωπικού (Δ.Ε.Π) του τμήματος για τις πολύτιμες γνώσεις που αποκόμισα στην επιστήμη της Πληροφορικής τα χρόνια της φοίτησης μου στο τμήμα Ψηφιακών Συστημάτων.

Επίσης, εδώ θα ήθελα να ευχαριστήσω ιδιαίτερα τους συναδέλφους και φίλους μου που μοιράστηκα μαζί τους κάθε προβληματισμό μου, χαρά και λύπη, ενώ αυτοί με περίσσεια αποθέματα αγάπης και στοργής με συμβούλεψαν και στάθηκαν στο πλευρό μου.

Τέλος, αλλά όχι με ελάσσονα σημασία, θα ήθελα να ευχαριστήσω από τα βάθη της καρδιάς μου τους γονείς μου, τον αδερφό μου, τον θείο, την θεία μου και τα ξαδέρφια μου που με την αγάπη και την υποστήριξή τους όλα αυτά τα χρόνια με βοήθησαν να ξεπεράσω κάθε εμπόδιο. Οι συζητήσεις όσο και η παρουσία τους σε κάθε δύσκολη στιγμή ήταν καθοριστικής σημασίας οδηγώντας στην ολοκλήρωση αυτής της διατριβής. Δίχως αυτούς τίποτα από τα παρακάτω δεν θα ήταν εφικτό. Η εμπιστοσύνη, η ανιδιοτέλεια τους και οι συμβουλές τους έχουν στιγματίσει ανεξίτηλα τη ζωή μου.

Δανειζόμενος μια φράση του Ισαάκ Νεύτωνα, θα ήθελα να εκφράσω πως:



"…Αν κατάφερα να δω λίγο μακρύτερα,
το κατάφερα στεκούμενος στους ώμους γιγάντων…"

Ισαάκ Νεύτωνας, 1643-1727

_____
_____
_____

ΠΕΡΙΕΧΟΜΕΝΑ

ΚΑΤΑΛΟΓΟΣ ΣΧΗΜΑΤΩΝ

ΚΑΤΑΛΟΓΟΣ ΕΙΚΟΝΩΝ

ΚΑΤΑΛΟΓΟΣ ΠΙΝΑΚΩΝ

ΠΡΟΛΟΓΟΣ

Η παρούσα εργασία αποτελεί το έντυπο παραδοτέο της Πτυχιακής Εργασίας με θέμα τη μελέτη επί της Μηχανικής Μάθησης με εφαρμογές στην Python, τόσο σε θεωρητικό όσο και σε επίπεδο εφαρμογής. Η εργασία αυτή πραγματοποιήθηκε στα πλαίσια της εκπόνησης της πτυχιακής εργασίας του φοιτητή του τμήματος Ψηφιακών Συστημάτων του Πανεπιστημίου Πειραιά, Σταθόπουλου Παναγιώτη, υπό την επίβλεψη του αναπληρωτή καθηγητή του τμήματος, κ. Μ.Φιλιππάκη.

1. Εισαγωγή στην Μηχανική Μάθηση

1.1. Ιστορική αναδρομή

Το επιστημονικό πεδίο της Μηχανικής Μάθησης (Machine Learning) είναι άμεσα συνδεδεμένο με τα αντικείμενο της εξόρυξης γνώσης και της πληροφορικής. Τα τελευταία χρόνια έχει μάλιστα παρατηρηθεί μία μεταστροφή προς αυτόν τον χώρο, τόσο από τη σκοπιά της εκπαίδευσης σε φοιτητές, αλλά και σε ερευνητικό επίπεδο με δεκάδες περιοδικά και συνέδρια να και να δημοσιεύονται και να διεξάγονται, αντίστοιχα, αδιάκοπα. Το ίδιο ισχύει βέβαια και σε εμπορικές και πρακτικές εφαρμογές. Χαρακτηριστικά παραδείγματα είναι οι δεκάδες ισότοποι που προφέρουν online υπηρεσίες στους πιθανούς πελάτες τους με σκοπό τη βελτιστοποίηση των παρεχόμενων υπηρεσιών των τελευταίων ή την πρόβλεψη πιθανών καταστάσεων, όπως σε συστήματα μελέτης και πρόβλεψης μελλοντικών οικονομικών καταστάσεων, όπως η πορεία των μετοχών στις χρηματαγορές ή πιθανή πτώχευση σε διαφόρων ειδών επιχειρήσεις.

Παρά το γεγονός πως τα τελευταία χρόνια, η μηχανική μάθηση αποτελεί ένα ιδιαίτερα ενδιαφέρον αντικείμενο και έχει καταφέρει να τραβήξει το ενδιαφέρον πολλών επιστημόνων και επιχειρηματιών, η ουσιαστική δημιουργία αυτού του τομέα, έγινε περίπου το 1700 και στηρίχθηκε στο θεώρημα των πιθανοτήτων του Bayes [1]. Στη συνέχεια, τα επόμενα αξιοσημείωτα επιστημονικά επιτεύγματα που αξιοποιούνται πλέον από τις εφαρμογές μηχανικής μάθησης σε δεδομένα, ήταν η τεχνική της Παλινδρόμησης (Regression) (1920), τα Νευρωνικά Δίκτυα (Neural Networks) και οι ταξινομητές Κ-Πλησιέστερων Γειτόνων (K Nearest Neighbors) που έκαναν την εμφάνιση τους στα μέσα του 20ου αιώνα. Τα επόμενα χρόνια και πιο συγκεκριμένα στις αρχές του 1960 ξεκινάει και πρακτικά η άνθηση της Μηχανικής Μάθησης, όπου εξαιτίας της τεράστιας αύξησης του όγκου των δεδομένων, με τα αποτελέσματα της ραγδαίας ανάπτυξης της επιστήμης των υπολογιστών και τις νέες αλγοριθμικές ιδέες και τακτικές που εμφανίζονται την εποχή αυτή, μπορούμε να πούμε πως έφτασε στο αποκορύφωμα της.

Ως αποτέλεσμα, οδηγηθήκαμε στην εμφάνιση εξαιρετικών τεχνικών και μεθόδων που αποτελούν ακόμη και σήμερα σε πολλά προβλήματα τις αποδοτικότερες προσεγγίσεις. Οι πιο σημαντικές από αυτές τις μεθόδους ήταν τα Δένδρα Αποφάσεων (Decision Trees), η Συσταδοποίηση (Clustering) και οι Γενετικοί Αλγόριθμοι (Genetic Algorithms) έως τέλη του 1960, καθώς και οι Μηχανές Διανυσματικής Στήριξης (Support Vector Machines) και οι Αποθήκες Δεδομένων (Data Warehouses) γύρω στο 1990.

Μελετώντας κανείς την προηγούμενη αναδρομή παρατηρεί πως ουσιαστικά το πεδίο της Μηχανικής Μάθησης χρησιμοποιεί και επηρεάζεται πρακτικά από μία σειρά από άλλα επιστημονικά πεδία, ευρύτερα αλλά και προγενέστερα από το αυτό. Με μία αυστηρή παρατήρηση αυτών, θα μπορούσαμε να απαριθμήσουμε τα εξής:

- Στατιστική
- Αλγόριθμοι
- Βάσεις Δεδομένων
- Τεχνητή Νοημοσύνη
- Ανάκτηση Πληροφοριών

Παρακάτω παρατίθενται μερικά ορόσημα επιτευγμάτων της Μηχανικής Μάθησης:

- Γέννηση [1952 - 1956] 1950 - Ο Alan Turing δημιουργεί το "Test Turing" για να διαπιστώσει αν μια μηχανή είναι πραγματικά έξυπνη. Για να περάσει τη δοκιμή, το μηχάνημα πρέπει να είναι ικανό να κάνει έναν άνθρωπο να πιστεύει ότι είναι ένας άλλος άνθρωπος αντί ενός υπολογιστή.

- 1952 - Ο Arthur Samuel γράφει το πρώτο πρόγραμμα υπολογιστή ικανό να μάθει. Το λογισμικό ήταν ένα πρόγραμμα που θα μπορούσε να παίξει πούλια(checkers) και να βελτιωθεί με κάθε παιχνίδι που έπαιζε.

- 1956 - Ο Martin Minsky και ο John McCarthy, με τη βοήθεια του Claude Shannon και του Nathan Rochester, πραγματοποίησαν μια διάσκεψη στο Dartmouth το 1956, το οποίο θεωρείται ότι είναι το σημείο όπου γεννήθηκε το πεδίο της Τεχνητής Νοημοσύνης. Ο Μίνσκι έπεισε τους παρευρισκόμενους να υιοθετήσουν τον όρο "Τεχνητή Νοημοσύνη" ως το όνομα για το νέο πεδίο.

- 1958 - Ο Frank Rosenblatt σχεδιάζει το Perceptron, το πρώτο τεχνητό νευρωνικό δίκτυο.

- 1967 - Γράφετε ο αλγόριθμος "Nearest Neighbor". Αυτό το ορόσημο θεωρείται η γέννηση του πεδίου της αναγνώρισης προτύπων στους υπολογιστές.

- Πρώτος Χειμώνας του AI [1974 - 1980] Το δεύτερο μισό της δεκαετίας του 1970, ο τομέας υπέστη τον πρώτο «χειμώνα». Διάφορα ινστιτούτα και επιχειρήσεις που χρηματοδοτούσαν την έρευνα της τεχνητής νοημοσύνης έκοψαν τα κεφάλαια μετά από χρόνια με μεγάλες προσδοκίες και μικρή πραγματική πρόοδο.

- 1979 - Οι φοιτητές του Πανεπιστημίου του Στάνφορντ επινοούν το "Stanford Cart", ένα κινητό ρομπότ ικανό να κινείται αυτόνομα γύρω από μια αίθουσα, αποφεύγοντας τα εμπόδια.

- Η Έκρηξη της δεκαετίας του 1980 [1980 - 1987] Η δεκαετία του '80 είναι γνωστή για την γέννηση των “expert systems” ειδικών συστημάτων, βασισμένων σε κανόνες. Αυτά υιοθετήθηκαν γρήγορα από τον εταιρικό τομέα, δημιουργώντας νέο ενδιαφέρον για τη Μηχανική Μάθηση.

- 1981 - Ο Gerald Dejong εισάγει την έννοια της “Explanation Based Learning”(Εκμάθησης βασισμένης στη μάθηση), στην οποία ένας

υπολογιστής αναλύει τα δεδομένα εκπαίδευσης και δημιουργεί γενικούς κανόνες που επιτρέπουν την απόρριψη των λιγότερο σημαντικών δεδομένων.

- 1985 - Ο Terry Sejnowski εφευρίσκει το NetTalk, το οποίο μαθαίνει να προφέρει λέξεις με τον ίδιο τρόπο που μαθαίνει ένα παιδί.

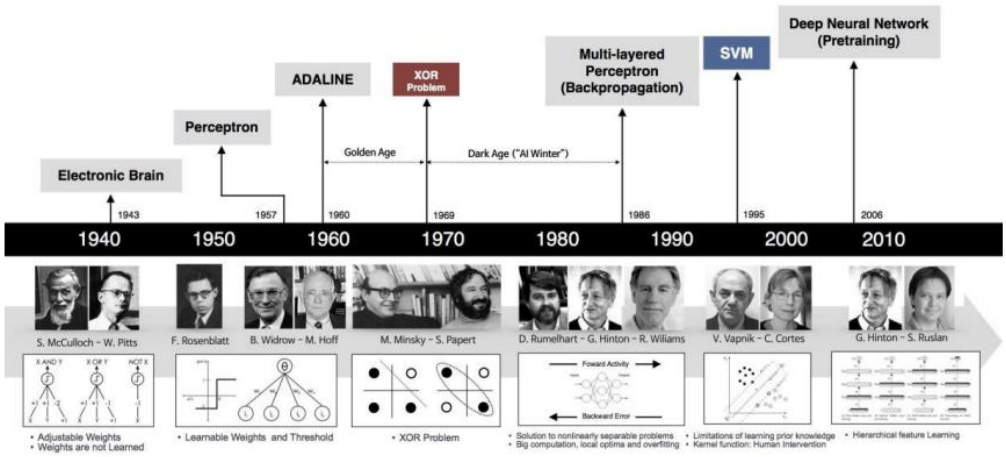
- Δεύτερος AI Winter [1987 - 1993] Στα τέλη της δεκαετίας του '80 και στις αρχές της δεκαετίας του '90, ο τομέας του AI γνώρισε ένα δεύτερο "χειμώνα". Αυτή τη φορά, τα αποτελέσματά του διήρκεσαν για αρκετά χρόνια και η φήμη του τομέα δεν ανακτήθηκε πλήρως μέχρι τις αρχές της δεκαετίας του 2000.

- Η δεκαετία του '90 - Η τεχνολογία της μηχανικής μάθησης μετακινείται από την εστίαση στη ανάπτυξη μοντέλων βασισμένα στην γνώση σε μοντέλα που βασίζονται στην μάθηση μέσω μεγάλων ποσοτήτων δεδομένων. Οι επιστήμονες αρχίζουν να δημιουργούν προγράμματα που αναλύουν μεγάλες ποσότητες δεδομένων και εξαγάγουν συμπεράσματα από τα αποτελέσματα.

- 1997 - Ο υπολογιστής Deep Blue της IBM, νικά τον παγκόσμιο πρωταθλητή σκακιού Gary Kasparov.

- Έκρηξη και εμπορευματοποίηση [2006 - Σήμερα] Η αύξηση της υπολογιστικής δύναμης των υπολογιστών μαζί με τη μεγάλη αφθονία των διαθέσιμων δεδομένων ξαναζωντάνεψαν το πεδίο της Μηχανικής Μάθησης. Πολλές επιχειρήσεις στρέφουν τις εταιρείες τους προς την συλλογή δεδομένων και ενσωματώνουν τη Μηχανική Μάθηση στις διαδικασίες, τα προϊόντα και τις υπηρεσίες τους, προκειμένου να αποκτήσουν πλεονέκτημα έναντι του ανταγωνισμού τους.

- 2006 - Ο Geoffrey Hinton εφηύρε τη φράση Βαθιά Μάθηση (Deep Learning) για να εξηγήσει τις νέες αρχιτεκτονικές των βαθιών νευρωνικών δικτύων ικανών να μάθουν πολύ καλύτερα μοντέλα.



Εικόνα 1: Ιστορική αναδρομή

- 2011 - Ο υπολογιστής Watson από την IBM νικά τους ανθρώπινους ανταγωνιστές στο Jeopardy, ένα τηλεοπτικό παιχνίδι που αποτελείται από απαντήσεις σε ερωτήσεις στη φυσική γλώσσα.

- 2012 - Ο Jeff Dean, με την βοήθεια του Andrew Ng (Πανεπιστήμιο του Στάνφορντ), ηγείται του GoogleBrain, το οποίο ανέπτυξε ένα βαθύ νευρωνικό δίκτυο χρησιμοποιώντας όλη την ικανότητα της υποδομής της Google για να ανιχνεύσει μοτίβα σε βίντεο και εικόνες.

- 2012 - Ο Geoffrey Hinton οδηγεί τη νικήτρια ομάδα στον διαγωνισμό Computer Vision στο Imagenet χρησιμοποιώντας ένα βαθύ νευρωνικό δίκτυο. Η ομάδα κέρδισε με μεγάλο περιθώριο, προκαλώντας την τρέχουσα έκρηξη της Μηχανικής Μάθησης με βάση τα Deep Neural Networks.

- 2012 - Το ερευνητικό εργαστήριο Google X χρησιμοποιεί το GoogleBrain για να αναλύει αυτόνομα τα βίντεο του YouTube και να ανιχνεύει αυτά που περιέχουν γάτες.

- 2014 - Το Facebook αναπτύσσει το DeepFace, έναν αλγόριθμο που βασίζεται σε DNN ικανά να αναγνωρίσουν ανθρώπους με την ίδια ακρίβεια με τον άνθρωπο.

- 2014 - Το Google αγοράζει το DeepMind, ένα βρετανικό start-up που ειδικεύεται στο deep learning, το οποίο είχε πρόσφατα καταδείξει τις δυνατότητες του με έναν αλγόριθμο ικανό να παίζει παιχνίδια Atari απλά βλέποντας τα pixel στην οθόνη, όπως θα έκανε και ο άνθρωπος. Ο αλγόριθμος, μετά από ώρες εκπαίδευσης, ήταν ικανός να κερδίσει ανθρώπους εμπειρογνώμονες στα παιχνίδια.

- 2015 - Η Amazon εγκαινιάζει τη δική της πλατφόρμα εκμάθησης μηχανών.

- 2015 - Η Microsoft δημιουργεί το “Distributed Machine Learning Toolkit”, το οποίο επιτρέπει την αποτελεσματική κατανομή των προβλημάτων μηχανικής μάθησης σε πολλούς υπολογιστές.

- 2015 - Ο Elon Musk και ο Sam Altman, μεταξύ άλλων, ιδρύουν τον μη κερδοσκοπικό οργανισμό OpenAI, παρέχοντάς του ένα δισεκατομμύριο δολάρια με στόχο να εξασφαλίσει ότι η τεχνητή νοημοσύνη έχει θετικό αντίκτυπο στην ανθρωπότητα.

- 2016 - Το Google DeepMind με το μοντέλο AlphaGo νικά τον επαγγελματία παίχτη του παιχνιδιού “Go” Lee Sedol 5 - 1 , το οποίο παιχνίδι θεωρείται ένα από τα πιο πολύπλοκα επιτραπέζια παιχνίδια. Οι επαγγελματίες παίκτες “Go” επιβεβαίωσαν ότι ο αλγόριθμος ήταν σε θέση να κάνει "δημιουργικές" κινήσεις που δεν είχαν ξαναδεί.

- 2017 - Η εταιρία Waymo βγάζει στην παραγωγή τα πρώτα πραγματικά αυτόνομα αυτοκίνητα με πραγματικούς αναβάτες χωρίς να υπάρχει ανθρώπινος χειριστής στο τιμόνι.

- 2018 – Η Airbus και η IBM στέλνει στο διάστημα το πρώτο ρομπότ τεχνητής νοημοσύνης. Το CIMON είναι ένα διαδραστικό ρομπότ με πλήρης λειτουργία φωνής και ο σκοπός του είναι να μειώσει το στρες των αστροναυτών.

- 2019 – Η Textron Systems λανσάρει το Ripsaw M5 Autonomous Battle Tank. Ένα μη επανδρωμένο άρμα μάχης, ηλεκτροκινούμενο με συστήματα τεχνητής νοημοσύνης.

- 2020 – Το Πανεπιστήμιο της Οξφόρδης σχεδίασε πρόγραμμα βασισμένο στη μηχανική μάθηση που είναι ικανό να διαχωρίζει περιστατικά μεταξύ COVID-19 και άλλων ασθενειών του αναπνευστικού. [2]

1.2 Η έννοια της Μηχανικής Μάθησης

Για την επίλυση ενός προβλήματος με τη χρήση υπολογιστή χρειαζόμαστε έναν αλγόριθμο, μια πεπερασμένη αλληλουχία βημάτων τα οποία θα πρέπει να εφαρμοστούν προκειμένου τα δεδομένα εισόδου να μετασχηματιστούν στην επιθυμητή έξοδο. Ας υποθέσουμε για παράδειγμα ότι θέλουμε να διατάξουμε σε αύξουσα σειρά ένα πλήθος αριθμών. Η είσοδος του προβλήματος είναι το σύνολο των αριθμών και η έξοδος είναι η διάταξη αυτών των αριθμών σε αύξουσα σειρά. Για την επίλυση του συγκεκριμένου προβλήματος μπορούν να εφαρμοστούν αρκετοί αλγόριθμοι, αλλά ίσως τελικά χρειαζόμαστε τον πιο αποδοτικό αλγόριθμο τόσο σε ταχύτητα, όσο και σε κατανάλωση μνήμης.

Υπάρχουν ωστόσο προβλήματα, στα οποία δεν είναι διαθέσιμος κάποιος αλγόριθμος, όπως για παράδειγμα στην περίπτωση που θέλουμε να διακρίνουμε την επιβίωση ή τον θάνατο ενός επιβάτη. Στην περίπτωση αυτή έχουμε ως είσοδο ένα πλήθος επιβατών και γνωρίζουμε το αποτέλεσμα της εξόδου: επέζησε/δεν επέζησε. Αυτό που δεν γνωρίζουμε είναι ο τρόπος (αλγόριθμος) σύμφωνα με τον οποίο τα δεδομένα εισόδου θα μετασχηματιστούν στην έξοδο. Στην πραγματικότητα, δεν γνωρίζουμε τον τρόπο με τον οποίο θα χαρακτηρίσουμε έναν επιβάτη ως επιζών ή μη επιζών σε κάποιο ατύχημα.

Με την εξέλιξη στην επιστήμη και τεχνολογία των υπολογιστών, η έλλειψη γνώσης μπορεί να αντισταθμιστεί από την ύπαρξη πληθώρας δεδομένων. Αν και δεν είμαστε σε θέση να κατανοήσουμε πλήρως τη διεργασία πίσω από αυτό τον μετασχηματισμό, μπορούμε να καταλήξουμε συχνά σε μια πολύ ακριβή και χρήσιμη προσέγγιση εντοπίζοντας συγκεκριμένα πρότυπα. Αυτό χαρακτηρίζει τελικά αυτό που αναφέρεται ως Μηχανική Μάθηση [3].

Για τον προσδιορισμό του όρου «Μηχανική Μάθηση» έχουν δοθεί αρκετοί ορισμοί, όπως για παράδειγμα:

> *«Μηχανική Μάθηση είναι η μελέτη υπολογιστικών μεθόδων για την απόκτηση νέας γνώσης, νέων δεξιοτήτων και νέων τρόπων οργάνωσης της υπάρχουσας γνώσης»* [4].

> *«Ένα πρόγραμμα υπολογιστή θεωρείται ότι μαθαίνει από την εμπειρία E ως προς κάποια κλάση εργασιών T και μέτρο απόδοσης P, αν η απόδοσή του σε εργασίες από το T, όπως μετριέται από το P, βελτιώνεται μέσω της εμπειρίας E»* [5].

> *«Κάτι μαθαίνει όταν αλλάζει τη συμπεριφορά του κατά τέτοιο τρόπο ώστε να αποδίδει καλύτερα στο μέλλον»* [6].

Η Μηχανική Μάθηση δεν είναι ένα απλό πρόβλημα διαχείρισης μεγάλου όγκου δεδομένων. Αποτελεί μέρος της Τεχνητής Νοημοσύνης (Artificial Intelligence). Για να είναι νοήμων ένα σύστημα, θα πρέπει να έχει την ικανότητα να μαθαίνει μέσα σε ένα διαρκώς μεταβαλλόμενο περιβάλλον. Εάν το σύστημα έχει την ικανότητα να μαθαίνει και να προσαρμόζεται σε αυτές τις μεταβολές, τότε ο σχεδιαστής του συστήματος δεν χρειάζεται να προβλέπει και να παρέχει λύσεις για όλες τις πιθανές καταστάσεις.

Σύμφωνα με τα παραπάνω, η Μηχανική Μάθηση είναι η εκτέλεση ενός προγράμματος με τη χρήση υπολογιστή με σκοπό τη μεγιστοποίηση των παραμέτρων ενός μοντέλου χρησιμοποιώντας δεδομένα εκπαίδευσης ή προηγούμενη εμπειρία [3]. Το μοντέλο μπορεί να χρησιμοποιηθεί για πρόβλεψη στο μέλλον ή/και για περιγραφή των δεδομένων του προβλήματος. Ο ρόλος της επιστήμης της Πληροφορικής στη Μηχανική Μάθηση είναι διττός:

- Κατά τη διάρκεια της εκπαίδευσης, χρειάζονται αποτελεσματικοί αλγόριθμοι για την μεγιστοποίηση των παραμέτρων που σχετίζονται με το πρόβλημα, καθώς και η απαραίτητη τεχνολογία για την αποθήκευση και επεξεργασία μεγάλου όγκου δεδομένων.

- Το μοντέλο θα πρέπει να είναι αρκετά αποτελεσματικό όσον αφορά την ακρίβεια και τον χρόνο της πρόβλεψης.

2. Κατηγοριοποίηση Αλγορίθμων

Οι αλγόριθμοι Μηχανικής Μάθησης, συνήθως, έχουν ως στόχο την προσαρμογή ενός μοντέλου στα δεδομένα πλησιέστερο στα χαρακτηριστικά που εξετάζονται. Ένας αλγόριθμος έχει συνήθως τρία μέρη, το μοντέλο, την προτίμηση και την αναζήτηση. Αρχικά, ο σκοπός του αλγορίθμου είναι η προσαρμογή του μοντέλου στα δεδομένα. Στο δεύτερο μέρος πρέπει να περιλαμβάνει κάποια κριτήρια σύμφωνα με τα οποία το μοντέλο αυτό να προτιμάται από ένα άλλο. Τέλος, ο αλγόριθμος πρέπει να εμπεριέχει μία τεχνική αναζήτησης των δεδομένων.

Ένα μοντέλο που συνήθως μπορεί να χαρακτηριστεί ως περιγραφικό ή προβλεπτικό. Το περιγραφικό μοντέλο αναγνωρίζει πρότυπα ή συσχετίσεις στα δεδομένα, και δρα ως μέσο διερεύνησης των ιδιοτήτων των δεδομένων υπό εξέταση και δεν προβλέπει νέες ιδιότητες. Ενώ το προβλεπτικό μοντέλο κάνει μία πρόβλεψη για τις τιμές των δεδομένων με τη χρήση των ιστορικών δεδομένων και αποτελεσμάτων [7].

Πίνακας 1: Κατηγοριοποίηση αλγορίθμων

Προβλεπτικά Μοντέλα	Περιγραφικά Μοντέλα
Κατηγοριοποίηση	Συσταδοποίηση
Παλινδρόμηση	Κανόνες Συσχετίσεων
Ανάλυση Χρονοσειρών	Ανακάλυψη Ακολουθιών
	Παρουσίαση Συνόψεων

Στα προβλεπτικά μοντέλα, λόγω του ότι προβλέπουν μία τιμή ή κάποιες τιμές, καταχωρούνται οι τεχνικές της κατηγοριοποίησης, της παλινδρόμησης, της ανάλυσης των χρονολογικών σειρών και της πρόβλεψης. Στα περιγραφικά μοντέλα, λόγω της αναγνώρισης των προτύπων, κατατάσσονται οι τεχνικές της συσταδοποίησης, της παρουσίασης συνόψεων, της ανακάλυψης ακολουθιών και των κανόνων συσχετίσεων.

Τα μοντέλα, επίσης, κατηγοριοποιούνται σε παραμετρικά και μη παραμετρικά. Στα παραμετρικά μοντέλα προσδιορίζεται η συσχέτιση που υπάρχει ανάμεσα στην είσοδο και στην έξοδο μέ τη χρήση αλγεβρικών εξισώσεων. Στις εξισώσεις αυτές υπάρχουν παράμετροι, που είναι απροσδιόριστες, και εκτιμώνται μέσω των παραδειγμάτων του συνόλου εκπαίδευσης που δίνονται ως είσοδος. Στη περίπτωση αυτή ένα συγκεκριμένο μοντέλο θεωρείται δεδομένο εκ των προτέρων για αυτό και απαιτείται περισσότερη γνώση από τα δεδομένα πριν ξεκινήσει η διαδικασία της μοντελοποίησης.

Αντίθετα, τα μη παραμετρικά μοντέλα δεν περιέχουν παραμέτρους αλλά καθοδηγούνται από τα δεδομένα. Δηλαδή, δεν υπάρχουν εξισώσεις για το μοντέλο αλλά προσαρμόζεται το μοντέλο μέσω των δεδομένων.

Πιο συγκεκριμένα, οι μη παραμετρικές τεχνικές δημιουργούν ένα μοντέλο που βασίζεται στην είσοδο για αυτό και είναι περισσότερο κατάλληλες για τις εφαρμογές της μηχανικής μάθησης. Οι μη παραμετρικές μέθοδοι περιέχουν τεχνικές μηχανικής μάθησης που έχουν τη δυνατότητα δυναμικής εκμάθησης με την πρόσθεση νέων δεδομένων στην είσοδο. Με αυτόν τον τρόπο, όσο περισσότερα δεδομένα προστίθενται τόσο καλύτερο είναι το μοντέλο που δημιουργείται. Αυτή η διαδικασία της μάθησης επιτρέπει στο μοντέλο να διευρύνεται συνεχώς καθώς εισάγονται νέα δεδομένα. Τα Νευρωνικά Δίκτυα, τα Δένδρα Αποφάσεων και οι Γενετικοί Αλγόριθμοι αποτελούν μερικά από τα παραδείγματα των μη παραμετρικών τεχνικών [7].

2.1 Κατηγοριοποίηση

Η Κατηγοριοποίηση (Classification) αποτελεί μια από τις βασικές τεχνικές μηχανικής μάθησης. Βασίζεται στην εξέταση των χαρακτηριστικών ενός νέου παραδείγματος (instance) το οποίο με βάση τα χαρακτηριστικά αυτά αντιστοιχίζεται σε ένα προκαθορισμένο σύνολο κλάσεων.

Τα παραδείγματα που πρόκειται να κατηγοριοποιηθούν αναπαριστάνονται γενικά από τις εγγραφές της βάσης δεδομένων και η διαδικασία της κατηγοριοποίησης αποτελείται από την ανάθεση κάθε εγγραφής σε κάποιες από τις προκαθορισμένες κλάσεις. Η διαδικασία της κατηγοριοποίησης χαρακτηρίζεται από έναν καλά καθορισμένο ορισμό των κλάσεων και το σύνολο που χρησιμοποιείται για την εκπαίδευση του μοντέλου αποτελείται από προ-κατηγοριοποιημένα παραδείγματα.

Η βασική διαδικασία έχει ως στόχο να δημιουργηθεί ένα μοντέλο το οποίο θα μπορούσε να εφαρμοστεί για να κατηγοριοποιήσει δεδομένα που δεν έχουν ακόμα κατηγοριοποιηθεί. Στις περισσότερες περιπτώσεις, υπάρχει ένας περιορισμένος αριθμός κατηγοριών και εμείς θα πρέπει να αναθέσουμε κάθε παράδειγμα στην κατάλληλη κατηγορία [8].

2.2 Παλινδρόμηση

Η Παλινδρόμηση είναι μια διαδικασία η οποία έχει μελετηθεί πολύ στην στατιστική. Κύριος σκοπός εδώ είναι η πρόβλεψη της τιμής μιας μεταβλητής μελετώντας τις τιμές που είχε στο παρελθόν. Η παλινδρόμηση καλύπτει ένα μεγάλο τμήμα του τομέα μηχανικής μάθησης που έχει να κάνει με προβλέψεις. Ένα χαρακτηριστικό παράδειγμα αλγορίθμου παλινδρόμησης είναι η Γραμμική Παλινδρόμηση (Linear Regression) ή η Λογιστική Παλινδρόμηση (Logistic Regression) [9].

2.3 Συσταδοποίηση

Η Συσταδοποίηση είναι η διαδικασία του καταμερισμού ενός ετερογενούς πληθυσμού σε ένα σύνολο περισσότερων ετερογενών συστάδων (clusters).

Αυτό που διαφοροποιεί τη Συσταδοποίηση από την Κατηγοριοποίηση είναι ότι η Συσταδοποίηση δε βασίζεται σε προκαθορισμένες κατηγορίες. Στην Κατηγοριοποίηση, ο πληθυσμός διαιρείται σε κατηγορίες αναθέτοντας κάθε στοιχείο ή εγγραφή σε μια προκαθορισμένη κατηγορία με βάση ένα μοντέλο που αναπτύσσεται μέσο της εκπαίδευσης του με παραδείγματα που έχουν κατηγοριοποιηθεί εκ των προτέρων. Όπως και στην Κατηγοριοποίηση έτσι και στη Συσταδοποίηση υπάρχουν πολλές εφαρμογές.

Για παράδειγμα, ας υποθέσουμε ότι έχουμε διαθέσιμα τα δεδομένα πελατών μιας εταιρίας πωλήσεων. Χρησιμοποιώντας τεχνικές Συσταδοποίησης, μπορούμε να βρούμε τον καταμερισμό των πελατών και της αγοράς, π.χ. μπορούμε να δούμε ποιοι πελάτες αγοράζουν για την οικογένεια τους και ποιοι για τον εαυτό τους ή ποιοι έχουν μεγάλο εισόδημά και ποιοι όχι [10].

2.4 Κανόνες Συσχέτισης

Η εξαγωγή Κανόνων Συσχέτισης (Association Rules) θεωρείται μια από τις σημαντικότερες διαδικασίες. Έχει προσελκύσει μεγάλο ενδιαφέρον γιατί παρέχει έναν συνοπτικό τρόπο για να εκφραστούν οι ενδεχομένως χρήσιμες πληροφορίες που γίνονται εύκολα κατανοητές από τους τελικούς χρήστες.

Οι Κανόνες Συσχέτισης ανακαλύπτουν κρυμμένες «συσχετίσεις» μεταξύ των γνωρισμάτων ενός συνόλου των δεδομένων. Αυτοί οι συσχετισμοί παρουσιάζονται στην ακόλουθη μορφή $A \rightarrow B$ όπου το *A* και το *B* αναφέρονται στα σύνολα γνωρισμάτων που υπάρχουν στα υπό ανάλυση δεδομένα [11].

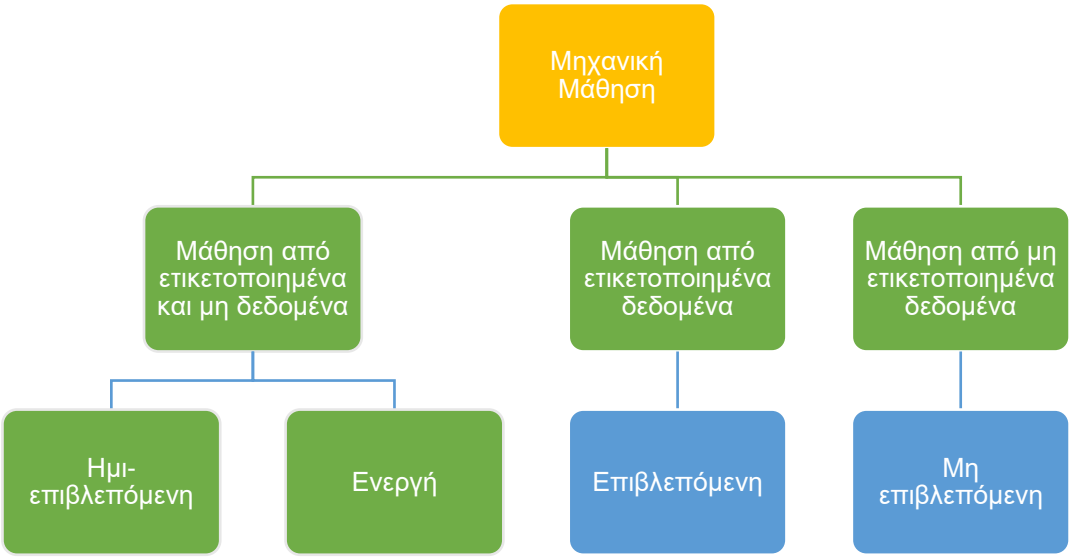
2.5 Ανάλυση Χρονολογικών Σειρών

Με την Ανάλυση Χρονολογικών Σειρών (Time Series Analysis), μελετάται η τιμή ενός χαρακτηριστικού καθώς μεταβάλλεται στο χρόνο. Οι τιμές συνήθως λαμβάνονται σε ίσα χρονικά διαστήματα (ημερήσια, εβδομαδιαία, ωριαία, κοκ.) και για να παρασταθούν οπτικά οι χρονοσειρές χρησιμοποιείται το διάγραμμα χρονοσειρών. Υπάρχουν τρεις βασικές διαδικασίες που διενεργούνται στην Ανάλυση Χρονοσειρών.
Στη μία περίπτωση, χρησιμοποιούνται μονάδες μέτρησης απόστασης για να καθορίσουν την ομοιότητα ανάμεσα σε διαφορετικές χρονοσειρές και στη δεύτερη περίπτωση, εξετάζεται η δομή της χρονοσειρές για να καθορίσει (και ίσως να κατηγοριοποιήσει) τη συμπεριφορά της. Συχνή είναι η χρήση διαγραμμάτων χρονοσειρών για την πρόβλεψη μελλοντικών τιμών [12].

3. Είδη Μηχανικής Μάθησης

Η Μηχανική Μάθηση διαχωρίζεται ανάλογα με το είδος των δεδομένων του συνόλου εκπαίδευσης (ετικετοποιημένα ή/και μη ετικετοποιημένα) στις παρακάτω τέσσερις κατηγορίες:

- Επιβλεπόμενη Μάθηση (Supervised Learning)
- Μη Επιβλεπόμενη Μάθηση (Unsupervised Learning)
- Ημι-επιβλεπόμενη Μάθηση (Semi-Supervised Learning)
- Ενεργή Μηχανική Μάθηση (Active Learning)



Σχήμα 1: Είδη Μηχανικής Μάθησης

3.1 Επιβλεπόμενη Μάθηση

Στην Επιβλεπόμενη Μάθηση το σύστημα μάθησης λαμβάνει ένα σύνολο από Δεδομένα Εκπαίδευσης (Training Data), τα οποία αποτελούνται από ζεύγη της μορφής (x_i, y_i), $i = 1, 2, ..., n$. Κάθε ζεύγος αποτελείται από ένα διάνυσμα τιμών x_i των χαρακτηριστικών εισόδου και την αντίστοιχη τιμή για την μεταβλητή απόφασης y_i [13]. Με βάση αυτά τα δεδομένα εκπαίδευσης δημιουργείται ένα μοντέλο κατηγοριοποίησης με σκοπό την πρόβλεψη της τιμής της μεταβλητής απόφασης y σε μελλοντικά δεδομένα x. Επομένως, η δημιουργία του μοντέλου είναι επαγωγική (inductive), ενώ η εφαρμογή του στην πρόβλεψη των τιμών νέων περιπτώσεων είναι συμπερασματική (deductive) [14].

Ανάλογα με τη μεταβλητή απόφασης y, τα προβλήματα Επιβλεπόμενης Μάθησης διαχωρίζεται σε προβλήματα Κατηγοριοποίησης (η μεταβλητή y παίρνει διακριτές τιμές) και προβλήματα Παλινδρόμησης (η μεταβλητή y παίρνει συνεχείς τιμές μέσα σε κάποιο διάστημα πραγματικών αριθμών). Με

βάση τα προαναφερόμενα, η Κατηγοριοποίηση και η Παλινδρόμηση ορίζονται ως εξής:

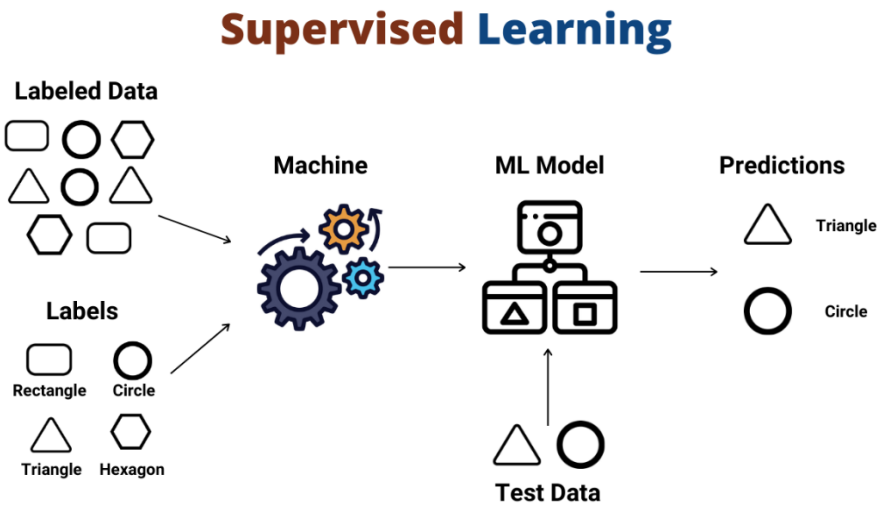
- Κατηγοριοποίηση είναι το πρόβλημα επιβλεπόμενης μάθησης με διακριτές κλάσεις y .

- Παλινδρόμηση είναι το πρόβλημα επιβλεπόμενης μάθησης με συνεχείς κλάσεις y .

Η εκτίμηση της αποτελεσματικότητας-ακρίβειας του μοντέλου σε ένα πρόβλημα Επιβλεπόμενης Μάθησης γίνεται χρησιμοποιώντας ένα ξεχωριστό σύνολο δεδομένων, τα Δεδομένα Δοκιμών (Test Data). Για τα Δεδομένα Δοκιμών είναι γνωστή η κατηγοριοποίηση των περιπτώσεων που περιέχει, όμως δεν χρησιμοποιείται κατά τη διάρκεια της εκπαίδευσης.

Έτσι, η ορθότητα της κατηγοριοποίησης των περιπτώσεων του δείγματος ελέγχου αποτελεί μια καλή εκτίμηση για την αποτελεσματικότητα του μοντέλου. Η διαδικασία της Επιβλεπόμενης Μάθησης φαίνεται στην Εικόνα 2 [15]. Σύμφωνα με την εικόνα, το πρόβλημα της Επιβλεπόμενης Μάθησης αποτελείται από τα εξής βήματα:

- Καθορισμός του προβλήματος.
- Αναγνώριση των δεδομένων του προβλήματος.
- Προ-επεξεργασία των δεδομένων.
- Καθορισμός του συνόλου εκπαίδευσης (ένα σύνολο από εγγραφές των οποίων οι κλάσεις είναι γνωστές).
- Επιλογή του κατάλληλου Ταξινομιτή (Classifier).
- Εκπαίδευση του ταξινομιτή με τη χρήση των δεδομένων του συνόλου εκπαίδευσης για την κατασκευή ενός μοντέλου κατηγοριοποίησης.
- Αξιολόγηση των αποτελεσμάτων στο σύνολο ελέγχου.



Εικόνα 2: Διαδικασία Επιβλεπόμενης Μάθησης

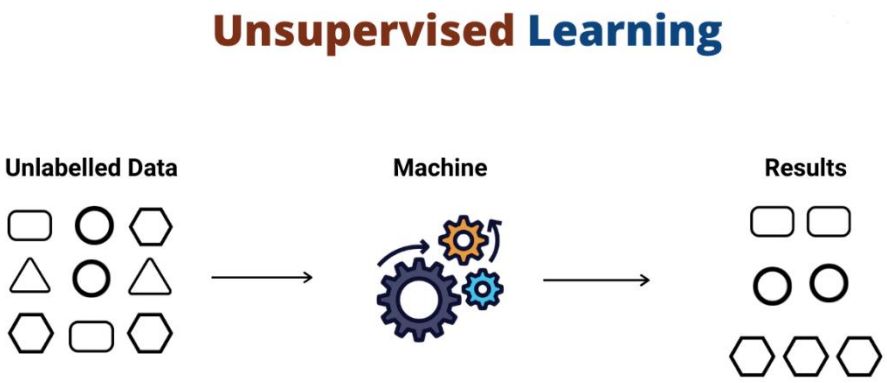
3.2 Μη-Επιβλεπόμενη Μάθηση

Στην Μη-Επιβλεπόμενη μάθηση το σύστημα μάθησης χρησιμοποιεί δεδομένα εκπαίδευσης για τα οποία δεν είναι γνωστές οι κλάσεις τους και προσπαθεί να τα χειριστεί με βάση ομοιότητες ή ανομοιότητες που μπορεί να παρουσιάζουν μεταξύ τους [13] . Επειδή δεν υπάρχουν δεδομένα με γνωστές κλάσεις είναι δύσκολο να γίνει ποσοτική αξιολόγηση της απόδοσης του συστήματος. Βασικά παραδείγματα προβλημάτων μάθηση χωρίς επίβλεψη αποτελούν:

- Η Συσταδοποίηση, στην οποία τα δεδομένα διαχωρίζονται σε n ομάδες (συστάδες).

- Η ελάττωση διαστάσεων (dimensionality reduction), η οποία προσπαθεί να αναπαραστήσει κάθε περίπτωση των δεδομένων εκπαίδευσης με μικρότερο πλήθος χαρακτηριστικών, διατηρώντας όμως παράλληλα τις χαρακτηριστικές ιδιότητες των δεδομένων.

- Ο εντοπισμός καινοτομιών (novelty detection), στην οποία αναγνωρίζονται κάποιες περιπτώσεις (λίγες), οι οποίες διαφέρουν από την πλειοψηφία των περιπτώσεων.



Εικόνα 3: Διαδικασία Μη-Επιβλεπόμενης Μάθησης

3.3 Ημι-Επιβλεπόμενη Μάθηση

Στην Ημι-Επιβλεπόμενη Μάθηση, το σύστημα μάθησης λαμβάνει ένα σύνολο δεδομένων εκπαίδευσης που αποτελείται από μικρό πλήθος δεδομένων με γνωστές τις κλάσεις τους και μεγάλο πλήθος δεδομένων χωρίς γνωστές κλάσεις και στη συνέχεια παράγει προβλέψεις για νέα δεδομένα [13] . Συγκεκριμένα, το πρόβλημα της Ημι-Επιβλεπόμενης Μάθηση μπορεί να διατυπωθεί ως εξής:

> Έστω ένα σύνολο δεδομένων εκπαίδευσης, το οποίο αποτελείται από ένα μικρό σύνολο δεδομένων $Ld = \{(x_1, y_1), (x_2, y_2), ..., (x_i, y_i)\}$ με γνωστές τις κλάσεις τους και ένα μεγάλο σύνολο δεδομένων $Ud = \{x_{i+1}, x_{i+2}, ..., x_{i+u}\}$ με άγνωστες κλάσεις, με $x_i \in \mathbb{R}^N$. Ο σκοπός της Ημι-επιβλεπόμενης μάθησης είναι διττός. Αρχικά, θέλουμε να δημιουργήσουμε ένα μοντέλο μάθησης βασισμένο στα δεδομένα εκπαίδευσης και στη συνέχεια να χρησιμοποιήσουμε το μοντέλο για την πρόβλεψη των τιμών μελλοντικών δεδομένων.

Η Ημι-Επιβλεπόμενη Μάθηση εφαρμόζεται συχνά σε προβλήματα όπου είναι εύκολη η συλλογή δεδομένων χωρίς να είναι γνωστές οι κλάσεις, ενώ αντίθετα τα δεδομένα με γνωστές κλάσεις είναι δύσκολο να αποκτηθούν, είτε λόγω διότι απαιτείται πολύς χρόνος, είτε λόγου μεγάλου κόστους. Διάφοροι τύποι προβλημάτων όπως η ταξινόμηση, η πρόβλεψη τιμής, η ταξινόμηση με βάση κριτήριο μπορούν να αντιμετωπιστούν ως προβλήματα Ημι-Επιβλεπόμενης Μάθησης.

Πολλές μελέτες έχουν δείξει ότι ο συνδυασμός Επιβλεπόμενης και Μη Επιβλεπόμενης Μάθησης μπορεί να οδηγήσει στην αξιοποίηση δεδομένων με άγνωστες κλάσεις για τη δημιουργία μοντέλων μάθησης με καλύτερη απόδοση από αυτά που δημιουργούνται μέσω της Επιβλεπόμενης Μάθησης [16].

Γνωστές μέθοδοι Ημι-Επιβλεπόμενης Μάθησης είναι:

- **Αυτό-εκπαίδευση (Self-training)**
  Η μέθοδος self-training θεωρείται από τις πιο απλές και αποτελεσματικές μεθόδους Ημι-επιβλεπόμενης Μάθησης [17]. Η συγκεκριμένη μέθοδος βασίζεται στις δικές της προβλέψεις σε μη ετικετοποιημένα δεδομένα για να εκπαιδευτεί.

  Αρχικά, ένας ταξινομητής εκπαιδεύεται σε μικρό πλήθος ετικετοποιημένων δεδομένων του συνόλου εκπαίδευσης και στη συνέχεια χρησιμοποιείται για να προβλέψει τις κλάσεις μη ετικετοποιημένων δεδομένων. Οι 60 περισσότερο σίγουρες προβλέψεις προστίθενται στο σύνολο των ετικετοποιημένων δεδομένων και η διαδικασία επαναλαμβάνεται για συγκεκριμένο πλήθος επαναλήψεων [13].

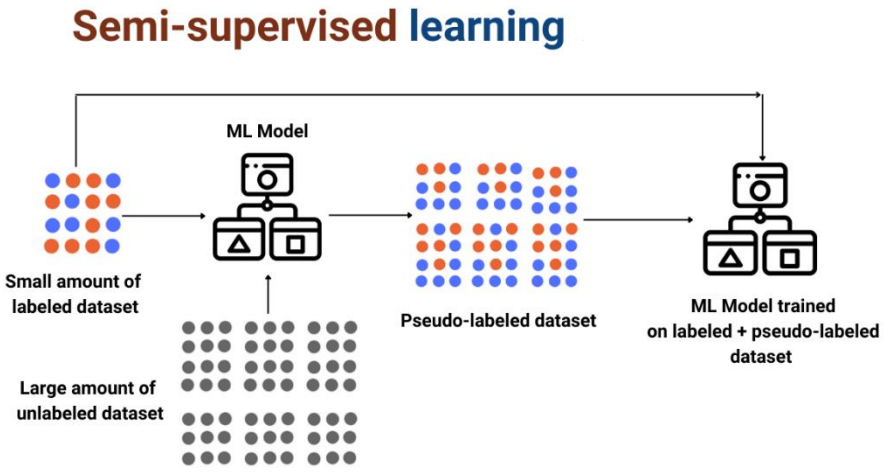
- **Συνεκπαίδευση (Co-training)**
  Η μέθοδος co-training έχει χρησιμοποιηθεί ευρέως σε πολλά προβλήματα Ημι-επιβλεπόμενης Μάθησης [18]. Η μέθοδος στηρίζεται στην υπόθεση ότι κάθε στιγμιότυπο του συνόλου δεδομένων μπορεί να διαχωριστεί σε δύο διακριτά σύνολα χαρακτηριστικών, τα οποία ονομάζονται πεδία. Καθένα από

τα πεδία αυτά είναι επαρκές για σωστή Κατηγοριοποίηση , ενώ είναι μεταξύ τους ανεξάρτητα. Σε αυτή τη βάση, δύο αλγόριθμοι μάθησης εκπαιδεύονται ξεχωριστά σε κάθε πεδίο με βάση τα ετικετοποιημένα δεδομένα εκπαίδευσης, οι πιο σίγουρες προβλέψεις καθενός στα μη ετικετοποιημένα δεδομένα προστίθενται στο σύνολο εκπαίδευσης του άλλου και η διαδικασία επαναλαμβάνεται για συγκεκριμένο πλήθος επαναλήψεων [18].

- **Τρι-εκπαίδευση (Tri-training)**
  Η μέθοδος tri-training αποτελεί παραλλαγή της μεθόδου co-training, αλλά δεν απαιτεί την ύπαρξη δύο ανεξάρτητων πεδίων χαρακτηριστικών [19].
  Αντίθετα, στηρίζεται στη μέθοδο εμφωλίασης (bagging) σύμφωνα με την οποία δημιουργούνται αυτοδύναμα υποσύνολα του αρχικού συνόλου δεδομένων ίδιου μεγέθους μέσω μιας δειγματοληπτικής επαναληπτικής διαδικασίας [20]. Στη συνέχεια χρησιμοποιεί τρεις αλγόριθμους μάθησης οι οποίοι εκπαιδεύονται στα υποσύνολα αυτά. Στην ουσία πρόκειται για έναν αλγόριθμο εμφωλίασης τριών ταξινομητών [21]. Αν δύο από τους αλγόριθμους συμφωνούν στην πρόβλεψη της κλάσης ενός μη ετικετοποιημένου στιγμιότυπου, τότε αυτό χρησιμοποιείται για την εκπαίδευση του τρίτου αλγόριθμου.

- **Δημοκρατική Συνεκπαίδευση (Democratic Co-training)**
  Αποτελεί μια επίσης παραλλαγή της μεθόδου co-training [22]. Σε αυτή την μέθοδο, τρεις αλγόριθμοι μάθησης εκπαιδεύονται στο ίδιο σύνολο ετικετοποιημένων δεδομένων. Αν δύο από τους αλγόριθμους συμφωνούν στην πρόβλεψη της κλάσης ενός μη ετικετοποιημένου στιγμιότυπου, τότε αυτό χρησιμοποιείται για την εκπαίδευση του τρίτου αλγόριθμου.

- **Τρι-εκπαίδευση με Επεξεργασία (Tri-training with Editing)** [23].
- **RASCO** [24].
- **Rel-RASCO** [25].
  Οι μέθοδοι αυτές αποτελούν επίσης τροποποιήσεις της μεθόδου co-training και έχουν εφαρμοστεί με επιτυχία για την επίλυση προβλημάτων κατηγοριοποίησης Ημι-επιβλεπόμενης Μηχανικής Μάθησης



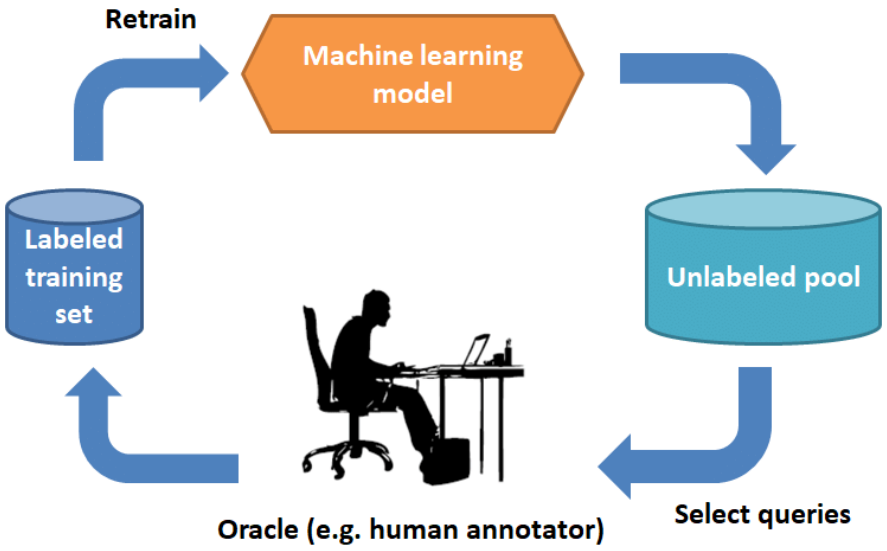
Εικόνα 4: Διαδικασία Ημι-Επιβλεπόμενης Μάθησης

3.4 Ενεργή Μηχανική Μάθηση

Όπως και στην Ημι-Επιβλεπόμενη Μάθηση, έτσι και στην Ενεργή Μηχανική Μάθηση, το σύστημα μάθηση λαμβάνει ένα σύνολο δεδομένων εκπαίδευσης που αποτελείται από μικρό πλήθος δεδομένων με γνωστές τις κλάσεις τους και μεγάλο πλήθος δεδομένων χωρίς γνωστές κλάσεις και στη συνέχεια παράγει προβλέψεις για νέα δεδομένα.

Σε αυτή την περίπτωση, το μοντέλο επιλέγει με προσοχή εκείνες τις περιπτώσεις για τις οποίες είναι περισσότερο αβέβαιο να προβλέψει την τιμή της μεταβλητής απόφασης (ετικέτα) και στη συνέχεια θέτει ερωτήματα και ζητά από έναν ειδικό (σύστημα ή άνθρωπο) τις ετικέτες αυτών των περιπτώσεων [26].

Το βασικό σημείο της Ενεργής Μηχανικής Μάθησης είναι η δημιουργία ενός ταξινομητή υψηλής ακρίβειας χωρίς να γίνουν πάρα πολλά ερωτήματα χρησιμοποιώντας ένα μικρό σύνολο δεδομένων εκπαίδευσης.



Εικόνα 5: Διαδικασία Ενεργής Μάθησης

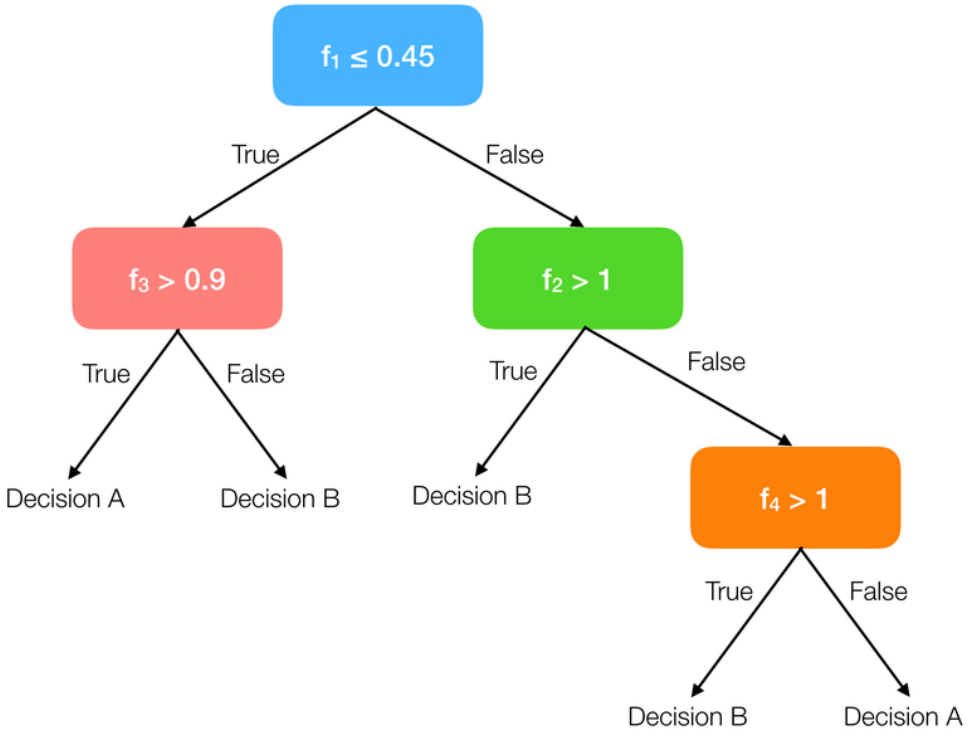
4. Μοντέλα - Αλγόριθμοι Πρόβλεψης

4.1 Εισαγωγή

Στο παρόν εδάφιο θα γίνει μια συνοπτική παρουσίαση μερικών από των βασικότερων Μοντέλων που μπορούν να εφαρμοστούν στο πεδίο της πρόβλεψης. Πέραν των Μοντέλων που παρουσιάζουμε εδώ μπορεί κανείς να εφαρμόσει μια πληθώρα άλλων μοντέλων καθώς και συνδυασμούς (Ensemble models) και τεχνικές αυτών.

4.2 Δέντρα Αποφάσεων

Τα Δένδρα Αποφάσεων αποτελούν μία από τις πιο σημαντικές και διαδεδομένες μεθόδους για την κατηγοριοποίηση δεδομένων, στην οποία επιχειρείται η προσέγγιση μιας τιμής μιας κατηγορικής συνάρτησης απόφασης ακολουθώντας την τεχνική του «διαίρει και βασίλευε» [27]. Ένα δέντρο απόφασης είναι μία γραφική απεικόνιση όλων των πιθανών διαδρομών που οδηγούν στο τελικό αποτέλεσμα (Σχήμα 2).



Σχήμα 2: Δέντρο Απόφασης για 4 τυχαίες μεταβλητές

Ένα Δένδρο Απόφασης έχει τους εξής τύπους κόμβων [14]:

- Τον αρχικό κόμβο (ρίζα), ο οποίος δεν έχει εισερχόμενες ακμές.
- Τους εσωτερικούς κόμβους, οι οποίοι αντιστοιχούν σε μια μεταβλητή που χρησιμοποιείται για περαιτέρω διαχωρισμό του δένδρου. Τους εξερχόμενες ακμές από τον αρχικό ή κάθε εσωτερικό κόμβο, αντιστοιχεί μία συνθήκη ελέγχου με βάση την τιμή τους μεταβλητής.
- Τους εξωτερικούς κόμβους (φύλλα), οι οποίοι αντιστοιχούν στα αποτελέσματα.

4.3 Αλγόριθμος Bayes

Αποτελεί μια απλή, γρήγορη και αρκετά αποτελεσματική μέθοδο ταξινόμησης η οποία χρησιμοποιεί πιθανοτικά μοντέλα τα οποία στηρίζονται στο θεώρημα του Bayes [28] σύμφωνα με το οποίο ισχύει ότι:

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

Όπου :

- $P(A \mid B)$ η δεσμευμένη πιθανότητα (a-posteriori probability) του ενδεχομένου A, δεδομένου του ενδεχομένου B.

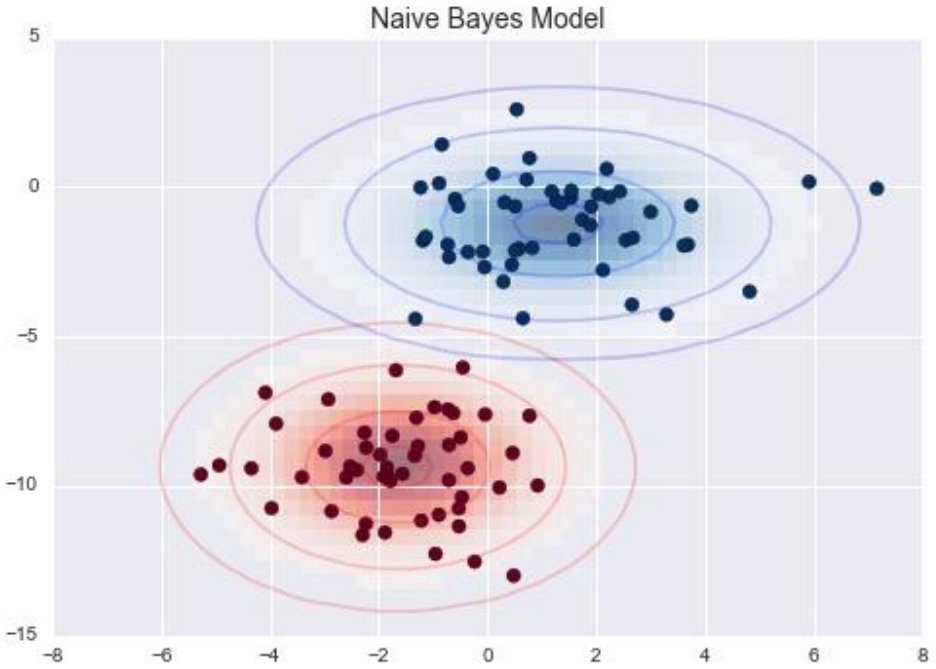
- $P(A)$ είναι η πιθανότητα πραγματοποίησης του ενδεχομένου A και είναι γνωστή ως «εκ των προτέρων πιθανότητα του A» (a-priori probability).

- $P(B \mid A)$ είναι η δεσμευμένη πιθανότητα του ενδεχομένου B, δεδομένου του A. Η πιθανότητα αυτή είναι δυνατόν να υπολογιστεί από τη γνώση που διαθέτουμε για το συγκεκριμένο πρόβλημα.

- $P(B)$ είναι η πιθανότητα πραγματοποίησης του ενδεχομένου B.

Ο κατηγοριοποιητής Bayes χρησιμοποιείται για την εκτίμηση της πιθανότητας ενός στιγμιότυπου να ανήκει σε μια από τις προκαθορισμένες κλάσεις υπό την υπόθεση ότι τα χαρακτηριστικά είναι μεταξύ τους ανεξάρτητα.

Η υπόθεση της ανεξαρτησίας των χαρακτηριστικών δεν ισχύει πάντοτε, της απλοποιεί κατά πολύ της υπολογισμούς οδηγώντας σε καλή εκτίμηση της πιθανότητας χωρίς να απαιτεί μεγάλο σύνολο εκπαίδευσης.



Σχήμα 3: Απεικόνιση του αλγόριθμου Bayes

4.4 Γραμμική Παλινδρόμηση

Η τεχνική της Παλινδρόμησης χρησιμοποιείται για την μοντελοποίηση και την ανάλυση αριθμητικών δεδομένων, μιας εξαρτημένης μεταβλητής και κάποιων ανεξάρτητων μεταβλητών.

Αναζητείται μια συνάρτηση συσχέτισης της εξαρτημένης μεταβλητής από τις ανεξάρτητες. Η μοντελοποίηση αυτή δεν απαιτεί να γνωρίζουμε εκ των προτέρων τον τρόπο που συνδέονται οι ανεξάρτητες μεταβλητές με την εξαρτημένη.

Η Γραμμική Παλινδρόμηση [29] υπολογίζει βάρη wi για το υπερεπίπεδο που αντιστοιχεί σε μια εξίσωση της μορφής $w_0 + w_1 x_1 + ... + w_i y \geq 0$. Στο δυσδιάστατο χώρο αυτό αντιστοιχεί σε ευθεία γραμμή με εξίσωση $w_0 + w_1 x + w_2 y = 0$.

Πιο συγκεκριμένα, στην απλή γραμμική παλινδρόμηση υπάρχει η ανεξάρτητη μεταβλητή x_i και δύο παράμετροι, w_0 και w_1. Το μοντέλο έχει την παρακάτω μορφή:

$$y_i = a + bx_i + \varepsilon_i, \quad \text{i=1,m}$$

όπου ε_i είναι το σφάλμα της πρόβλεψης.

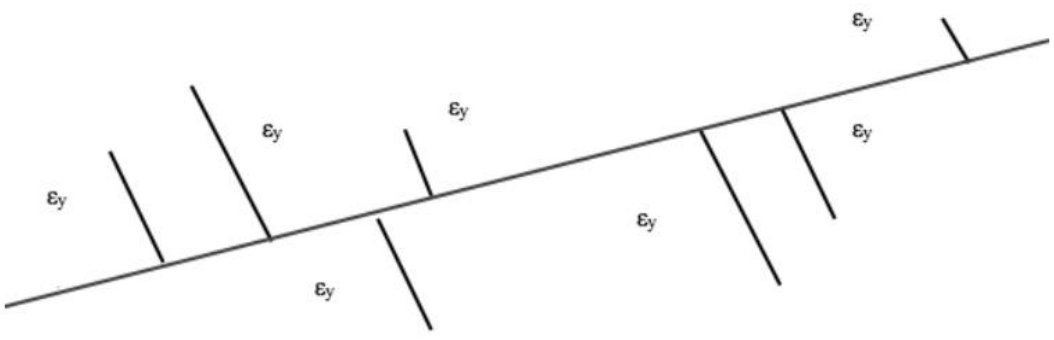
Ο στόχος λοιπόν είναι η ελαχιστοποίηση του αθροίσματος των τετραγωνικών σφαλμάτων με τον προσδιορισμό των κατάλληλων παραμέτρων. Η σχέση υπολογισμού του αθροίσματος τετραγωνικών σφαλμάτων δίνεται από τον παρακάτω τύπο:

$$SSE = \sum_{i=1}^{N} e_i^2$$

Στην περίπτωση της απλής Γραμμικής Παλινδρόμησης ο προσδιορισμός των παραμέτρων που ελαχιστοποιούν την παραπάνω σχέση πραγματοποιείται με τη επίλυση δύο απλών εξισώσεων:

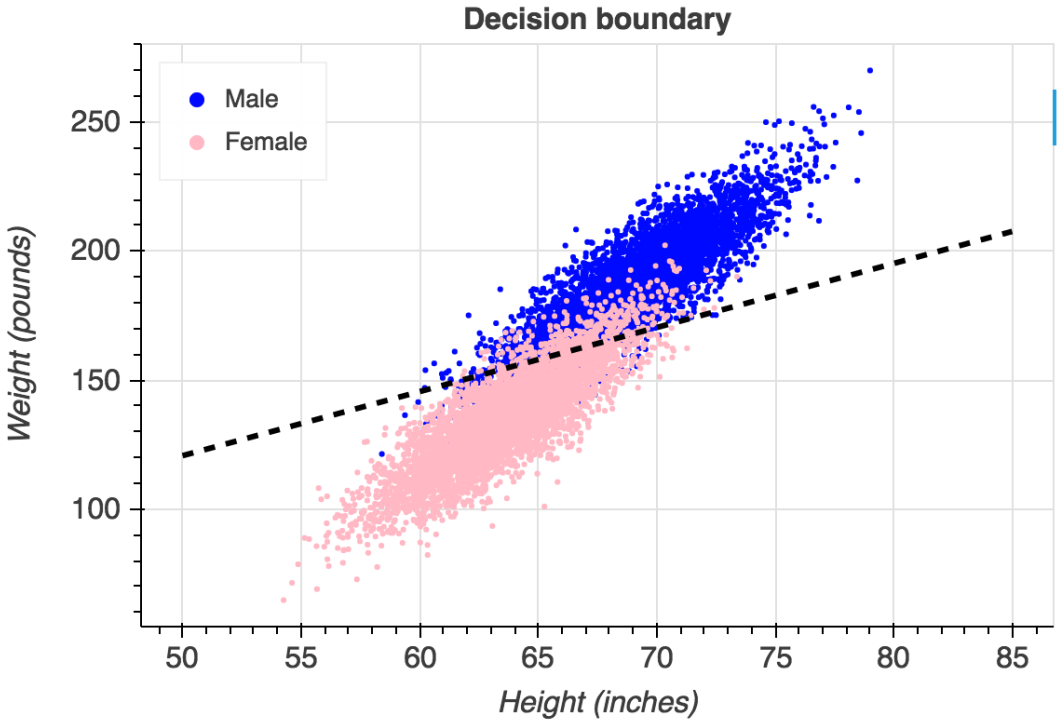
$$\hat{\beta} = \frac{\sum_{i=1}^{n}(x_i - \bar{X})(y_i - \bar{Y})}{\sum_{i=1}^{n}(x_i - \bar{X})^2} \qquad \text{και} \qquad \hat{\alpha} = \bar{y} - \hat{\beta}\bar{x}$$

όπου $\bar{X}$, ο μέσος όρος της ανεξάρτητης μεταβλητής και $\bar{Y}$, ο μέσος όρος των τιμών y. Ουσιαστικά λοιπόν αναζητούμε την ευθεία εκείνη οποία ελαχιστοποιεί τα ε_y στο παρακάτω σχήμα:



Σχήμα 4: Απεικόνιση σφαλμάτων γραμμικού μοντέλου

Στο παρακάτω σχήμα (Σχήμα 5) βλέπουμε το διαχωρισμό που κάνει ο αλγόριθμος με βάση το ύψος και το βάρος, αποφασίζοντας αν είναι άνδρας ή γυναίκα.



Σχήμα 5: Απεικόνιση του αλγόριθμου Γραμμικής Παλινδρόμησης

4.5 Κ - Πλησιέστεροι Γείτονες

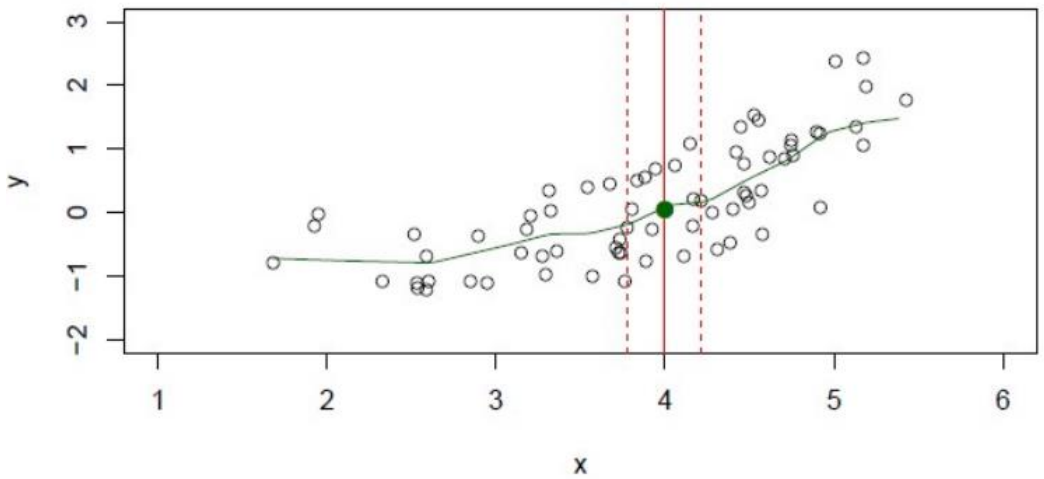
Η μέθοδος των Κ-Πλησιέστερων Γειτόνων [30] βρίσκεται ανάμεσα στις πιο δημοφιλείς μεθόδους μηχανικής μάθησης. Αντιπροσωπεύει την παλαιότερη (μη-παραμετρική) γενιά μεθόδων που προτάθηκαν για αυτό το πρόβλημα και έχει ενδελεχώς εφαρμοστεί στο πεδίο της στατιστικής. Τελευταία έχει ανανεωθεί το ενδιαφέρον για αυτή στην κοινότητα της Μηχανικής Μάθησης. Παρόλη τη βασική της απλότητα και το γεγονός ότι πολλές εναλλακτικές προηγμένες τεχνικές έχουν αναπτυχθεί μετά την πρώτη της εμφάνιση, η μέθοδος των Κ-Πλησιέστερων Γειτόνων παραμένει ακόμα η πιο επιτυχημένη για πολλά προβλήματα.

Αυτός ο αλγόριθμος ανήκει σε μια κατηγορία αλγόριθμων που βασίζονται στα στιγμιότυπα (instances) του ιστορικού συνόλου δεδομένων για να προσεγγίσουν τη λύση σε ένα νέο στιγμιότυπο. Ο αλγόριθμος αυτός, αντίθετα με άλλους, δεν κατασκευάζει ένα μοντέλο από τα δεδομένα εκπαίδευσης, για να το εφαρμόσει στα δεδομένα εξέτασης για την ταξινόμηση ονοματικών (nominal) ή την πρόβλεψη συνεχών (continues) τιμών.

Ο αλγόριθμος αυτός αναθέτει σε ένα αντικείμενο με άγνωστη τιμή εξαρτημένης μεταβλητής την πλειοψηφική τιμή που προκύπτει από k αντικείμενα εγνωσμένης

τιμής για την εξαρτημένη μεταβλητή, κοντύτερα σε αυτό. Επομένως δεν κατασκευάζει αρχικά κάποιο μοντέλο. Αυτού του είδους η μάθηση καλείται “μάθηση Βασισμένη-σε-Περίπτωση” (Instance Based learning, IB), ή “Οκνηρή Εκμάθηση” (Lazy Learning). Η έννοια της εγγύτητας συνήθως εκφράζεται από ένα κριτήριο απόστασης στον ευκλείδειο χώρο με άξονες τις μεταβλητές εισόδου των αντικειμένων (τις ιδιότητες).

Στην πλειονότητα των περιπτώσεων εφαρμογής του αλγόριθμου σε σχετικά περιορισμένο όγκο δεδομένων η απλή ευκλείδεια απόσταση προτιμάται λόγω απλότητας υλοποίησης. Στο παρακάτω σχήμα βλέπουμε την εφαρμογή του *KNN* σε αριθμητικά δεδομένα.



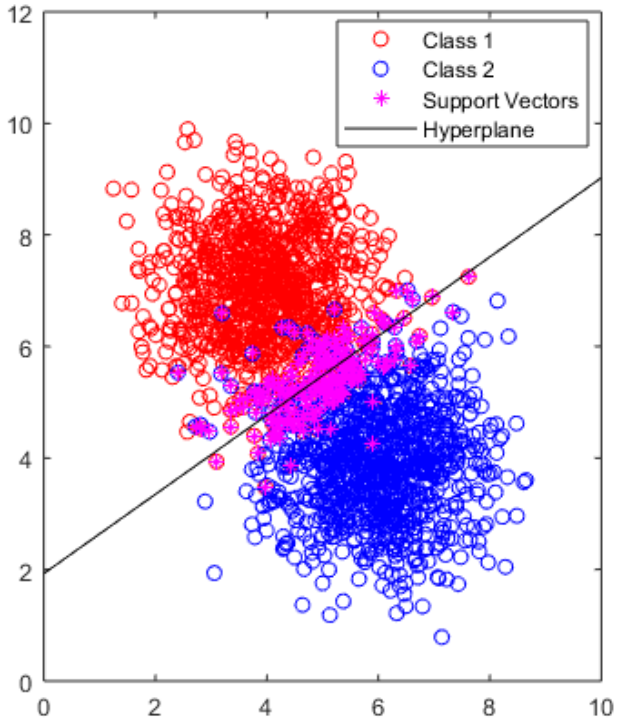
Σχήμα 6: Απεικόνιση του αλγόριθμου K- Πλησιέστεροι Γείτονες

4.6 Μηχανές Διανυσμάτων Υποστήριξης

Οι Μηχανές Διανυσμάτων Υποστήριξης (Support Vector Machines) (SVM) ταξινομεί κατασκευάζοντας ένα N-διαστάσεων υπερ-επίπεδο που διαχωρίζει βέλτιστα τα δεδομένα σε δύο κατηγορίες.

Τα μοντέλα SVM είναι ενός στενός ξάδερφος των πολλαπλών στρωμάτων νευρωνικών δίκτυών Perceptron. Χρησιμοποιώντας μια λειτουργία πυρήνα, τα SVM είναι μια εναλλακτική μέθοδος κατάρτισης για την λειτουργία βάσης και πολλαπλών στρώσεων κατηγοριοποιητών Perceptron.

Έτσι, ο στόχος των μοντέλων SVM είναι να βρούνε το βέλτιστο υπερ-επίπεδο διαχωρίζοντας έτσι διαφορετικές περιπτώσεις των κλάσεων, εντός ενός πολυδιάστατου χώρου με την δημιουργία υπερ-επιπέδων, υποστηρίζοντας με αυτόν τον τρόπο περιπτώσεις ταξινόμησης, παλινδρόμησης αλλά και συνεχείς μεταβλητές και μεταβλητές χαρακτηριστικών [31].



Σχήμα 7: Απεικόνιση του αλγόριθμου Μηχανές Διανυσμάτων Υποστήριξης

4.7 Τεχνητά Νευρωνικά Δίκτυα

Τα Τεχνητά Νευρωνικά Δίκτυα [32] αρχικά προτάθηκαν ως ένα μαθηματικό μοντέλο προσομοίωσης της πολύπλοκης λειτουργίας του ανθρώπινου εγκεφάλου. Η δομή του εγκεφάλου είναι τέτοια ώστε να είναι δυνατή η παράλληλη επεξεργασία δεδομένων και η συνεχής μάθησης μέσω της αλληλεπίδρασης με το περιβάλλον. Τα δύο αυτά βασικά χαρακτηριστικά του επιτρέπουν να είναι σε θέση να εκτελεί δύσκολες διεργασίες αναγνώρισης προτύπων αλλά και να εξελίσσεται συνεχώς.

Η δομή του Τεχνητού Νευρωνικού Δικτύου βασίζεται σε εκείνη του βιολογικού νευρωνικού δικτύου, ώστε να εμφανίζει παρόμοιες ιδιότητες. Έτσι, όπως και ένα δίκτυο νευρώνων εγκεφάλου, ένα τεχνητό δίκτυο αποτελείται από ένα σύνολο τεχνητών νευρώνων που αλληλοεπιδρούν, ενώ συνδέονται μεταξύ τους με τις λεγόμενες συνάψεις (synapses). Ο βαθμός αλληλεπίδρασης διαφέρει για κάθε ζεύγος νευρώνων και καθορίζεται από τα λεγόμενα συναπτικά βάρη (synaptic weights). Συγκεκριμένα, κατά την αλληλεπίδραση του Νευρωνικού Δικτύου με το

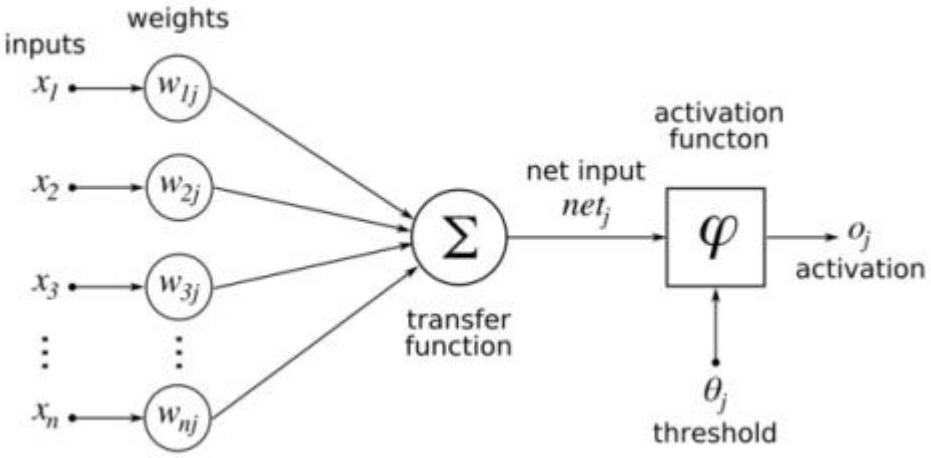
περιβάλλον ,τα συναπτικά βάρη μεταβάλλονται συνεχώς, ενδυναμώνοντας ή αποδυναμώνοντας την ισχύ του κάθε δεσμού. Με αυτόν τον τρόπο όλη η εμπειρική γνώση που αποκτά το Νευρωνικό Δίκτυο από το περιβάλλον κωδικοποιείται στα συναπτικά βάρη. Αυτά αποτελούν το χαρακτηριστικό εκείνο που δίνει στο δίκτυο την ικανότητα για εξέλιξη και προσαρμογή στο περιβάλλον.

Υπάρχουν δύο τρόποι να εκπαιδεύσουμε ένα δίκτυο. Κατά τον πρώτο τρόπο, η εκπαίδευση γίνεται με εποπτεία. Στην περίπτωση αυτή το δίκτυο τροφοδοτείται με ένα σύνολο γνωστών παραδειγμάτων, δηλαδή ένα σύνολο καταστάσεων στις οποίες μπορεί να περιέλθει το δίκτυο, μαζί με τα αποτελέσματα που θέλουμε να δίνει το δίκτυο για τις καταστάσεις αυτές. Για να μάθει το δίκτυο τα παραδείγματα αυτά, χρησιμοποιούμε έναν αλγόριθμο εκπαίδευσης.

Ο αλγόριθμος εκπαίδευσης που θα χρησιμοποιηθεί εξαρτάται από το εκάστοτε πρόβλημα και από τη δομή του δικτύου που επιλέγουμε για να το αντιμετωπίσουμε. Κατά το δεύτερο τρόπο, η εκπαίδευση γίνεται χωρίς εποπτεία. Στην περίπτωση αυτή το δίκτυο καλείται να αναγνωρίσει ομοιότητες και μοτίβα σε δεδομένα που του έχουμε τροφοδοτήσει. Τα δεδομένα παρουσιάζονται στο δίκτυο και αυτό πρέπει να προσαρμοστεί έτσι ώστε να τα χωρίσει σε ομάδες. Η διαδικασία αυτή επαναλαμβάνεται, ώσπου δεν παρατηρείται μεταβολή στην ταξινόμηση των δεδομένων.

Το βασικό πλεονέκτημα των νευρωνικών δικτύων είναι ότι μπορούν να αποθηκεύσουν γνώση και εμπειρία από το περιβάλλον, την οποία μπορούν στη συνέχεια να ανακαλέσουν. Επιπλέον, έχουν τη δυνατότητα να γενικεύουν, δηλαδή να εξάγουν τα βασικά χαρακτηριστικά ενός συστήματος, ακόμα και όταν αυτά είναι κρυμμένα σε θορυβώδη δεδομένα.

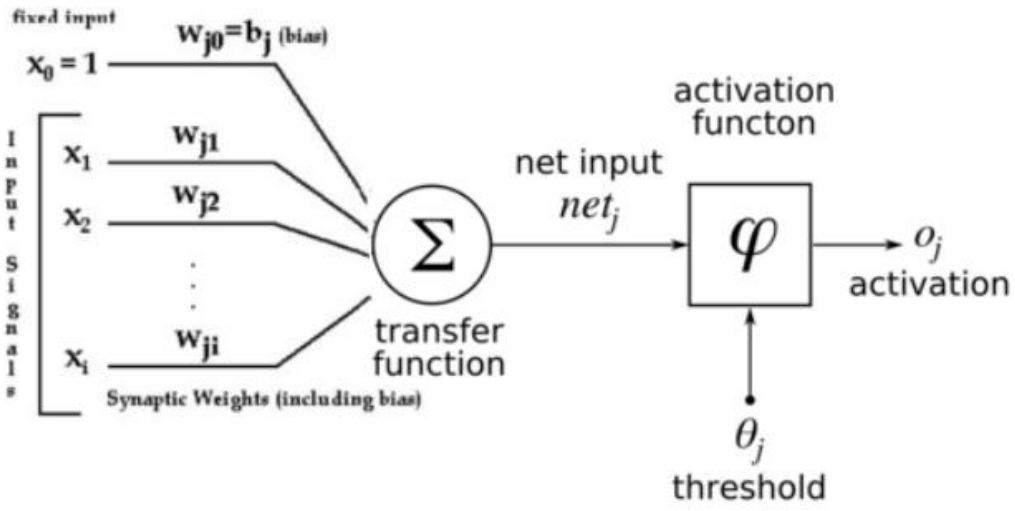
Σε αναλογία με το βιολογικό νευρώνα του εγκεφάλου, ο τεχνητός νευρώνας είναι η δομική μονάδα του Τεχνητού Νευρωνικού Δικτύου. Σε αυτόν συντελείται όλη η επεξεργασία της πληροφορίας. Κάθε νευρώνας δέχεται πληροφορία, την επεξεργάζεται και δίνει μία τιμή εξόδου. Οι είσοδοί του είναι είτε οι έξοδοι άλλων νευρώνων, είτε το πρωταρχικό σήμα εισόδου του δικτύου. Στην παρακάτω εικόνα παρουσιάζεται το βασικό μμοντέλο του νευρώνα που χρησιμοποιείται ως επί το πλείστων κατά την υλοποίηση Τεχνητών Νευρωνικών Δικτύων.



Σχήμα 8: Βασικό μοντέλο νευρώνα

Στο νευρώνα αυτό, η πληροφορία κινείται πάντα προς μία κατεύθυνση, από αριστερά προς τα δεξιά, δεν υπάρχει δηλαδή κανένας βρόχος ανάδρασης. Με βάση αυτό μπορούμε λοιπόν να διακρίνουμε τρεις βασικές φάσεις της λειτουργίας του:

1. Κατά την πρώτη φάση, κάθε είσοδος πολλαπλασιάζεται με το συναπτικό βάρος που της αντιστοιχεί.
2. Στη δεύτερη φάση οι σταθμισμένες πλέον είσοδοι και ένας εξωτερικά εφαρμοζόμενος παράγοντας, η μεροληψία ή πόλωση ή κατώφλι (bias, threshold), αθροίζονται και δίνουν το τοπικό πεδίο (net input, induced local field, activation potential). Για λόγους απλούστευσης, η μεροληψία μπορεί να θεωρηθεί ως μία επιπλέον είσοδος, με συναπτικό βάρος ίσο προς την τιμή του και πάγια τιμή εισόδου ίση προς τη μονάδα. Στην περίπτωση αυτή ο νευρώνας παίρνει τη μορφή που φαίνεται στην παρακάτω εικόνα:



Σχήμα 9: Νευρώνας - Συνάρτηση ενεργοποίησης

Μέχρι εδώ, ο νευρώνας δεν κάνει τίποτα άλλο από το να δίνει έναν γραμμικό συνδυασμό των εισόδων, με συντελεστές τα προσαρμοζόμενα συναπτικά βάρη. Αν η λειτουργία του λοιπόν σταματούσε εδώ, τότε θα είχαμε έναν γραμμικό νευρώνα, που θα έδινε ένα γραμμικό προσαρμοζόμενο φίλτρο (linear adaptive filter). Ένα Τεχνητό Νευρωνικό Δίκτυο που αποτελείται από τέτοιους νευρώνες θα είναι γραμμικό.

3. Τέλος, στην τρίτη φάση, εφαρμόζεται η συνάρτηση ενεργοποίησης ή συνάρτηση μεταφοράς (activation function ή squashing function) στο τοπικό πεδίο και το αποτέλεσμα δίνει την έξοδο του νευρώνα.

Στα πρώτα μοντέλα νευρώνα, η συνάρτηση ενεργοποίησης [33] ήταν μία Βηματική Συνάρτηση (Step Function) π.χ.

$$\varphi(x) = \begin{cases} 0, & x \le \theta \\ 1, & x > \theta \end{cases}$$

Αν η έξοδος του νευρώνα ήταν ίση προς 0 ο νευρώνας ήταν αδρανής διαφορετικά ήταν ενεργοποιημένος. Το παραπάνω μοντέλο αναφέρεται συχνά ως μοντέλο McCulloch-Pitts [14] προς τιμή αυτών που το πρότειναν. Αργότερα, η εξέλιξη στη θεωρία των Τεχνητών Νευρωνικών Δικτύων απέδειξε ότι η παράγωγος της συνάρτησης ενεργοποίησης μπορεί να δώσει χρήσιμες πληροφορίες για το νευρωνικό δίκτυο και να χρησιμοποιηθεί στην εκπαίδευσή του, γεγονός που οδηγεί στο συμπέρασμα ότι είναι προτιμότερο να χρησιμοποιηθεί μία παραγωγίσιμη συνάρτηση και όχι η βηματική συνάρτηση, που προφανώς δεν είναι παραγωγίσιμη.

Σήμερα, στα περισσότερα μοντέλα η συνάρτηση ενεργοποίησης είναι μία Σιγμοειδής Συνάρτηση (Sigmoid Function). Αυτή είναι γενικά μία πραγματική, συνεχής και φραγμένη συνάρτηση, της οποίας η παράγωγος είναι θετική. Το πεδίο ορισμού της μπορεί θεωρητικά να είναι όλο το σύνολο των πραγματικών αριθμών, αλλά στην πράξη μπορεί να περιοριστεί, θέτοντας όρια στις τιμές των συναπτικών βαρών. Το σύνολο τιμών είναι συνήθως το διάστημα [0,1] ή [-1,1]. Ένα από τα πιο γνωστά παραδείγματα σιγμοειδούς συνάρτησης που χρησιμοποιείται ως συνάρτηση ενεργοποίησης είναι η Λογιστική Συνάρτηση (Logistic Function), που δίνεται από τον τύπο:

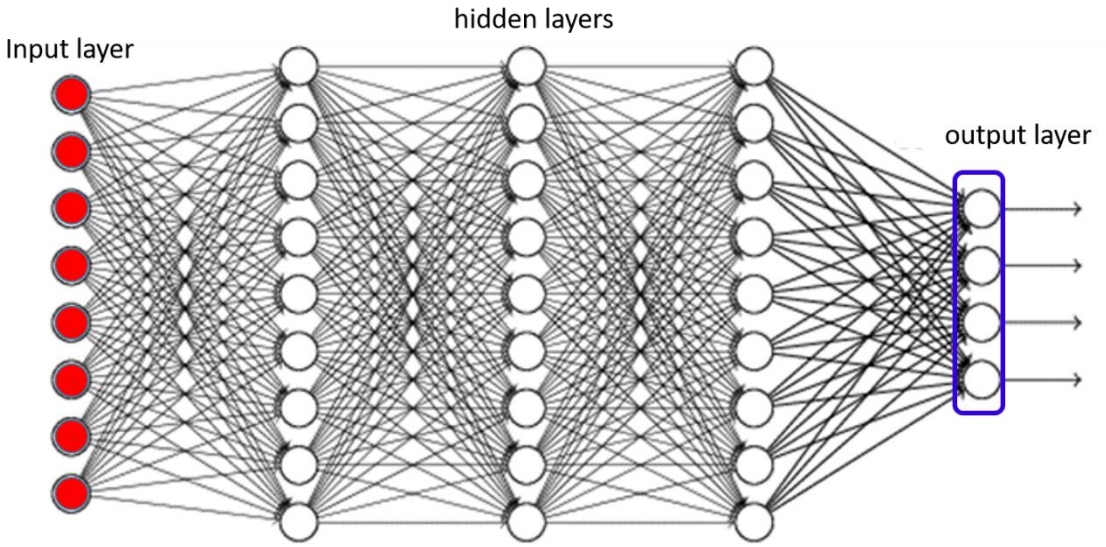
$$\varphi(x) = \frac{1}{1+e^{-ax}}$$

όπου α η παράμετρος κλίσης. Μεταβάλλοντας την παράμετρο κλίσης, παίρνουμε διαφορετικές συναρτήσεις. Όσο το α τείνει στο άπειρο, η Λογιστική Συνάρτηση τείνει προς τη Βηματική Συνάρτηση και έχουμε και πάλι το μοντέλο McCulloch-Pitts [34].

Με την εισαγωγή της συνάρτησης ενεργοποίησης, ο νευρώνας γίνεται μη γραμμικός. Αντίστοιχα, ένα τεχνητό νευρωνικό δίκτυο που αποτελείται από τέτοιους νευρώνες θα είναι μη γραμμικό. Αυτή η εγγενής μη γραμμικότητα των νευρωνικών δικτύων είναι ένα πλεονέκτημα έναντι άλλων γνωστών μεθόδων αντιμετώπισης πολλών προβλημάτων.

Για παράδειγμα, όταν σε ένα πρόβλημα πρόβλεψης το σύστημα που μελετάμε είναι μη γραμμικό και ιδιαίτερα όταν παρουσιάζει χαοτική συμπεριφορά, τα γνωστά γραμμικά μοντέλα πρόβλεψης αδυνατούν να δώσουν σωστά

αποτελέσματα. Σε αυτές τις περιπτώσεις, τα μη γραμμικά τεχνητά νευρωνικά δίκτυα είναι προτιμότερα.



Σχήμα 10: Απεικόνιση Νευρωνικού Δικτύου με τρία κρυμμένα στρώματα νευρώνων

Υπάρχουν τρεις τύποι νευρώνων: οι νευρώνες εισόδου, οι νευρώνες εξόδου και οι υπολογιστικοί νευρώνες ή κρυμμένοι νευρώνες

5. Μετρικές Αξιολόγησης μεθόδων Μηχανικής Μάθησης

5.1 Συντελεστής Προσδιορισμού R^2

Ο συντελεστής προσδιορισμού R^2 χρησιμοποιείται ως μετρική αξιολόγησης σε μοντέλα Μηχανικής Μάθησης που σχετίζονται με την Παλινδρόμηση.

Κύριος σκοπός είναι είτε η πρόβλεψη μελλοντικών αποτελεσμάτων είτε η δοκιμή υποθέσεων, βάσει άλλων σχετικών πληροφοριών. Παρέχει ένα μέτρο για το πόσο καλά τα παρατηρημένα αποτελέσματα αναπαράγονται από το μοντέλο, με βάση το ποσοστό της συνολικής μεταβολής των αποτελεσμάτων που εξηγείται από το μοντέλο [35].

Είναι σημαντικό να γίνει η παραδοχή ότι για αλγόριθμους Ταξινόμησης δεν λειτουργεί ο συντελεστής προσδιορισμού R^2 λόγο της φύσης του και καταφεύγουμε σε άλλα εργαλεία αξιολόγησης. Αποτελεί βασικό μέτρο αξιολόγησης για έναν αλγόριθμο Παλινδρόμησης.

Η ποσότητα $R^2 = \frac{SSR}{SSTO}$ ονομάζεται *συντελεστής προσδιορισμού (coefficient of determination)*[36].Η σχέση αυτή γράφεται:

$$R^2 = \frac{SSR}{SSTO} = \frac{SSTO - SSE}{SSTO} = 1 - \frac{SSE}{SSTO}$$

όπου $SSR = \sum_{i=1}^{n} (\hat{y}_i - \bar{y})^2$, το άθροισμα των τετραγώνων της Παλινδρόμησης

$SSE = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$, το άθροισμα τετραγώνων των εκτιμημένων σφαλμάτων

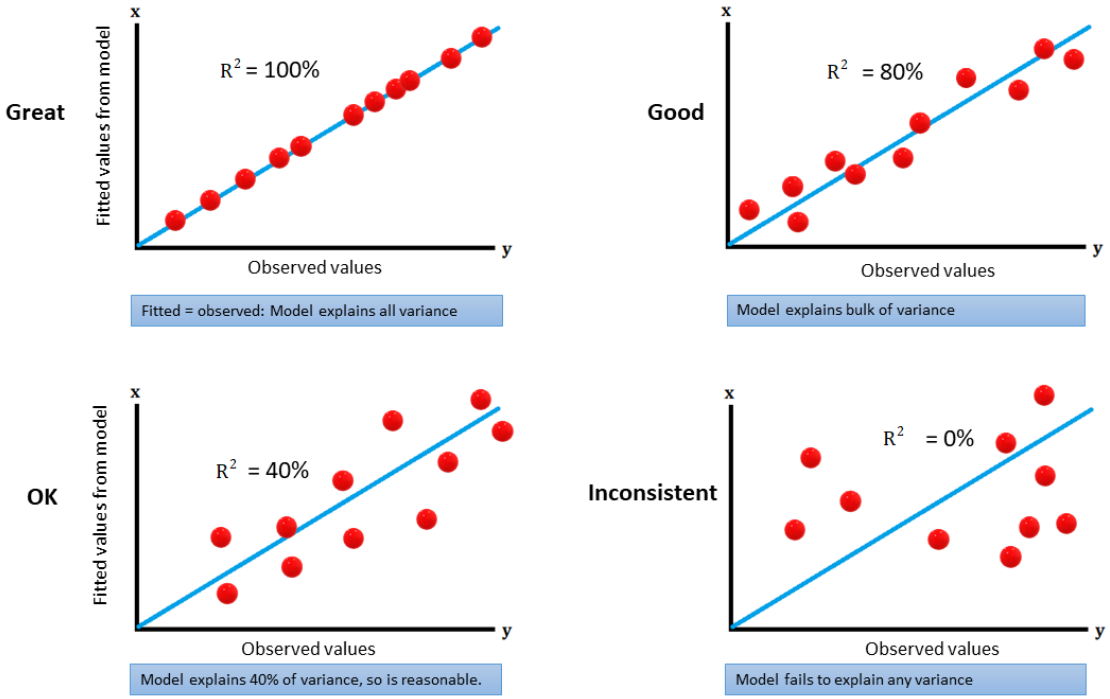
$SSTO = \sum_{i=1}^{n} (y_i - \bar{y})^2$, το ολικό άθροισμα τετραγώνων

όπου $\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x$, η ευθεία ελαχίστων τετραγώνων

$y = \beta_0 + \beta_1 x$, η ευθεία παλινδρόμησης

Ο συντελεστής προσδιορισμού εκφράζει το ποσοστό της συνολικής μεταβλητότητας των y_i που εξηγείται (απορροφάται) από την παλινδρόμηση. Οι τιμές του είναι μεταξύ 0 και 1 $(0 \le R^2 \le 1)$ [36].

- Όσο το R^2 πλησιάζει την τιμή 1 τότε τα σφάλματα μηδενίζονται και συνεπώς το μοντέλο περιγράφει τέλεια τα δεδομένα μας [36].
- Όσο το R^2 πλησιάζει την τιμή 0 τότε το μοντέλο μας δεν περιγράφει καλά τα δεδομένα μας [36].



Σχήμα 11: Σύγκριση R^2 για διάφορα γραμμικά μοντέλα (ίδιο dataset)

5.2 Καμπύλες Διαχείρισης Λειτουργικών Χαρακτηριστικών (ROC Curves)

Οι καμπύλες ROC (Receiver Operating Characteristic) χρησιμοποιούνται για την αξιολόγηση τεχνικών Μηχανικής Μάθησης και την Εξόρυξη Δεδομένων (Data Mining) με μία από τις πρώτες εφαρμογές τη σύγκριση και την αξιολόγηση διαφόρων αλγορίθμων ταξινόμησης [37][38].

Οι καμπύλες ROC, είναι γραφικές μέθοδοι αξιολόγησης των χαρακτηριστικών διαγνωστικών δοκιμών. Αποτελούν γραφικές παραστάσεις για την ανταλλαγή (trade-off) μεταξύ ευαισθησίας και ιδιαιτερότητας, ειδικότερα μεταξύ των εσφαλμένων αρνητικών [False Negative (FN)] και εσφαλμένων θετικών [False Positive (FP)] τιμών για κάθε πιθανή διακοπή. Μία καμπύλη ROC δείχνει τη σχέση δύο ειδικών κατανομών υπό την ίδια τάξη μονοτονικών μετασχηματισμών.

Η καμπύλη ROC παρέχει μία οπτική αναπαράσταση των σχετικών διαφορών μεταξύ των ωφελημάτων (αληθή θετικά) και του κόστους (εσφαλμένα θετικά) της ταξινόμησης για τις κατανομές δεδομένων. Στην περίπτωση ταξινομήσεων σκληρού-τύπου με διακριτές ετικέτες τάξεων, κάθε ταξινομητής παράγει ένα ζεύγος (TP_rate, FP_rate) που αντιστοιχεί σε ένα απλό σημείο της καμπύλης ROC [38][39]. Οι καμπύλες ROC χρησιμοποιούνται για την αξιολόγηση της ακρίβειας των προβλέψεων. Σημειώνεται ότι οι προβλέψεις αποτελούν βασικό μέρος κάθε επιχείρησης και έρευνας επιστημονικών πεδίων [40].

Η αξιολόγηση των καμπύλων ROC χρησιμοποιεί την αναλογία δύο μετρικών αξιολόγησης (με δύο στήλες): την αληθή θετική τιμή (TP rate) και την εσφαλμένη θετική τιμή (FP rate).

Το διάγραμμα της καμπύλης ROC σχηματίζεται σχεδιάζοντας τις τιμές TP και FP, και κάθε σημείο του διαστήματος ROC αντιστοιχεί στην απόδοση ενός απλού ταξινομητή για μία δεδομένη κατανομή. Η καμπύλη ROC είναι χρήσιμη επειδή παρέχει μία ορατή αναπαράσταση των σχετικών διαφορών μεταξύ των πλεονεκτημάτων (από τις αληθείς θετικές) και μειονεκτημάτων/κόστη (εσφαλμένες θετικές) της ταξινόμησης σε σχέση με τις κατανομές δεδομένων.

Συνήθως χρησιμοποιείται ένας Πίνακας Σύγχυσης ο (Confusion Matrix), γνωστός ως και πίνακας συνάφειας ή πίνακας σφάλματος, που είναι μια συγκεκριμένη διάταξη πίνακα η οποία απεικονίζει την απόδοση ενός αλγορίθμου, συνήθως με Επιβλεπόμενη Μάθηση (στη μάθηση χωρίς επίβλεψη συνήθως ονομάζεται Πίνακας Ταιριάσματος (Matching Matrix)). Κάθε στήλη του πίνακα αντιπροσωπεύει τις περιπτώσεις σε μια προβλεπόμενη τάξη, ενώ κάθε σειρά αντιπροσωπεύει τις περιπτώσεις σε μια πραγματική κλάση. Το όνομα προέρχεται από το γεγονός ότι καθιστά εύκολο να διαπιστωθεί αν το σύστημα συγχέει δύο κλάσεις, δηλαδή υπάρχει εσφαλμένη επισήμανση ως προς ένα άλλο.

Οι σημαντικότεροι βασικοί παράγοντες τεχνικών μετρικής είναι η Ακρίβεια (Accuracy) και η τιμή σφάλματος (error rate). Αν θεωρήσουμε ένα βασικό πρόβλημα ταξινόμησης δύο τάξεων και {p,n} είναι ετικέτες της αληθούς θετικής p και της αρνητικής n τάξης, τότε μία αναπαράσταση της απόδοσης ταξινόμησης μπορεί να δοθεί από ένα Πίνακα Σύγχυσης (Confusion Matrix) [41]. Συνήθως ένας πίνακας διαστάσεων (2x2) αναφέρεται στον αριθμό των ψευδώς θετικών

(false positives), ψευδώς αρνητικών (false negatives),αληθώς θετικών (true positives), αληθώς αρνητικών (true negatives) αποτελεσμάτων.

Η καμπύλη ROC επίσης δείχνει τις ενδείξεις TPR(True Positive Rate) στον άξονα των y και FPR (False Positive Rate) στον άξονα x. Η απόδοση κάθε ταξινομητή αναπαρίσταται ως ένα σημείο στην καμπύλη ROC.

Η αποτίμηση του μοντέλου ROC σχετίζεται με τις ακόλουθες εξισώσεις :

$$TPR = \frac{TP}{TP + FN}, \quad \text{θετικές τιμές}$$

$$FPR = \frac{FP}{TN + FP}, \quad \text{αρνητικές τιμές}$$

Η τεχνική αξιολόγησης με καμπύλες ROC χρησιμοποιεί την αναλογία δύο απλών στηλών που βασίζονται σε μετρικές αξιολογήσεις, δηλαδή την αληθή-θετική τιμή (TP_rate) και την εσφαλμένη θετική τιμή (FP_rate), οι οποίες ορίζονται με τον ακόλουθο τρόπο:

$$TP\_\text{rate} = \frac{TP}{Pc} \qquad \text{και} \qquad FP\_\text{rate} = \frac{FP}{Nc}$$

Πίνακας 2: Πίνακας σύγχυσης για αναπαράσταση απόδοσης ταξινόμησης

		Πρόβλεψη		
		Θετικό	Αρνητικό	
Πραγματικό (Αποτελέσματα Τεστ)	Θετικό	True Positive (TP)	False Positive (FP)	Πρόβλεψη (Precision)
	Αρνητικό	False Negative (FN)	True Negative (TN)	Αρνητική Προβλεπόμενη Τιμή
		Ευαισθησία (Sensitivity)	Ειδικότητα (Specificity)	Ακρίβεια (Accuracy)

Η ακρίβεια και η τιμή σφάλματος δίνονται από τους τύπους:

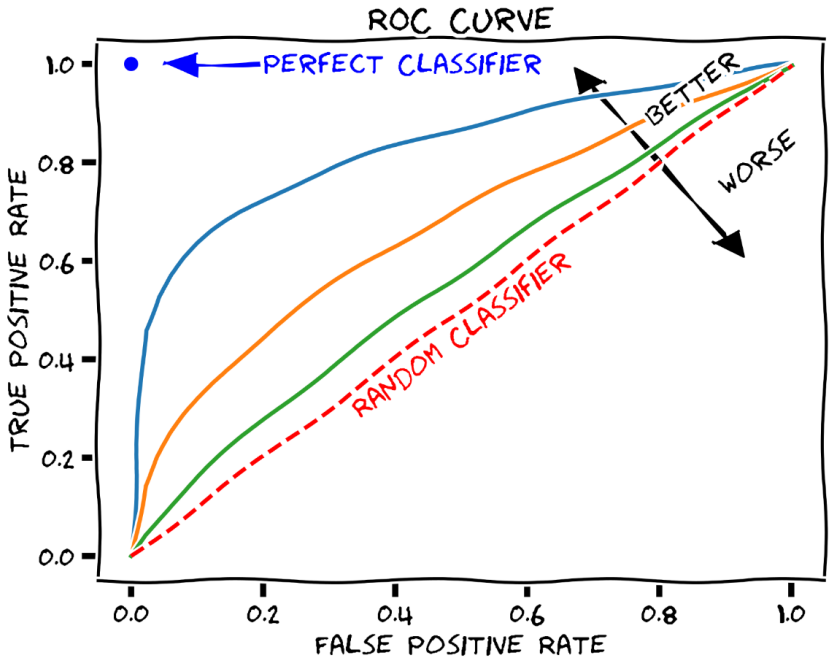
- $\text{Accuracy} = \frac{TP + TN}{Pc + Nc}$, Ακρίβεια
- $\text{Error\_Rate} = 1 - \text{Accuracy}$, Τιμή Σφάλματος
- $\text{Precision} = \frac{TP}{TR + FP}$, Ακρίβεια Προσέγγισης
- $\text{Recall} = \frac{TP}{TP + FN}$, Ανάκληση

  - όπου TP είναι True Positive, TN είναι True Negative,
  - Pc, Nc είναι Column Counts

Το γράφημα της καμπύλης ROC σχηματίζεται σχεδιάζοντας την TP_rate πάνω από την FP_rate, και κάθε σημείο της ROC αντιστοιχεί στην απόδοση ενός απλού ταξινομητή σε μια δεδομένη κατανομή [38].

Οι καμπύλες ROC (Receiver Operating Characteristic) μπορούν να χρησιμοποιηθούν για συνολική απόδοση ταξινομητών που αναφέρονται σε ένα φάσμα ανταλλαγών μεταξύ αληθών θετικών τιμών σφαλμάτων και μη-αληθών θετικών τιμών σφαλμάτων [42].

Η περιοχή κάτω από την καμπύλη (Area Under the Curve [AUC]) αποτελεί ένα αποδεκτό κριτήριο μέτρησης της απόδοσης για μια καμπύλη ROC [43].



Σχήμα 12: Η καμπύλη ROC για έναν καλύτερο και χειρότερο ταξινομητή

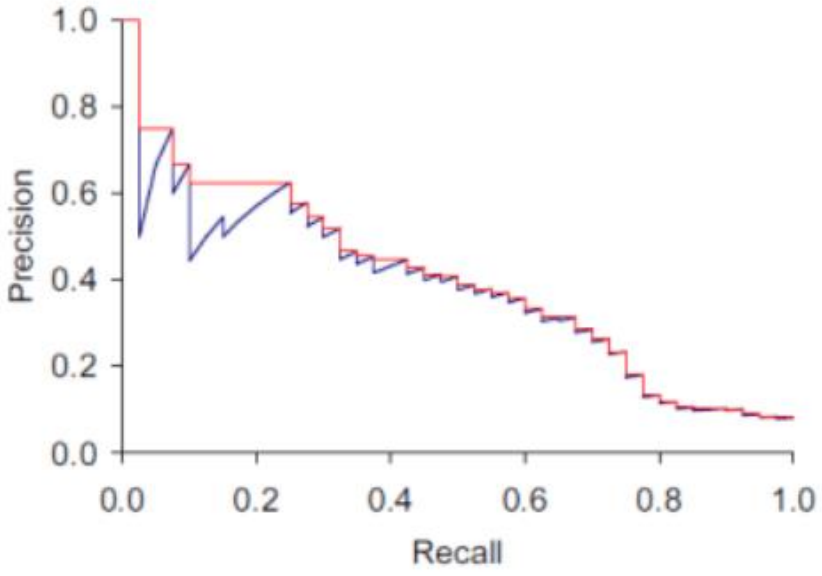
5.3 Καμπύλες Ανάκλησης Ακρίβειας (PR Curves)

Οι Καμπύλες Ανάκλησης Ακρίβειας (Precision Recall Curves) (PR Curves), που αντιστοιχούν στις καμπύλες ROC , μπορούν να δώσουν μια πληρέστερη πληροφοριακή αναπαράσταση της αξιολόγησης απόδοσης από τις καμπύλες ROC . Μια καμπύλη ROC ορίζεται σχεδιάζοντας την τιμή ακριβείας (precision rate) πάνω από την τιμή ανάκλησης (recall rate). Μια καμπύλη κυριαρχεί στο χώρο ROC αν και μόνο αν κυριαρχεί στο χώρο PR [44].

Οι καμπύλες ROC μπορούν να δώσουν μια εποπτική αξιολόγηση της απόδοσης της μεθόδου, αλλά παρουσιάζουν μια αισιόδοξη άποψη της απόδοσης του αλγορίθμου. Σε τέτοιες περιπτώσεις οι PR Curves μπορούν να δώσουν πληρέστερες πληροφοριακές παραστάσεις για την αξιολόγηση απόδοσης των τεχνικών καμπυλών PR [44].

Η τιμή ακριβείας στον άξονα Υ υπολογίζεται από τον αριθμό των σχετικών στοιχείων που ανακτώνται προς τα σχετικά στοιχεία. Συνηθίζεται να παρουσιάζεται μια γραφική παράσταση με διακριτά «οδοντωτό» σχήμα.

Για παράδειγμα, εάν έχουμε το $(\kappa+1)$ - οστό στοιχείο που ανακτάται και δεν είναι σχετικό, παρατηρούμε ότι η ανάκληση είναι η ίδια ,όμως η ακρίβεια μειώνεται. Εάν έχουμε το $(\kappa+1)$ - οστό στοιχείο που ανακτάται και είναι σχετικό, τότε η ανάκληση και η ακρίβεια αυξάνονται και η καμπύλη δημιουργεί απότομες γωνίες προς τα δεξιά.



Σχήμα 13: Σχέση Precision και Recall

Για να αποφύγουμε αυτές τις απότομες γωνίες μπορεί να χρησιμοποιηθεί η τιμή ακρίβειας με παρεμβολή, όπου σε ένα ορισμένο επίπεδο ανάκλησης r ορίζεται ως υψηλότερη ακρίβεια που συναντάμε για κάθε επίπεδο ανάκλησης $r' \geq r$ με σχετικό τύπου

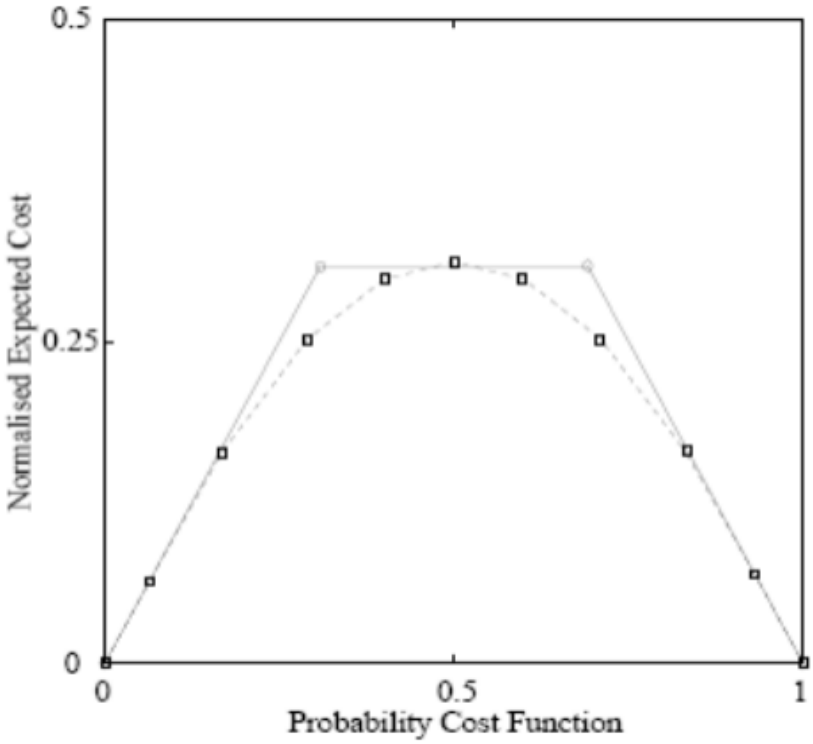
$$P_{\text{int}ep}(r) = \max_{r' \geq r} p(r')$$

5.4 Καμπύλες Ακρίβειας / Κόστους

Οι Καμπύλες Κόστους παρέχουν μια κατανοητή αποτίμηση μετρικής για την απόδοση ταξινομητών σε μια μεταβαλλόμενη τάξη πιθανοτήτων ή κόστη εσφαλμένης ταξινόμησης [45].

Οι καμπύλες ROC αδυνατούν να δώσουν διαστήματα εμπιστοσύνης για την απόδοση ενός ταξινομητή και δεν μπορούν να συνάγουν την στατιστική σημασία της απόδοσης διαφόρων ταξινομητών [45]. Η καμπύλη κόστους που έχει την ικανότητα να εκφράσει μια τεχνική αποτίμησης με ευαισθησία κόστους που έχει την ικανότητα να εκφράσει άμεσα την απόδοση ενός ταξινομητή για μεταβαλλόμενα κόστη εσφαλμένης τοποθέτησης και τάξης κατανομών σε ένα οπτικό περίγραμμα (visual format). Η μέθοδος καμπύλης κόστους έχει τα χαρακτηριστικά αναπαράστασης της ανάλυσης ROC και προσφέρει διευρυμένες πληροφορίες για την απόδοση ταξινόμησης [45]. Το αναμενόμενο κόστος ενός ταξινομητή μπορεί να παρασταθεί άμεσα από την καμπύλη , που είναι εύκολο να κατανοηθεί.

Η καμπύλη κόστους επιτρέπει την άμεση παρατήρηση του διαστήματος κόστους και καταδεικνύει αν ένας ταξινομητής είναι καλύτερος και ποσοτικά πόσο καλύτερος από άλλους ταξινομητές . Το αναμενόμενο κόστους ενός ταξινομητή για όλες τις πιθανές επιλογές εσφαλμένου κόστους και κατανομών τάξεων εμφανίζεται στο ακόλουθο διάγραμμα .



Σχήμα 14: Αναμενόμενο Κόστος Ταξινομητών

Στο Σχήμα 14 ο άξονας X είναι η πιθανότητα συνάρτησης κόστους για θετικά παραδείγματα και ορίζεται ως

$$PCF(+) = \frac{w_+}{(w_+ + w_-)}$$

Και ο άξονας Y είναι το αναμενόμενο κανονικοποιημένο κόστους ως προς το κόστος που προκύπτει όταν κάθε παράδειγμα είναι ταξινομημένο εσφαλμένα και ορίζεται ως

$$NE[C] = (1 - TP)w_+ + FPww_+ + w_-$$

όπου

- $w_+ = p(+)C(-|+)$
- $w_- = p(-)C(+|-)$

- $P(a)$: η πιθανότητα ενός δεδομένου παραδείγματος να είναι σε μια τάξη a και

- $C(a|b)$: το κόστος που προκύπτει αν ένα παράδειγμα σε μια τάξη b ταξινομηθεί εσφαλμένα ότι ανήκει στην τάξη α.

6. Εφαρμογή σε Python

6.1 Εισαγωγή

Στο παρόν εδάφιο θα οργανώσουμε ένα πείραμα με πραγματικά δεδομένα με σκοπό να δείξουμε και στην πράξη την χρήση των τεχνικών εξόρυξης και επεξεργασίας δεδομένων αλλά και των αλγορίθμων πρόβλεψης της Μηχανικής Μάθησης.

Το πείραμα αυτό θα αφορά την επιβίωση ή όχι ενός επιβάτη του Τιτανικού, που ως γνωστόν ξεκίνησε το παρθενικό του ταξίδι από το Σαουθάμπτον προς τη Νέα Υόρκη το 1912. Κατά τη διάρκεια αυτού προσέκρουσε σε παγόβουνο στον Βόρειο Ατλαντικό προκαλώντας έτσι την τραγική βύθιση του επιβατηγού υπερωκεάνιου και τον θάνατο σε πάνω από 1500 άτομα όπου επέβαιναν σε αυτό. Είναι μέχρι και σήμερα ένα από τα πιο εικονικά και θανατηφόρα ναυτικά δυστυχήματα σε καιρό ειρήνης.

Πιο συγκεκριμένα, θα ξεκινήσουμε την διαδικασία αυτή με την απόκτηση ενός Συνόλου Δεδομένων (Dataset) (https://www.kaggle.com/c/titanic/data).

Εν συνεχεία θα παρουσιάσουμε τα δεδομένα αυτά με σχήματα ώστε να γίνει ευνόητο το σύνολο των δεδομένων που θα επεξεργαστούμε.

Σε δεύτερο χρόνο, θα επεξεργαστούμε τα χαρακτηριστικά (features) δηλαδή τις στήλες του dataset ώστε να καταλήξουμε σε ένα τροποποιημένο dataset, πιο εύχρηστο και στοχευμένο στην εξαγωγή των αποτελεσμάτων μας. Παράλληλα, θα δούμε νέα γραφήματα από αυτά τα δεδομένα.

Επίσης, μετά από όλη αυτή την επεξεργασία αφού θα έχουμε πλέον τα δεδομένα μας έτοιμα, θα χρησιμοποιήσουμε διάφορους αλγόριθμους πρόβλεψης καθώς το πρόβλημά μας είναι πρόβλημα Κατηγοριοποίησης (Classification) και Παλινδρόμησης (Regression). Θέλουμε να αναγνωρίσουμε την σχέση μεταξύ της εξόδου που είναι η επιβίωση ή όχι και άλλων χαρακτηριστικών (features) όπως φύλο, ηλικία, λιμάνι επιβίβασης, κλπ.

Με βάση αυτά κινούμαστε στην κατηγορία της Επιβλεπόμενης Μάθησης και οι αλγόριθμοι – μοντέλα που θα χρησιμοποιήσουμε είναι τα παρακάτω:

- Decision Tree (Classification)
- Naive Bayes classifier (Classification)
- Logistic Regression (Regression)
- KNN or k-Nearest Neighbors (Regression)
- Support Vector Machines (Regression)
- Perceptron (Classification)
- Neural Network (Classification)

Τέλος, θα δούμε την Ακρίβεια του κάθε μοντέλου και θα γίνει σύγκριση. Πρέπει εδώ να γίνει η παραδοχή ότι διαφορετικά μοντέλα για διαφορετικά dataset αποδίδουν αλλότροπα, αφού τα χαρακτηριστικά (features), η συσχέτιση (correlation) αυτών και ο όγκος των δεδομένων παίζουν καθοριστικό ρόλο στο αποτέλεσμα.

Για την υλοποίηση θα χρησιμοποιηθεί η γλώσσα προγραμματισμού Python. Η ανάπτυξη θα γίνει με την βοήθεια του Anaconda (https://www.anaconda.com/) μια διανομή της Python και του Jupyter Notebook (https://jupyter.org/) ένα ολοκληρωμένο περιβάλλον ανάπτυξης (IDE).

Τα υπογραμμισμένα τμήματα του κειμένου που ακολουθούν παραπέμπουν στο αντίστοιχο κομμάτι κώδικα όπου βρίσκεται στο σύνολο του στο Παράρτημα - Κώδικας. Κάθε γκρίζο κουτί είναι και ένα κελί (cell) στην Python.

6.2 Το Σύνολο Δεδομένων και τα Χαρακτηριστικά

Το dataset που θα χρησιμοποιηθεί προέρχεται από τα στοιχεία που καταγράφηκαν για τους επιβάτες του Τιτανικού.

Τα δεδομένα έχουν διαιρεθεί σε δύο γκρουπ συν ένα υπο-γκρούπ:

- Training set (train.csv) (Συλλογή Εκπαίδευσης)
- Test set (test.csv) (Συλλογή Δοκιμής)
    - Gender submission (gender_submission.csv) ()

Συνολικά έχουμε 1309 εισόδους εκ των οποίων:

- 891 στο train.csv
- 418 στο test.csv

Το Training set θα χρησιμοποιηθεί για να εκπαιδευτεί το εκάστοτε μοντέλο μας, ώστε να μάθει να κατηγοριοποιεί τα δεδομένα στο μέλλον κατά βούληση.

Το Test set θα χρησιμοποιηθεί ως είσοδος δεδομένων στο εκάστοτε μοντέλο μας. Αυτή η συλλογή μένει «κρυφή» από τον αλγόριθμο πρόβλεψης, όσο αυτός εκπαιδεύεται. Σε αυτή τη συλλογή δεν υπάρχει το χαρακτηριστικό Survived και έτσι θα πρέπει το μοντέλο μας να αποφασίσει εάν επιβίωσε ή όχι στην βύθιση του Τιτανικού.

Το Gender submission περιέχει μόνο το χαρακτηριστικό Survived ως υποσύνολο του Test set και αποτελείται από την βασική αλήθεια (ground truth), ώστε στο στάδιο της αξιολόγησης να αντικρούσουμε τα αποτελέσματα των Μοντέλων με τα πραγματικά.

Στον παρακάτω πίνακα (Πίνακας 3) φαίνεται ο κατάλογος των χαρακτηριστικών των δεδομένων.

Πίνακας 3:Κατάλογος Χαρακτηριστικών και Δεδομένων

Feature (Χαρακτηριστικό)	Description (Περιγραφή)	Value (Τιμή)
PassengerId	Passenger ID	Integer (Ακέραιος)
Survived	Survived or not (Επιβίωσε ή όχι)	0 = No, 1 = Yes
Pclass	Ticket class (Τάξη εισιτηρίου)	1 = 1st, 2 = 2nd, 3 = 3rd
Name	Name (Όνομα)	
Sex	Sex (Φύλο)	male / female (άρρεν / θήλυ)
Age	Age in years (Ηλικία σε χρόνια)	Integer (Ακέραιος)
SibSp	# of siblings / spouses aboard the Titanic (# αδερφιών / συζύγων επιβαίνοντες στον Τιτανικό)	Integer (Ακέραιος)
Parch	# of parents / children aboard the Titanic (# γονέων / παιδιών επιβαίνοντες στον Τιτανικό)	Integer (Ακέραιος)
Ticket	Ticket number (Αριθμός εισιτήριου)	Alphanumeric (Αλφαριθμητικό)
Fare	Passenger fare (Ναύλος επιβατών)	Float (Δεκαδικός)
Cabin	Cabin number (Αριθμός Καμπίνας)	Alphanumeric (Αλφαριθμητικό)
Embarked	Port of embarkation (Λιμάνι επιβίβασης)	C = Cherbourg, Q = Queenstown, S = Southampton

όπου

- Pclass: μέτρο για την κοινωνικοοικονομική κατάσταση (SES)
  - 1st = ανώτερη τάξη
  - 2nd = μεσαία τάξη
  - 3rd = κατώτερη τάξη
- SibSp: περιγράφει τις οικογενειακές σχέσεις ως εξής
  - Sibling = αδελφός, αδελφή
  - Spouse = σύζυγος
- Parch: περιγράφει τις οικογενειακές σχέσεις ως εξής
  - Parent = πατέρας, μητέρα
  - Child = γιος, κόρη

  Κάποια παιδιά ταξίδεψαν με νταντά και συνεπώς Parch = 0

6.3 Πρώτη ματιά στα Δεδομένα

Αρχικά εισάγουμε τις βιβλιοθήκες και τα datasets που θα μας βοηθήσουν κατά τη διάρκεια του πειράματος.

Βλέπουμε τα data types/formats των στοιχείων:

Πίνακας 4: Datatypes των στοιχείων των 2 γκρουπ δεδομένων

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object
 11  Embarked     889 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
None
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 418 entries, 0 to 417
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  418 non-null    int64
 1   Pclass       418 non-null    int64
 2   Name         418 non-null    object
 3   Sex          418 non-null    object
 4   Age          332 non-null    float64
 5   SibSp        418 non-null    int64
 6   Parch        418 non-null    int64
 7   Ticket       418 non-null    object
 8   Fare         417 non-null    float64
 9   Cabin        91 non-null     object
 10  Embarked     418 non-null    object
dtypes: float64(2), int64(4), object(5)
memory usage: 36.0+ KB
None
```

Μερικά στατιστικά μέτρα/στοιχεία:

Πίνακας 5: Στατιστικά μέτρα/στοιχεία του συνόλου δεδομένων

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

Οι πρώτες 5 είσοδοι:

Πίνακας 6: Πρώτες 5 είσοδοι στο σύνολο δεδομένων

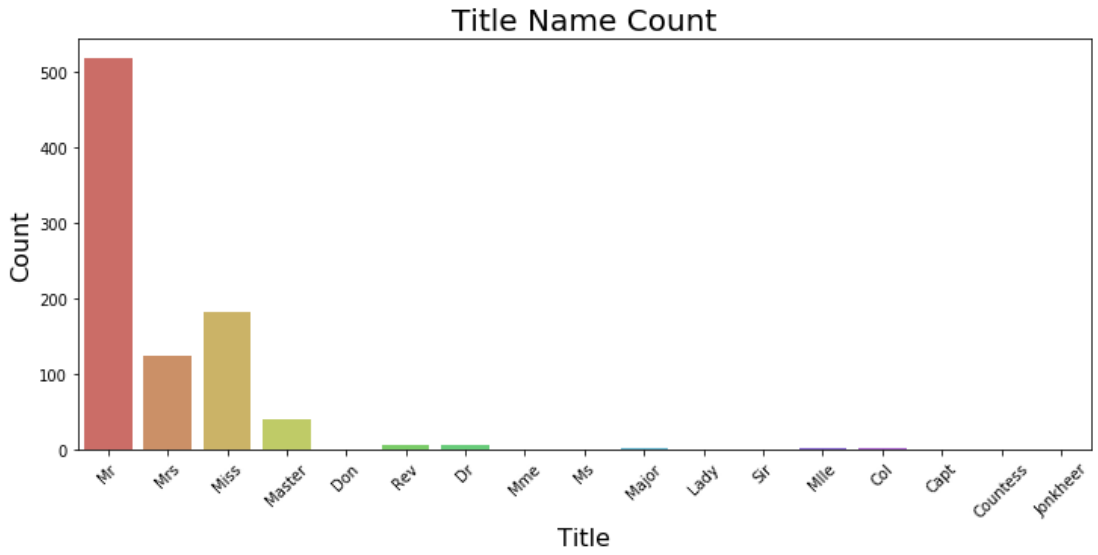
```
   PassengerId  Survived  Pclass  \
0            1         0       3
1            2         1       1
2            3         1       3
3            4         1       1
4            5         0       3

                                                Name     Sex   Age  SibSp  \
0                            Braund, Mr. Owen Harris    male  22.0      1
1  Cumings, Mrs. John Bradley (Florence Briggs Th...  female  38.0      1
2                             Heikkinen, Miss. Laina  female  26.0      0
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
4                           Allen, Mr. William Henry    male  35.0      0

   Parch            Ticket     Fare Cabin Embarked
0      0         A/5 21171   7.2500   NaN        S
1      0          PC 17599  71.2833   C85        C
2      0  STON/O2. 3101282   7.9250   NaN        S
3      0            113803  53.1000  C123        S
4      0            373450   8.0500   NaN        S
```

6.4 Επεξεργασία και Οπτικοποίηση Δεδομένων

Για μια νέα προσέγγιση στα δεδομένα θα επεξεργαστούμε μερικά χαρακτηριστικά (features).

Θα ξεκινήσουμε από το feature Name όπου παρατηρούμε ότι έχουμε τίτλους και άλλες προσφωνήσεις. Είναι ένα feature που σχετίζεται άρρηκτα με το ενδεχόμενο επιβίωσης. Θα προσπαθήσουμε στη συνέχεια να ομαδοποιήσουμε μερικά από αυτά ώστε να μειωθεί η πολυπλοκότητα του πειράματος.

Εδώ φαίνεται με την βοήθεια regular expressions το συνολικό μέγεθος των τίτλων και των προσφωνήσεων:



Σχήμα 15: Καταμέτρηση των τίτλων και προσφωνήσεων

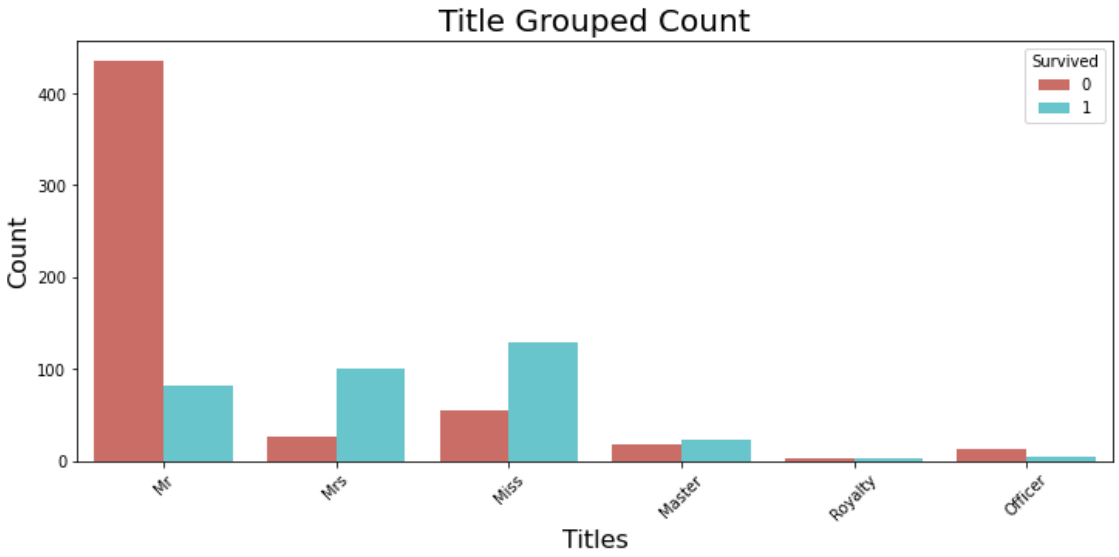
Κάνουμε το ίδιο για το df_test.

Εν συνεχεία, θα κάνουμε grouping (ομαδοποίηση) μερικών εξ αυτών όπου τελικά θα εντοπίσουμε την κοινωνική τάξη τους.

Παρατηρούμε την πιθανότητα επιβίωσης βασισμένοι στα ακριβώς παραπάνω. Ήδη ως γνωστόν θα δούμε ότι τα γυναικόπαιδα είχαν προτεραιότητα:

Πίνακας 7: Πιθανότητα επιβίωσης σε σχέση με τον τίτλο και την προσφώνηση

```
Chances to survive based on titles:
Title
Master     0.575000
Miss       0.701087
Mr         0.156673
Mrs        0.795276
Officer    0.277778
Royalty    0.500000
Name: Survived, dtype: float64
```



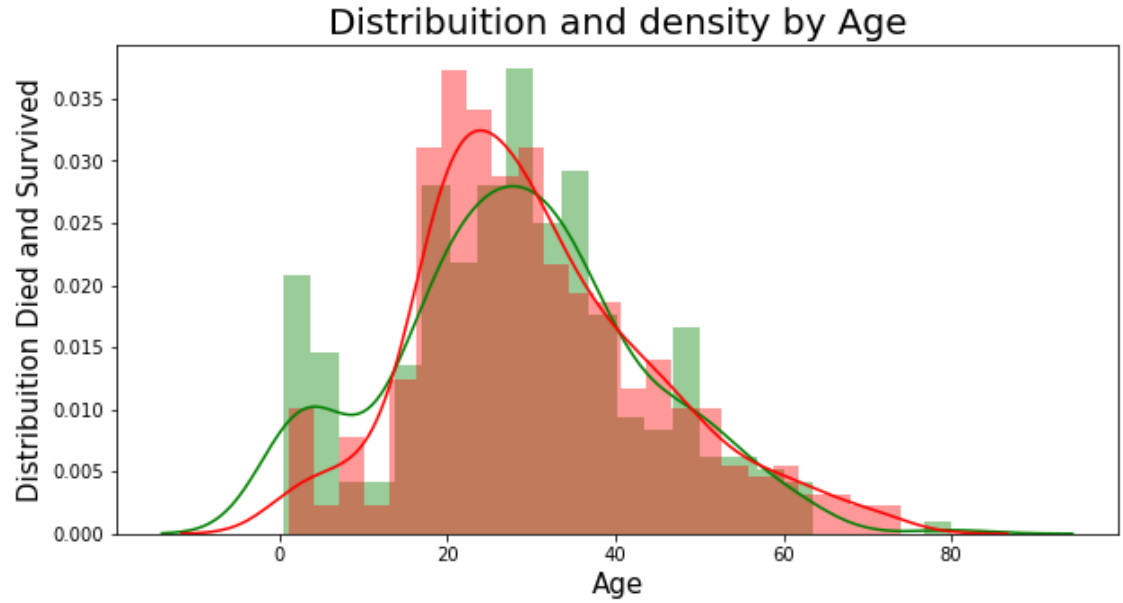
Σχήμα 16: Καταμέτρηση επιβίωσης με βάση τον τίτλο και την προσφώνηση

Στη συνέχεια, θα ασχοληθούμε με το feature Age όπου φαίνεται ότι υπάρχουν τιμές null. Θα προσπαθήσουμε να συμπληρώσουμε αυτά τα πεδία με στατιστικά εργαλεία.

Στο σημείο αυτό βλέπουμε δύο επικαλυπτόμενες κατανομές με βασικό χαρακτηριστικό την ηλικία, μια πράσινη και μια κόκκινη

- Πράσινη, είναι η επιβίωση
- Κόκκινη, είναι η μη επιβίωση

Θα αποφύγουμε τις null τιμές για αυτό το διάγραμμα και αμέσως βλέπουμε ότι οι μικρές ηλικίες έχουν μεγαλύτερη πιθανότητα επιβίωσης:



Σχήμα 17: Κατανομή επιβίωσης με βάση την ηλικία

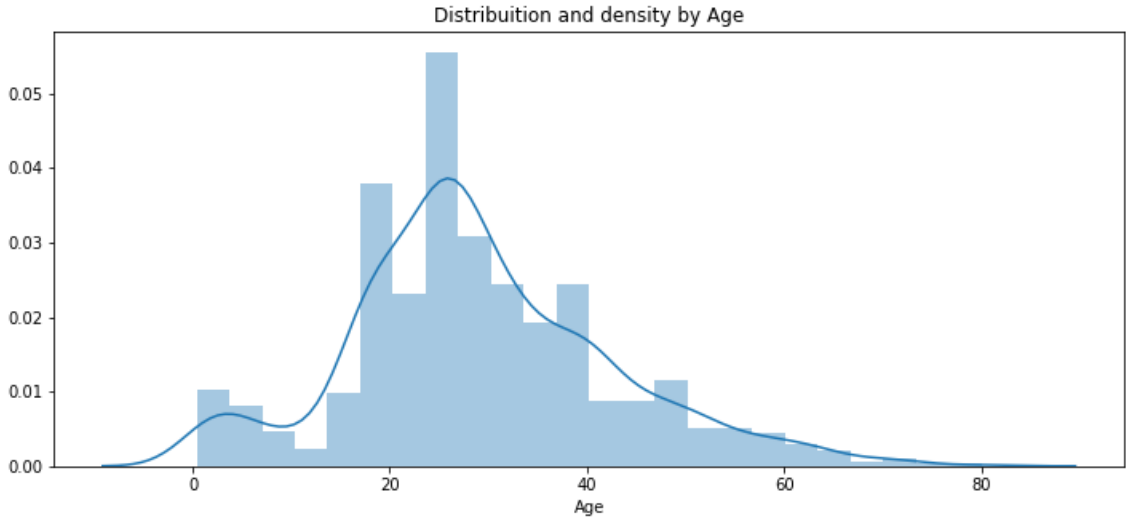
Σύμφωνα με τα παραπάνω θα ομαδοποιήσουμε τη διάμεσο της ηλικίας με βάση το Sex, Pclass και Title. Η τιμή της διαμέσου φαίνεται στα δεξιά:

Πίνακας 8: Ομαδοποίηση διαμέσου ηλικίας με βάση το φύλο, την τάξη εισιτηρίου και τον τίτλο

```
Sex     Pclass  Title
female  1       Miss       30.0
                Mrs        40.0
                Officer    49.0
                Royalty    48.0
        2       Miss       24.0
                Mrs        31.5
        3       Miss       18.0
                Mrs        31.0
male    1       Master      4.0
                Mr         40.0
                Officer    51.0
                Royalty    40.0
        2       Master      1.0
                Mr         31.0
                Officer    46.5
        3       Master      4.0
                Mr         26.0
Name: Age, dtype: float64
```

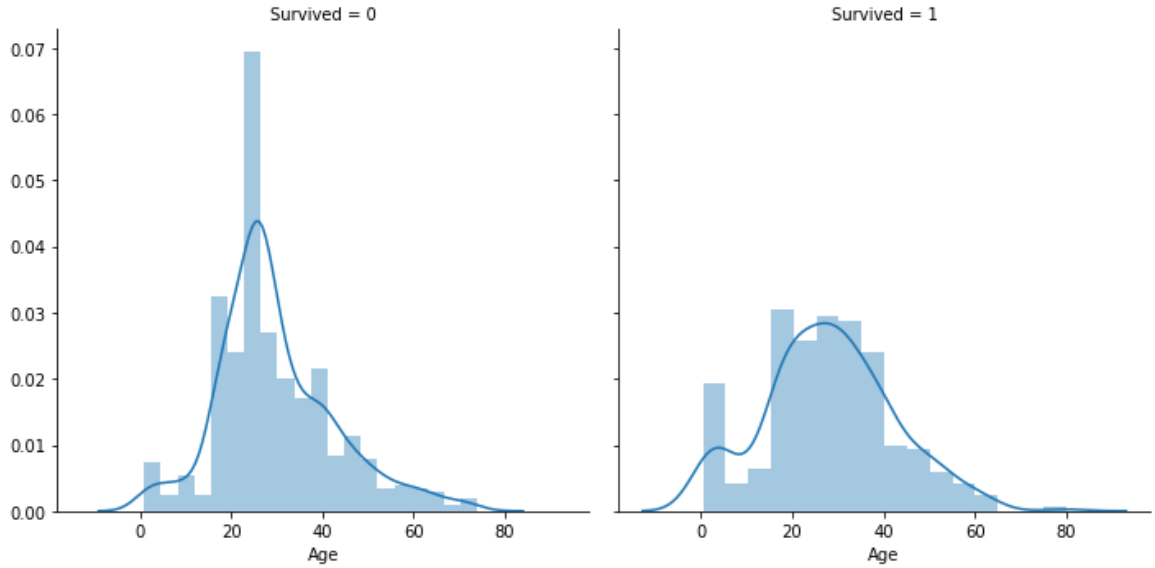
Μετά, θα συμπληρώσουμε με βάση τη διάμεσο όλες τις null τιμές.

Πάμε να δούμε το μετασχηματισμένο χαρακτηριστικό:



Σχήμα 18: Κατανομή ηλικίας έπειτα από μετασχηματισμό

Και σε ξεχωριστά διαγράμματα:



Σχήμα 19: Κατανομή ηλικίας έπειτα από μετασχηματισμό σε σχέση με την επιβίωση

Σε αυτό το σημείο θα κάνουμε κατηγοριοποίηση στο Age όπως κάναμε και στο Title με παρόμοια λογική. Βλέπουμε τις νέες κατηγορίες μας:

Πίνακας 9: Κατηγοριοποίηση του χαρακτηριστικού ηλικία

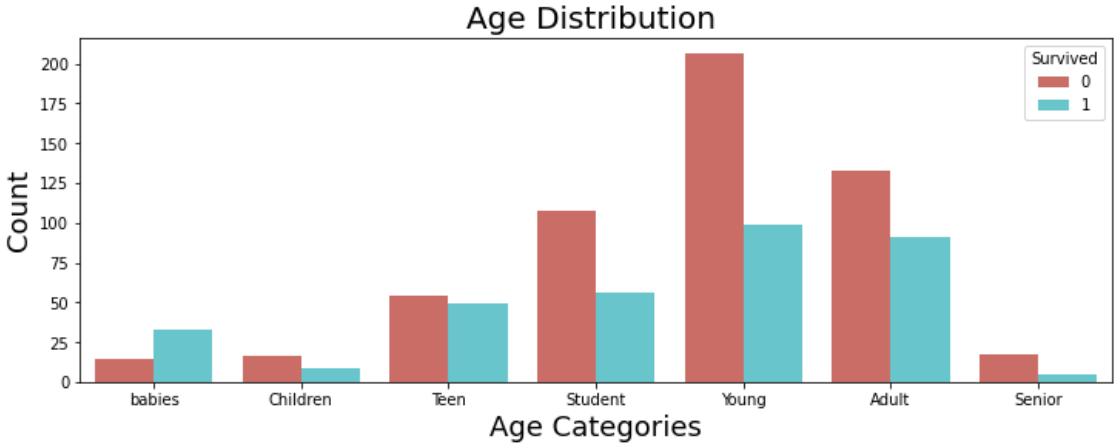
```
0    Student
1      Adult
2      Young
3      Young
4      Young
Name: Age_cat, dtype: category
Categories (7, object): ['babies' < 'Children' < 'Teen' < 'Student' <
'Young' < 'Adult' < 'Senior']
```

Κάνουμε το ίδιο για το df_test.

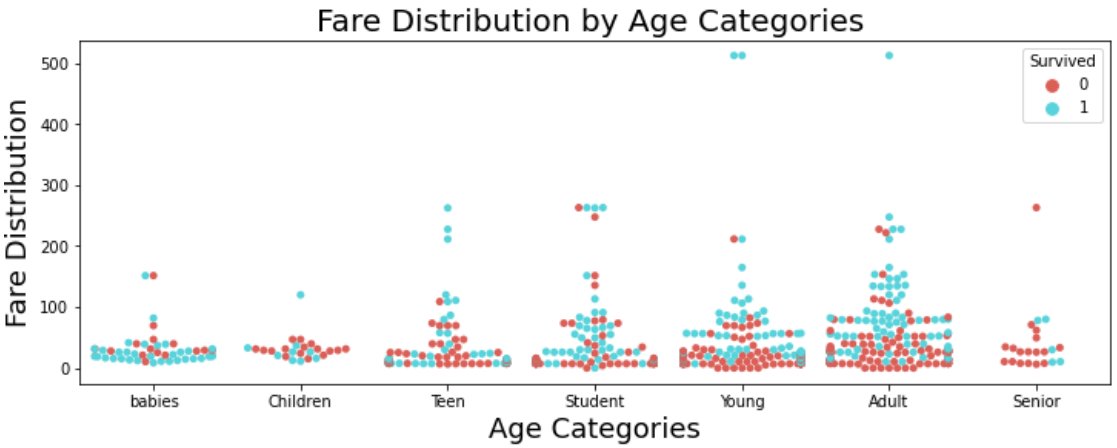
Παρακάτω βλέπουμε σύμφωνα με τις νέες κατηγορίες που φτιάξαμε να γεμίζουν με τα data από το Age. Τα διαγράμματα αυτά δείχνουν πιο καθαρά πλέον τα στοιχεία μας.

Πίνακας 10: Καταμέτρηση επιβατών που επιβίωσαν με βάση τις κατηγορίες ηλικίας

Survived	0	1
Age_cat		
babies	15	33
Children	16	9
Teen	54	49
Student	108	56
Young	206	99
Adult	133	91
Senior	17	5



Σχήμα 20: Καταμέτρηση επιβίωσης σε σχέση με τις κατηγορίες ηλικίας



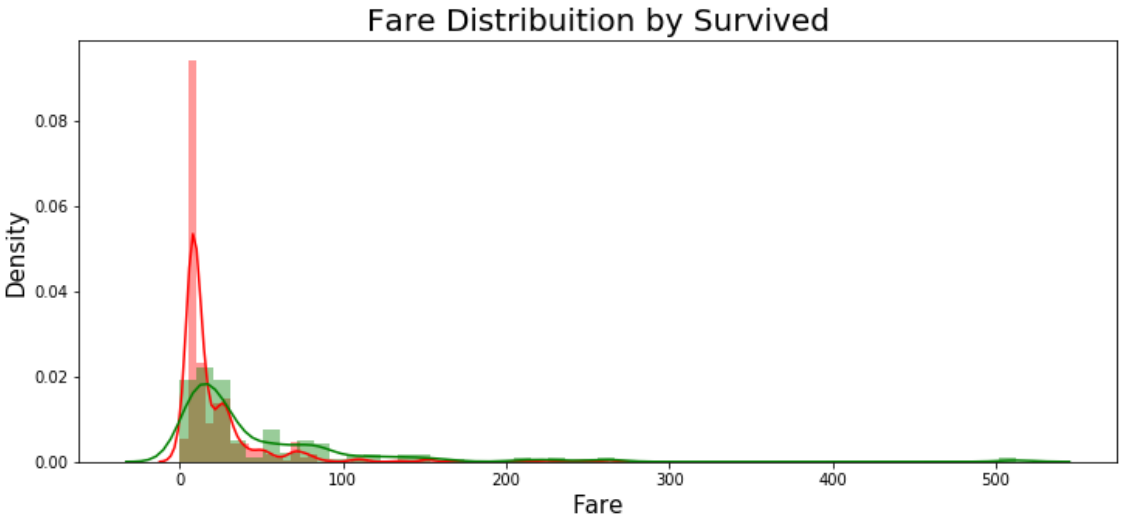
Σχήμα 21: Κατανομή ναύλου σε σχέση με τις κατηγορίες ηλικίας

Παράλληλα, θα ασχοληθούμε με το χαρακτηριστικό Pclass σε συνδυασμό με το Age_cat, δηλαδή την τάξη εισιτηρίου με την ηλικιακή κατηγορία. Βλέπουμε τη μέση τιμή του ναύλου:

Πίνακας 11: Μέση τιμή ναύλου ανά κατηγορία ηλικίας και τάξη εισιτηρίου

	mean						
Age_cat	babies	Children	Teen	Student	Young	Adult	Senior
Pclass							
1	128.319	120	122.538	113.002	92.4441	72.5561	59.969
2	28.1795	30.5625	21.1726	24.588	17.382	19.7606	10.5
3	23.7953	27.3262	15.1472	8.90337	12.7273	13.3342	7.82

Εδώ φαίνεται η κατανομή του ναύλου σε σχέση με την επιβίωση ή μη των επιβατών:

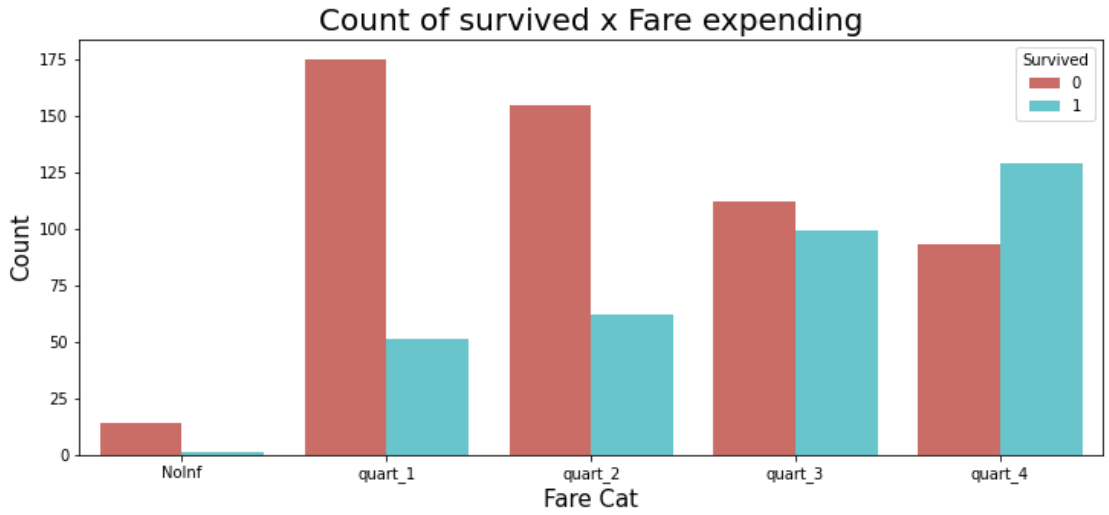


Σχήμα 22: Κατανομή ναύλου σε σχέση με την επιβίωση

Χωρίζουμε κατά παρεμφερή τρόπο το χαρακτηριστικό Fare σε τέσσερα κομμάτια με αύξοντα τιμή ναύλου αντίστοιχα. Για 15 στοιχεία δεν έχουμε τιμή και τα βάζουμε σε άλλη κατηγορία:

Πίνακας 12: Καταμέτρηση επιβατών με βάση τις κατηγορίες ναύλου

Survived	0	1
Fare_cat		
NoInf	14	1
quart_1	175	51
quart_2	155	62
quart_3	112	99
quart_4	93	129



Σχήμα 23: Καταμέτρηση επιβίωσης σε σχέση με τις κατηγορίες ναύλου

Κάνουμε το ίδιο για το df_test.

Σε αυτή τη φάση θα αντικαταστήσουμε τα παλιά με τα νέα features που δημιουργήσαμε, τα οποία είναι καλύτερα δομημένα και συνεπώς θα βοηθήσουν τα Μοντέλα μας μετέπειτα. Τα πρώτα 5 data με την νέα δομή:

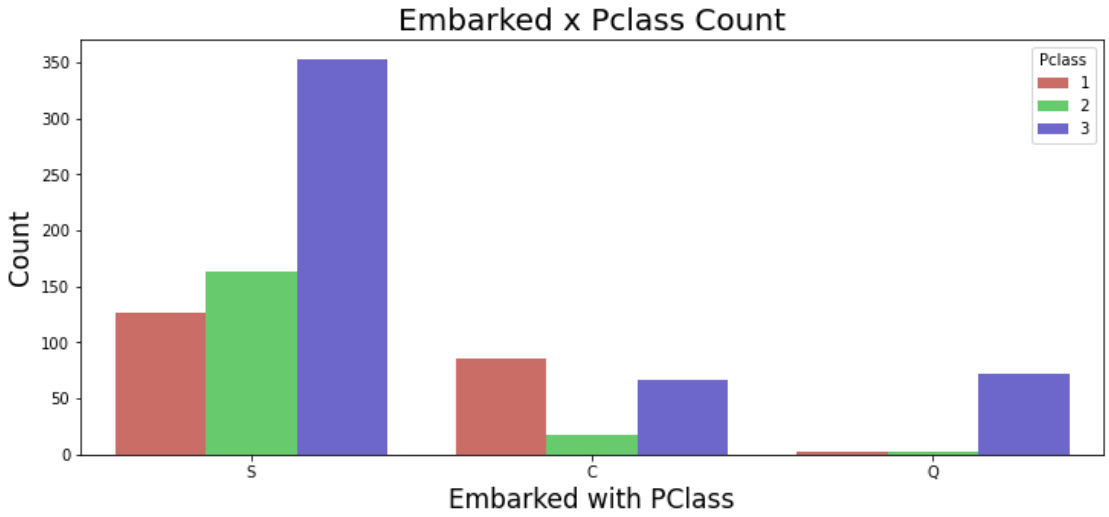
Πίνακας 13: Πρώτα 5 στοιχεία με την νέα δομή

	PassengerId	Survived	Pclass	Sex	SibSp	Parch	Embarked	Title	Age_cat	Fare_cat
0	1	0	3	male	1	0	S	Mr	Student	quart_1
1	2	1	1	female	1	0	C	Mrs	Adult	quart_4
2	3	1	3	female	0	0	S	Miss	Young	quart_1
3	4	1	1	female	1	0	S	Mrs	Young	quart_4
4	5	0	3	male	0	0	S	Mr	Young	quart_2

Εν συνεχεία, θα εξετάσουμε το χαρακτηριστικό Pclass και Embarked, δηλαδή τι κλάση είχαν επιλέξει και από ποιο λιμάνι επιβιβάστηκαν, ώστε να δούμε αν μπορούμε να εξάγουμε κάποια πληροφορία:

Πίνακας 14: Καταμέτρηση επιβατών στα λιμάνια επιβίβασης με βάση την τάξη εισιτηρίου

Embarked	C	Q	S
Pclass			
1	85	2	127
2	17	3	164
3	66	72	353



Σχήμα 24: Καταμέτρηση επιβατών στα λιμάνια επιβίβασης σε σχέση με την τάξη εισιτηρίου

Παρακάτω βλέπουμε την επιβίωση ή μη με βάση το Embarked και το Pclass. Με την πρώτη ματιά φαίνεται ότι οι επιβάτες 3ης κλάσης που επιβιβάστηκαν στο Southampton είχαν μεγάλη πιθανότητα να μην επιβιώσουν.

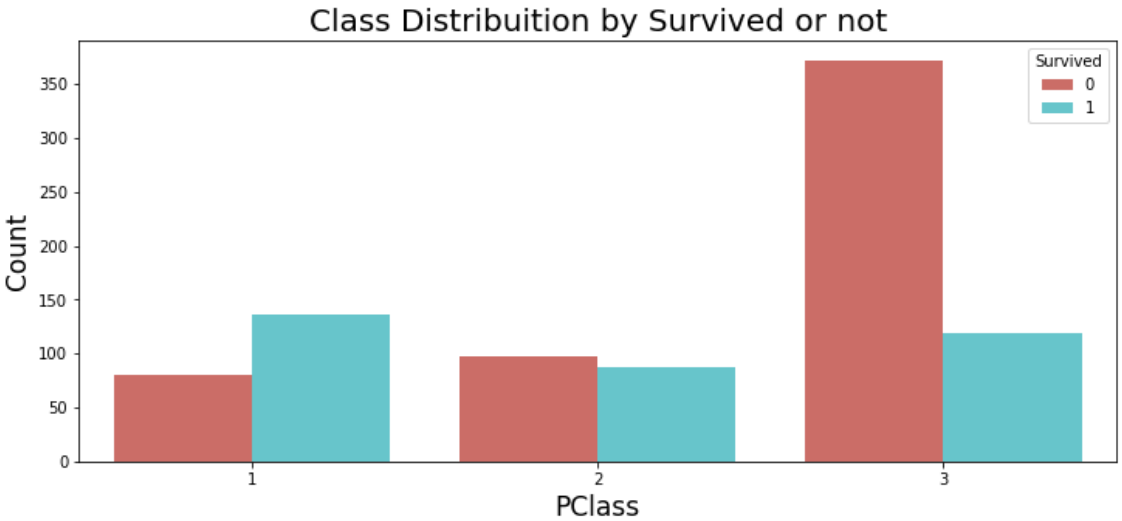
Πίνακας 15: Καταμέτρηση επιβίωσης με βάση τα λιμάνια επιβίβασης

Embarked	C	Q	S
Survived			
0	75	47	427
1	93	30	219

Σχήμα 25: Καταμέτρηση επιβίωσης σε σχέση με τα λιμάνια επιβίβασης

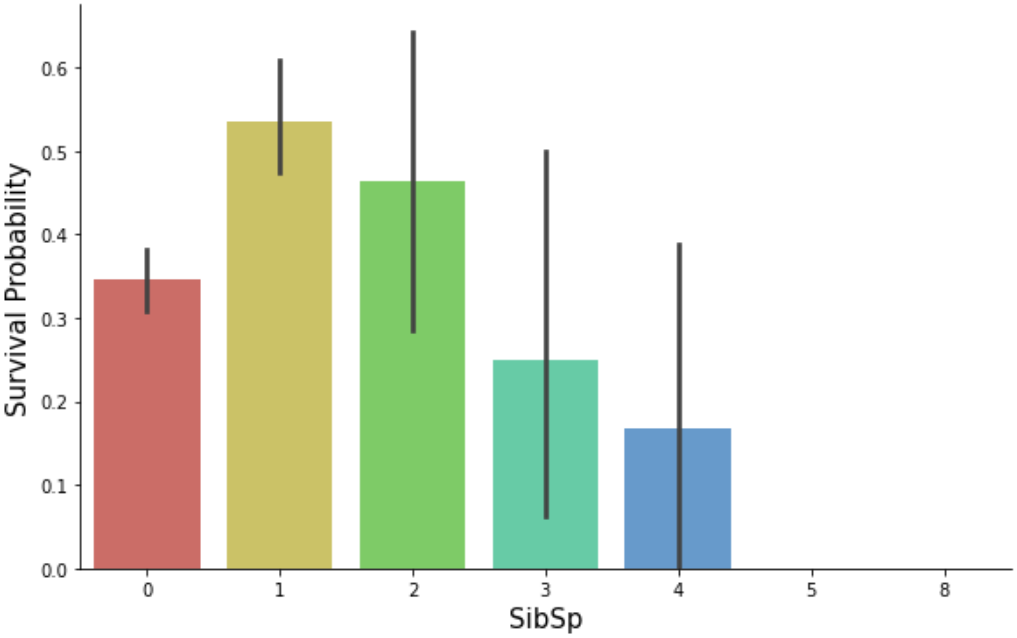
Πίνακας 16: Καταμέτρηση επιβίωσης με βάση την τάξη εισιτηρίων

Pclass	1	2	3
Survived			
0	80	97	372
1	136	87	119

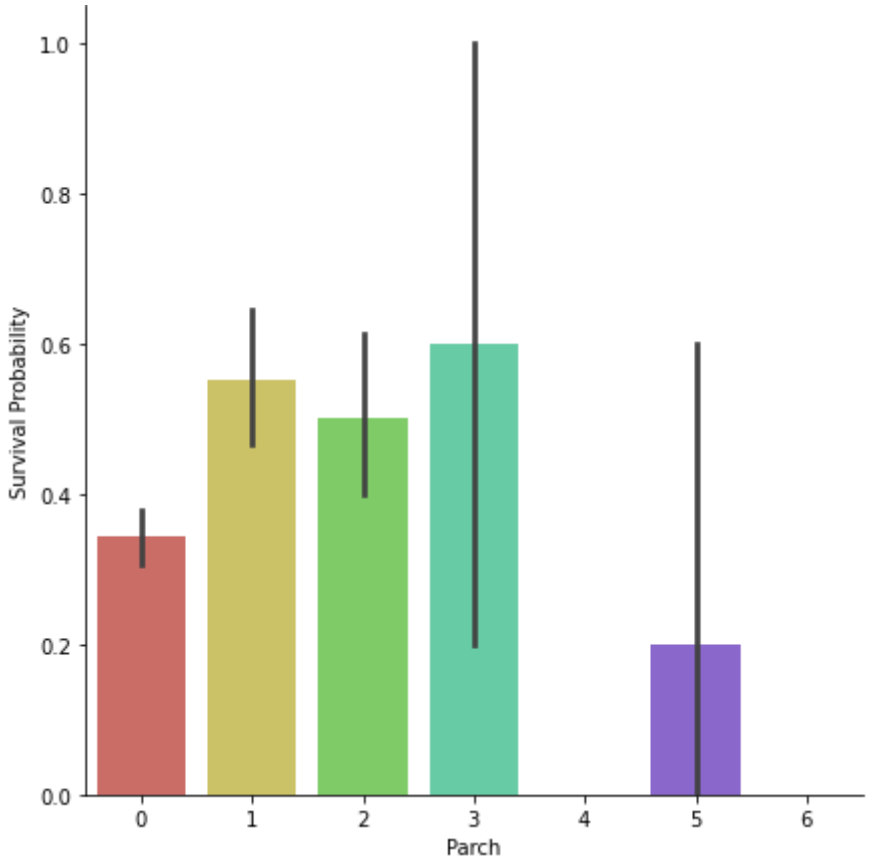


Σχήμα 26: Καταμέτρηση επιβίωσης σε σχέση με την τάξη εισιτηρίων

Τελικά, θα μελετήσουμε τα χαρακτηριστικά SibSp και Parch σε σχέση με την επιβίωση ή μη, δηλαδή ο αριθμός αδερφιών/συζύγων και γονέων/παιδιών. Ενδιαφέρον είναι το γεγονός ότι οι επιβάτες με 1 ή 2 αδέρφια/συζύγους και οι μικρές οικογένειες (1,2) έχουν μεγαλύτερες πιθανότητες επιβίωσης. Επίσης παρατηρείται μεγάλη διακύμανση στην τιμή 3 του Parch.



Σχήμα 27: Πιθανότητα επιβίωσης σε σχέση με τον αριθμό αδερφιών/συζύγων

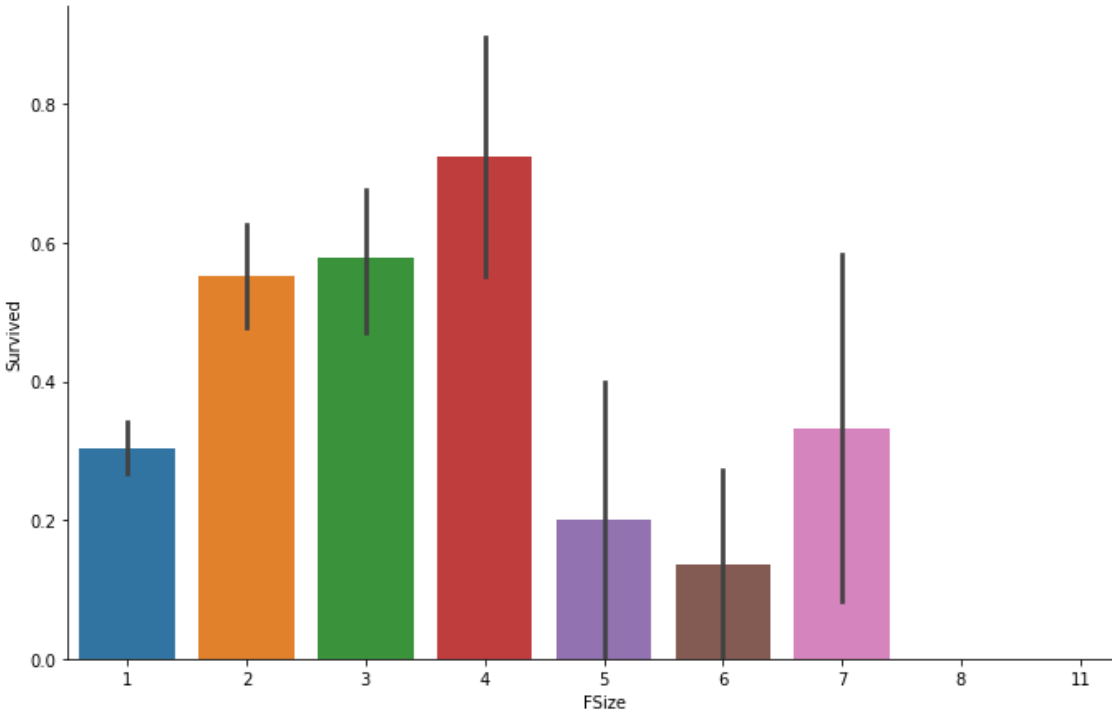


Σχήμα 28: Πιθανότητα επιβίωσης σε σχέση με τον αριθμό γονέων/παιδιών

Συνεπώς θα ήταν χρήσιμο να μειώσουμε την πολυπλοκότητα και να μετασχηματίσουμε τα δύο αυτά χαρακτηριστικά σε ένα Fsize, όπου θα προσμετράτε και ο ίδιος ο επιβάτης.

Πίνακας 17: Καταμέτρηση επιβίωσης με βάση τις κατηγορίες του μεγέθους των οικογενειών

Survived	0	1
FSize		
1	374	163
2	72	89
3	43	59
4	8	21
5	12	3
6	19	3
7	8	4
8	6	0
11	7	0

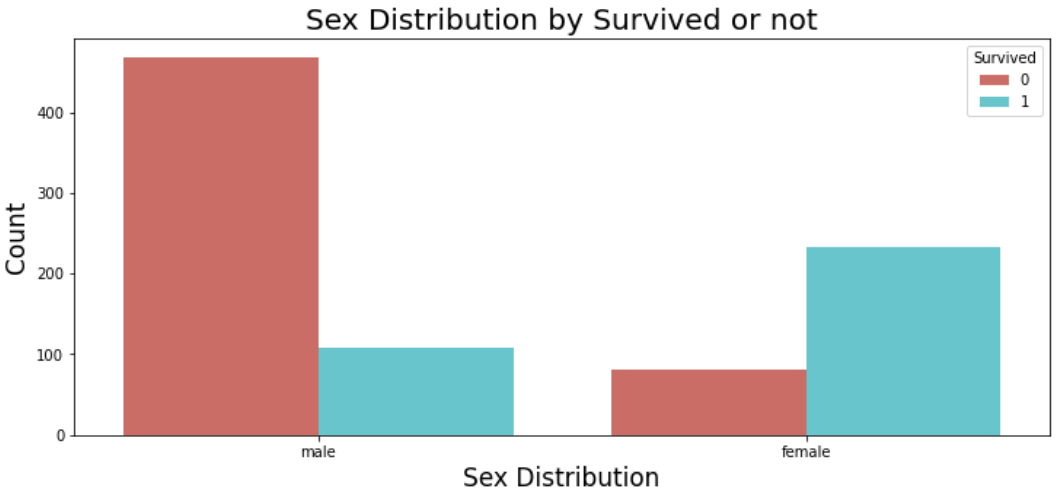


Σχήμα 29: Πιθανότητα επιβίωσης σε σχέση με το μέγεθος οικογένειας

Φτάνοντας στα τελικό χαρακτηριστικό Sex, όπου ως γνωστών καθόρισε σε μεγάλο βαθμό το αποτέλεσμα. Φαίνεται ότι μεγαλύτερη πιθανότητα επιβίωσης είχαν οι γυναίκες.

Πίνακας 18: Καταμέτρηση επιβίωσης με βάση το φύλο

Sex	female	male
Survived		
0	81	468
1	233	109

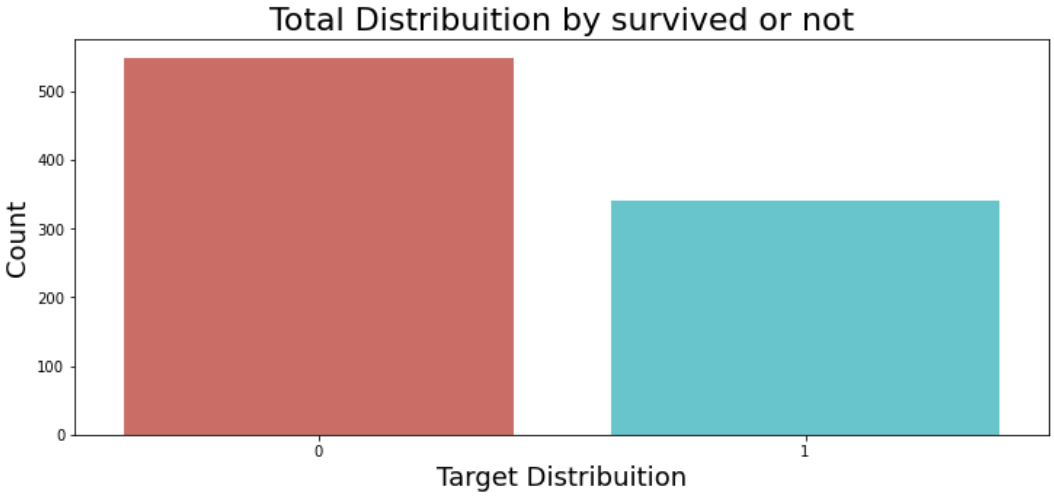


Σχήμα 30: Καταμέτρηση επιβίωσης σε σχέση με το φύλο

Συνολικά, για όλα τα δεδομένα έχουμε:

Πίνακας 19: Γενική καταμέτρηση επιβίωσης

```
Total of Survived or not:
Survived
0    549
1    342
Name: PassengerId, dtype: int64
```



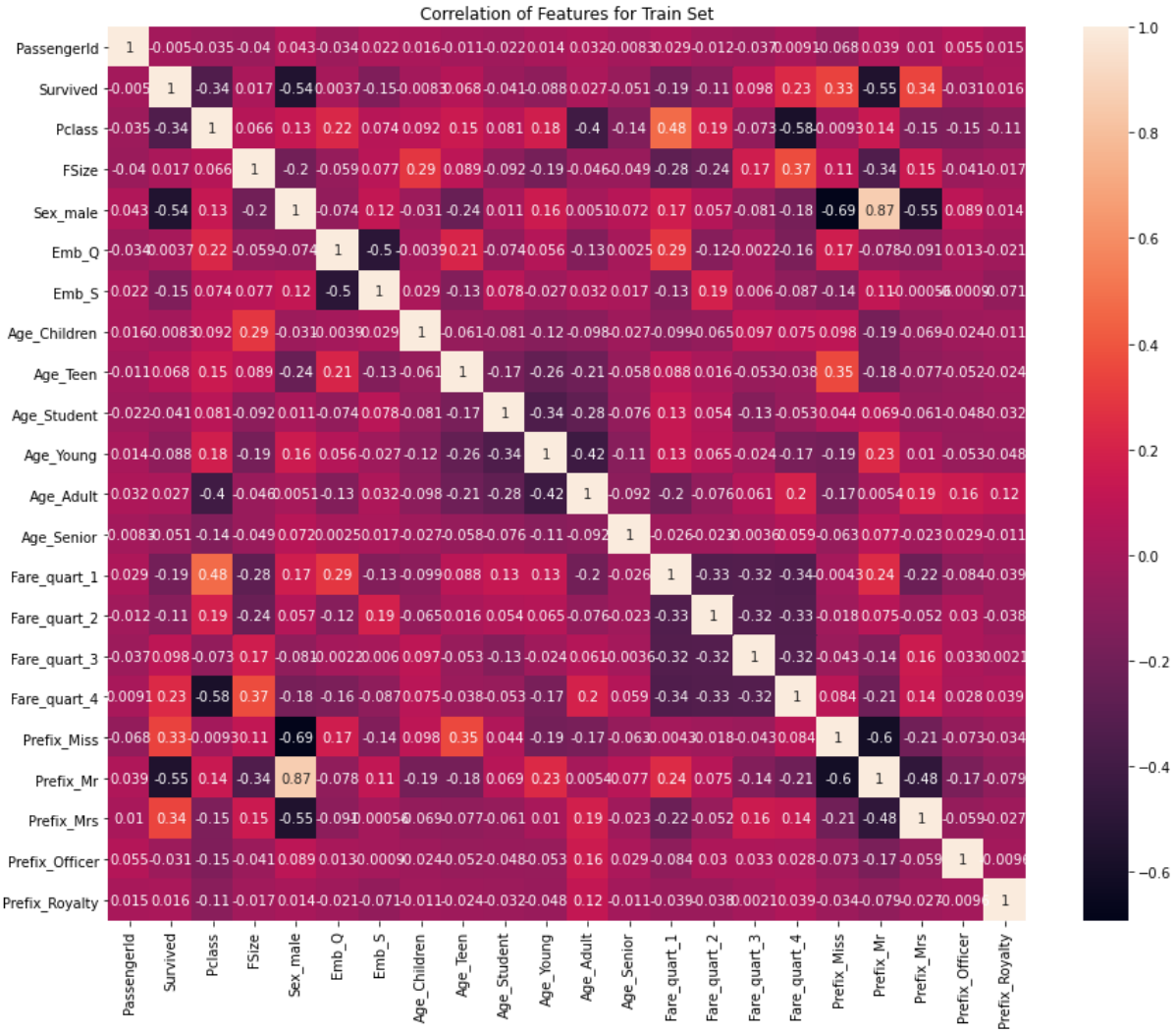
Σχήμα 31: Γενική καταμέτρηση επιβίωσης

Έχουμε την εξής νέα μορφολογία :

Πίνακας 20: Νέα μορφολογία του συνόλου δεδομένων

	PassengerId	Survived	Pclass	Sex	SibSp	Parch	Embarked	Title	Age_cat	Fare_cat
0	1	0	3	male	1	0	S	Mr	Student	quart_1
1	2	1	1	female	1	0	C	Mrs	Adult	quart_4
2	3	1	3	female	0	0	S	Miss	Young	quart_1
3	4	1	1	female	1	0	S	Mrs	Young	quart_4
4	5	0	3	male	0	0	S	Mr	Young	quart_2

Τελικά, θα δούμε την Συσχέτιση (Correlation) μεταξύ των χαρακτηριστικών, αφού μετασχηματίσουμε όλα τα στοιχεία μας σε δυαδικά (binary), 0 και 1. Προκύπτουν όπως φαίνεται 22 χαρακτηριστικά από την συνολική επεξεργασία.



Σχήμα 32: Θερμικός χάρτης συσχέτισης χαρακτηριστικών

6.5 Αλγόριθμοι πρόβλεψης - Μοντέλα

Εφόσον έχουμε επεξεργαστεί και δομήσει τα δεδομένα μας, είναι πλέον εφικτό να περάσουμε στο επόμενο στάδιο του πειράματος. Σε αυτό το σημείο θα εφαρμόσουμε τους αλγόριθμους πρόβλεψης – Μοντέλα έπειτα από μια προεπεξεργασία.

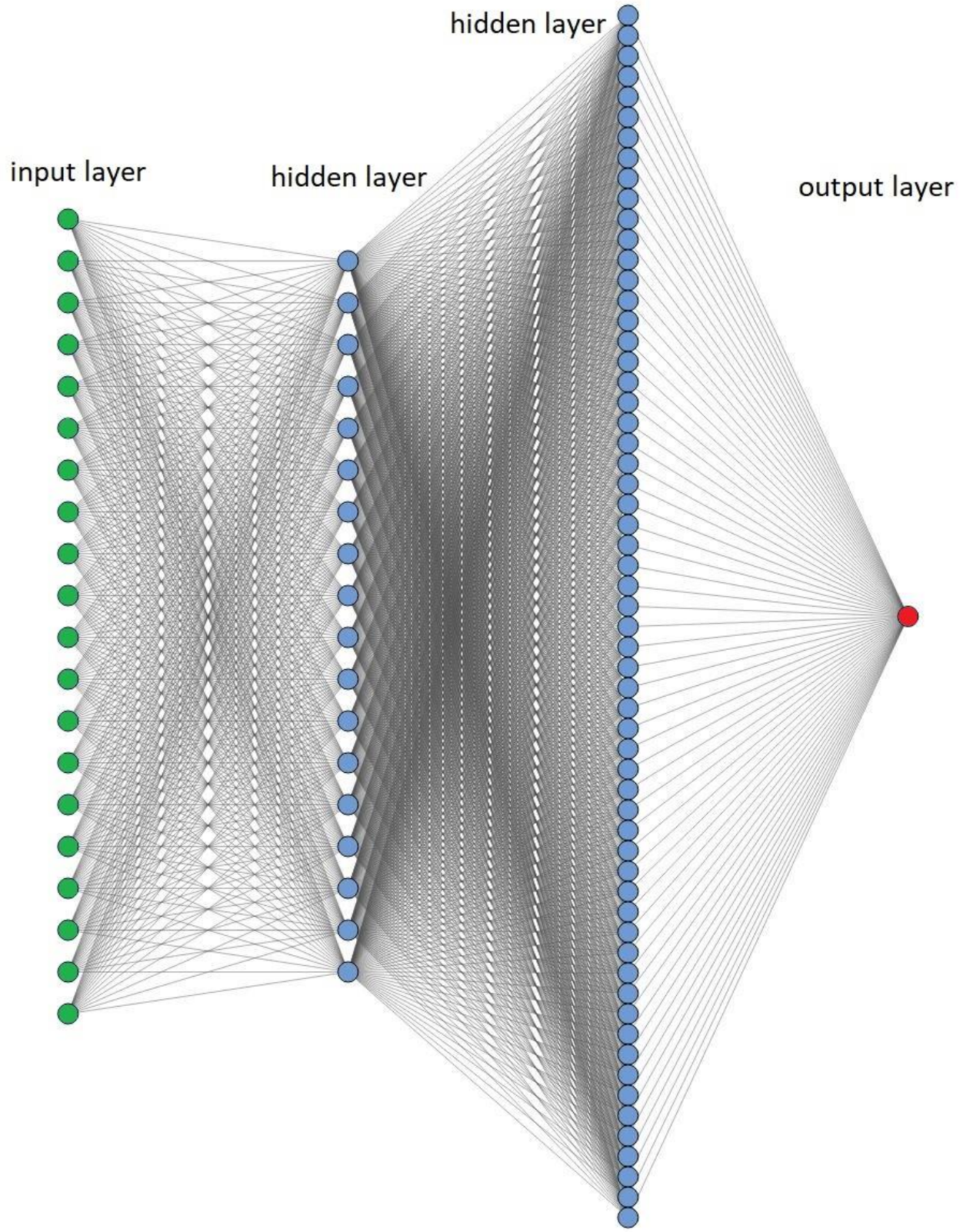
- Decision Tree
- Naive Bayes classifier
- Logistic Regression
- KNN or k-Nearest Neighbors
- Support Vector Machines
- Perceptron
- Neural Network με ανατομία ως εξής:
    - 20 νευρώνες εισόδου, όσα είναι και τα χαρακτηριστικά μας
    - 18 κρυφοί νευρώνες υπολογισμού
    - 1 στρώμα κανονικοποίησης Dropout
    - 60 κρυφοί νευρώνες υπολογισμού
    - 1 στρώμα κανονικοποίησης Dropout
    - 1 νευρώνας εξόδου
    - Συνάρτηση Ενεργοποίησης είναι η ReLU (Rectified Linear Unit) στα πρώτα στρώματα και Sigmoid στο στρώμα εξόδου
    - Συνάρτηση Απώλειας είναι η Binary Cross Entropy
    - Αλγόριθμος βελτιστοποίησης είναι ο Adam με learning rate 0.001
    - 200 εποχές (epochs)

Οι νευρώνες κανονικοποίησης είναι σημαντικό μέρος του Νευρωνικού Δικτύου καθώς μειώνουν την υπερ-εκπαίδευση (overfitting), όπου θα αναλύσουμε παρακάτω. Μετά από κάθε κρυφό στρώμα υπάρχει ένα στρώμα Dropout.

Πίνακας 21: Αρχιτεκτονική Νευρωνικού Δικτύου

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 18)                378
_________________________________________________________________
dropout (Dropout)            (None, 18)                0
_________________________________________________________________
dense_1 (Dense)              (None, 60)                1140
_________________________________________________________________
dropout_2 (Dropout)          (None, 60)                0
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 61
=================================================================
Total params: 1,579
Trainable params: 1,579
Non-trainable params: 0
```

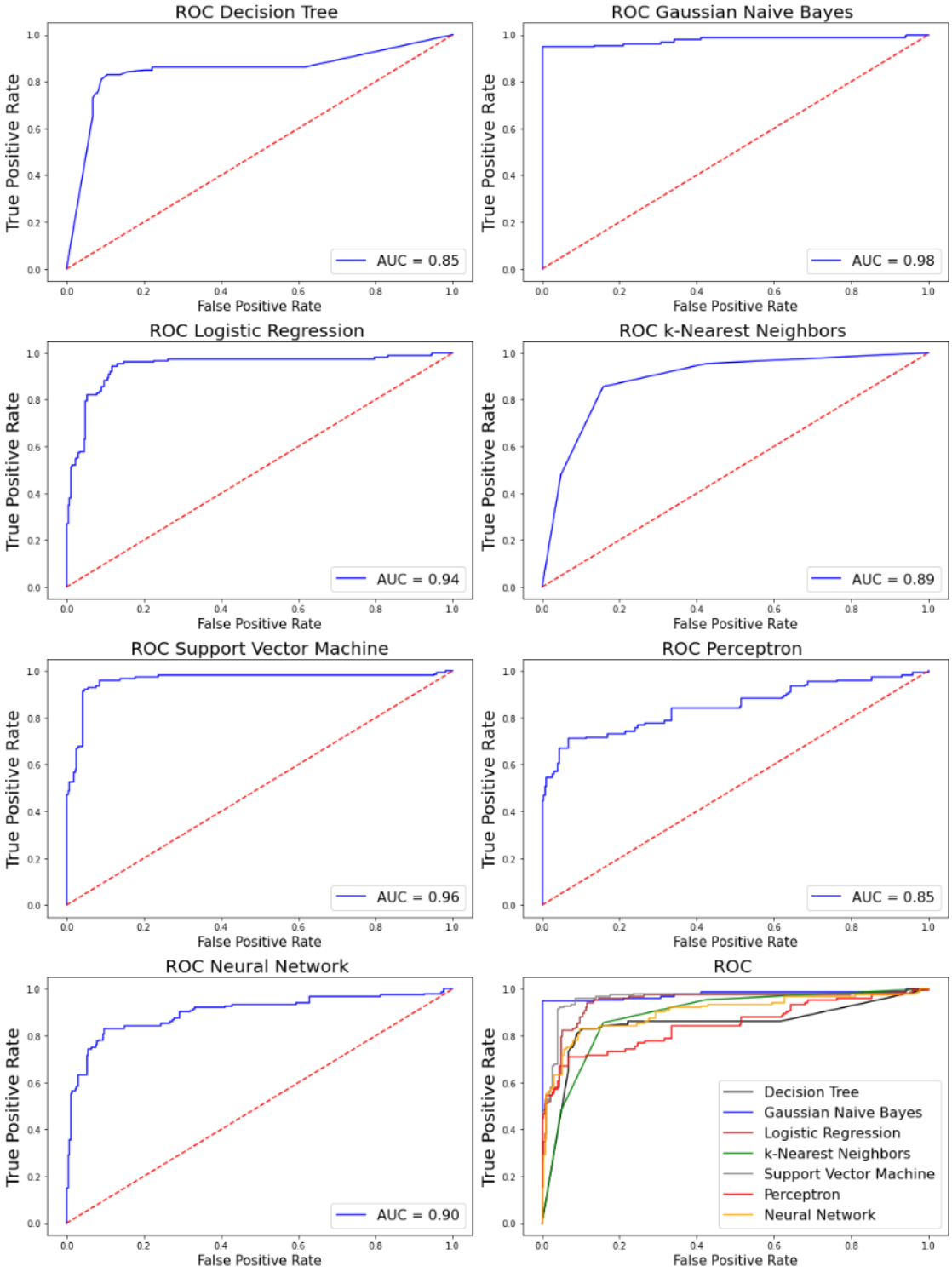
Οπτικοποίηση του Νευρωνικού Δικτύου:



Σχήμα 33: Απεικόνιση του Νευρωνικού Δικτύου

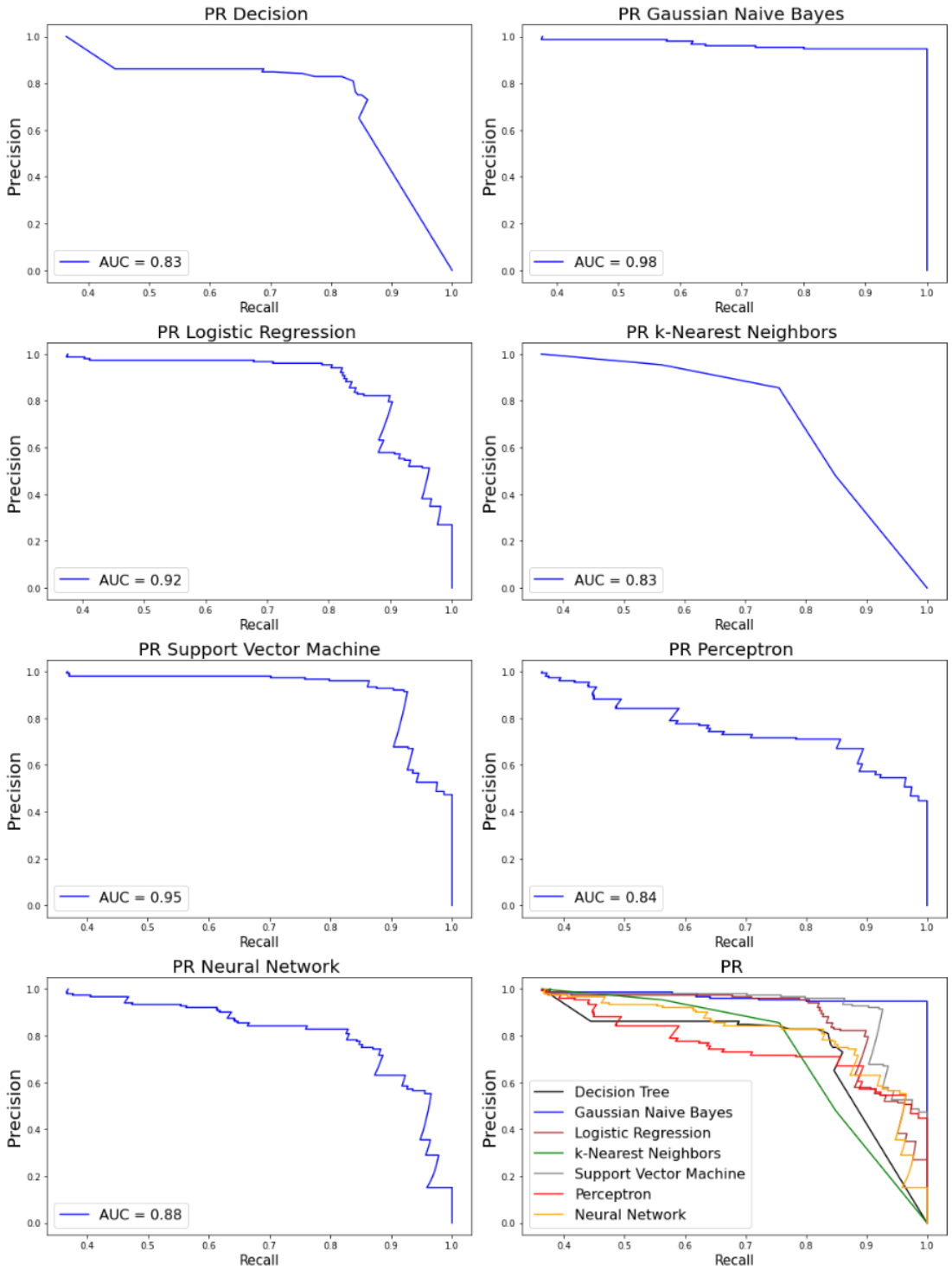
6.6 Σύγκριση Μοντέλων με βάση την Ακρίβεια

Στο τελευταίο αυτό κεφάλαιο, θα εξετάσουμε μερικά στοιχεία αξιολόγησης για τα Μοντέλα μας.
Αρχικά, βλέπουμε τις καμπύλες ROC για κάθε Μοντέλο, παρατηρώντας την διαγνωστική ικανότητα αυτών. Το μέτρο AUC (Area Under Curve) (χώρος κάτω από την καμπύλη) δείχνει αυτή την ικανότητα. Όσο υψηλότερη η τιμή του AUC τόσο το καλύτερο με όριο τιμής το 1. Αν φτάσει την τιμή 1 τότε το Μοντέλο είναι τέλειο. Παρατηρούμε καλή έως πολύ απόδοση για τα Μοντέλα μας.



Σχήμα 34: Καμπύλες ROC για κάθε Μοντέλο

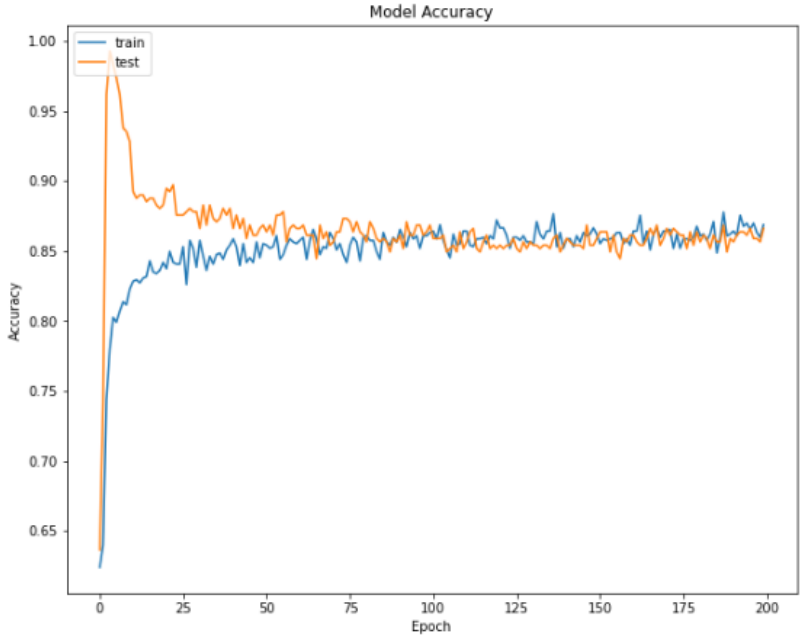
Παρακάτω, βλέπουμε τις καμπύλες PR για κάθε Μοντέλο. Παρατηρούμε, αντίστοιχα ότι έχουμε επιθυμητές καμπύλες, που σημαίνει ότι τα Μοντέλα μας διαχωρίζουν πολύ καλά τις δύο καταστάσεις: επιβίωση ή μη. Βλέπουμε, όπως ήταν αναμενόμενο σχεδόν απόλυτη συμμετρία με τις καμπύλες ROC



Σχήμα 35: Καμπύλες PR για κάθε Μοντέλο

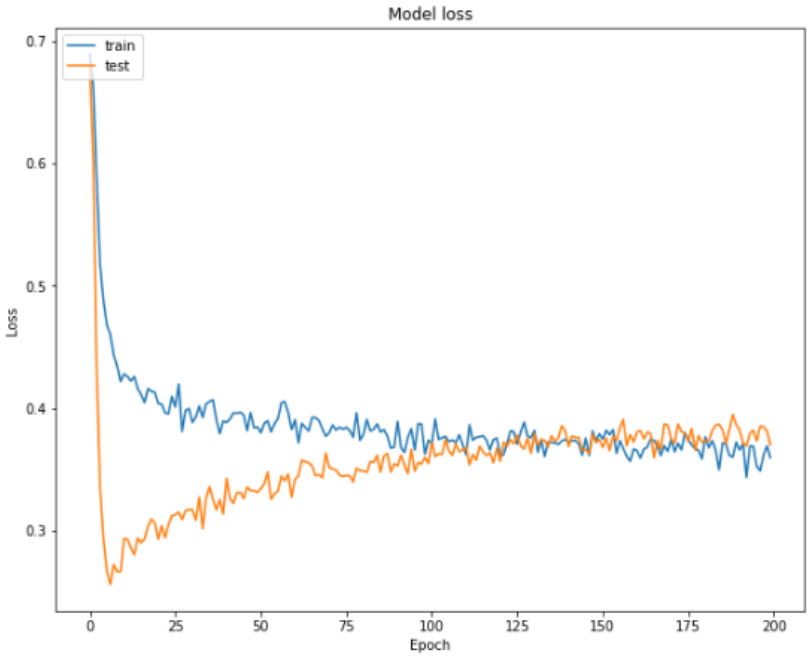
Μερικά επιπλέον στοιχεία για την αποτελεσματικότητα του Νευρωνικού Δικτύου που δημιουργήσαμε ενόσω εκπαιδευόταν :

Ακρίβεια (Accuracy) . Φαίνεται ότι δεν υπάρχει υπολογίσιμη υπερ-εκπαίδευση (overfitting). Αν ήταν μεγάλο το χάσμα θα σήμαινε ότι το Νευρωνικό Δίκτυο δεν είναι κατάλληλο για την κατηγοριοποίηση. Όσο μεγαλύτερο το overfitting τόσο το Μοντέλο ταιριάζει ακριβώς με τα δεδομένα εκπαίδευσης, αδυνατώντας στην πορεία να αποδώσει με ακρίβεια στα «αόρατα» δεδομένα, χάνοντας έτσι τον σκοπό του:



Σχήμα 36: Ακρίβεια Νευρωνικού Δικτύου κατά την διάρκεια της εκπαίδευσης

Απώλεια (Loss) . Και σε αυτό το διάγραμμα διακρίνουμε θετικά αποτελέσματα. Η καμπύλη καταδεικνύει μικρή απώλεια που σημαίνει ότι ο ρυθμός μάθησης είναι επιθυμητός :



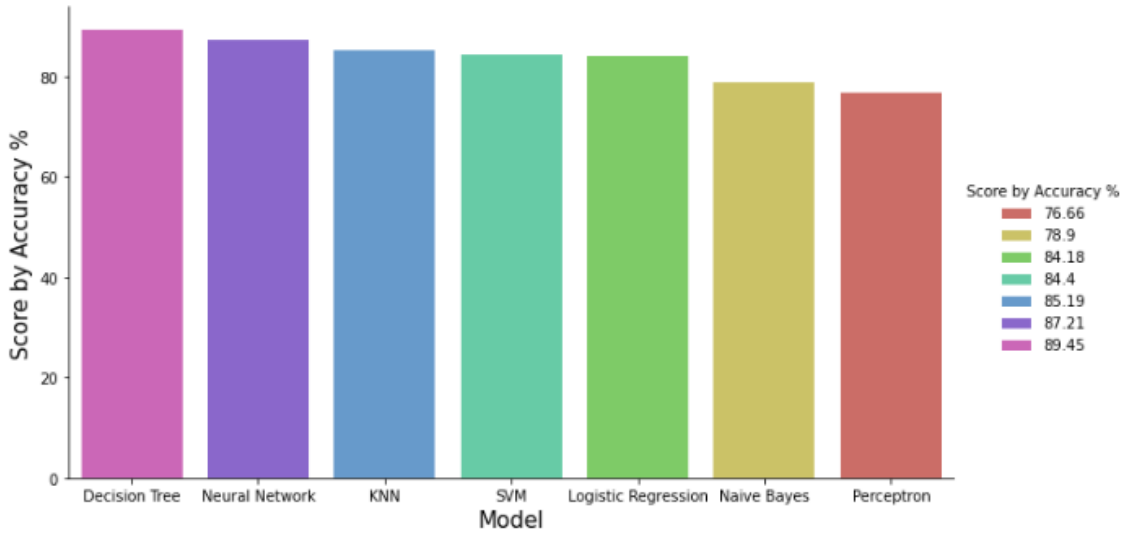
Σχήμα 37: Απώλεια Νευρωνικού Δικτύου κατά την διάρκεια της εκπαίδευσης

Τελικά, ένα από τα βασικότερα μέτρα η Ακρίβεια (Accuracy) μας δίνει μια καθαρή εικόνα για την απόδοση των Μοντέλων. Παρατηρούμε, πολύ καλή έως και αποδεκτή Ακρίβεια.

Πίνακας 22: Ακρίβεια Μοντέλων

	Model	Score by Accuracy %
5	Decision Tree	89.45
6	Neural Network	87.21
1	KNN	85.19
0	SVM	84.40
2	Logistic Regression	84.18
3	Naive Bayes	78.90
4	Perceptron	76.66

Στο παρακάτω σχήμα (Σχήμα 38) βλέπουμε σε γραφική παράσταση την Ακρίβεια ανά Μοντέλο. Παρατηρούμε, αμέσως, ότι τα Δέντρα Απόφασης και το Νευρωνικό μας Δίκτυο έχουν την δυνατότητα να κατατάσσουν τους επιβάτες με ιδιαίτερα υψηλές επιδόσεις.



Σχήμα 38: Ακρίβεια Μοντέλων

ΣΥΜΠΕΡΑΣΜΑΤΑ

Μέσα από την ενασχόλησή μας με τις συναρπαστικές τεχνολογίες της Μηχανικής Μάθησης, την επεξεργασία των δεδομένων και την υλοποίηση διάφορων Μοντέλων, καταλήγουμε στα ακόλουθα εξής συμπεράσματα:

- Η Μηχανική Μάθηση προσφέρει πολλές δυνατότητες στους χρήστες και στους προγραμματιστές, βοηθάει στην δημιουργία εφαρμογών και συστημάτων των οποίων η υλοποίηση ήταν ανέφικτη πριν από μερικά χρόνια.
- Η Μηχανική Μάθηση, όντας καινοτόμα και νεοεισαχθείσα, γίνεται όλο και πιο δημοφιλής στον χώρο της πληροφορικής παρέχοντας λύσεις και σε επιστήμες όπως η υγεία, οικονομία, κυβερνητικό έργο, πωλήσεις, marketing, κλπ.
- Το σύνολο δεδομένων της επιβίωση ή μη των επιβατών του Τιτανικού που χρησιμοποιήσαμε είναι ιδανικό για να γνωρίσει κανείς τον κόσμο της Μηχανικής Μάθησης, καθώς είναι σχετικά μικρό και κυριότερα περιέχει δεδομένα εύκολα ως προς την κατανόηση.
- Η αποτελεσματική εκπαίδευση των Μοντέλων είναι μια διαδικασία χρονοβόρα και σε πολλές περιπτώσεις απρόβλεπτη αφού εξαρτάται από πολλές συνιστώσες, όπως η αρχιτεκτονική αυτών, ο ρυθμός μάθησης και η ικανότητά τους να αυτοβελτιώνονται.
- Σημαντικό βάρος ενασχόλησης, όπως είδαμε, πρέπει να δοθεί στα ίδια τα δεδομένα, δηλαδή στη μορφολογία τους, στο μέγεθός τους και στην καθαριότητα τους από κενές ή διπλές τιμές. Η οπτικοποίηση τους σε σχήματα παίζει καθοριστικό ρόλο για την κατανόησή τους.
- Παρατηρήσαμε ότι μερικά Μοντέλα απέδωσαν καλύτερα από κάποια άλλα, χωρίς αυτό να σημαίνει ότι τα λιγότερο αποδοτικά είναι χειρότερα εξ' αυτών. Υπό άλλες συνθήκες δεν θα ήταν απίθανο να δούμε εντελώς άλλη ιεραρχία.
- Είδαμε, ωστόσο, μεγάλη διαγνωστική ικανότητα σε όλα τα Μοντέλα μέσω των καμπυλών ROC και των καμπυλών PR, κάτι που σηματοδοτεί την ευχρηστία και την αποτελεσματικότητα τους, καθιστώντας τα ως τουλάχιστον επιθυμητά.
- Τα 2 πρώτα Μοντέλα, τα Δέντρα Απόφασης και το Νευρωνικό Δίκτυο δικής μας αρχιτεκτονικής πέτυχαν ιδιαίτερα υψηλή επίδοση στην Ακρίβεια (>87%), που σημαίνει ότι αποφασίζει σωστά για το αν επιβίωσαν ή όχι περίπου 9 στους 10 επιβάτες
- Είναι σαγηνευτικό το γεγονός ότι ένας αλγόριθμος μπορεί να αναπτύξει κριτική σκέψη για ένα μικρό περιβάλλον ερεθισμάτων που δέχεται. Είναι ένα τεράστιο ορόσημο στην ιστορία των υπολογιστών. Όπως ένα νεογέννητο μωρό αναπτύσσει την σκέψη του με το πέρασμα των χρόνων έτσι και η Μηχανική Μάθηση ως υποσύνολο της Τεχνίτης Νοημοσύνης δύναται να αναπτυχθεί περεταίρω. Είναι αναπόφευκτο το γεγονός ότι μια μέρα θα υπάρξει έξυπνη συμπεριφορά από τις μηχανές δυσδιάκριτη από την ανθρώπινη. Για αυτόν ακριβώς τον λόγο χρειάζεται ιδιαίτερη προσοχή από την παγκόσμια κοινότητα.

ΠΙΝΑΚΑΣ ΟΡΟΛΟΓΙΑΣ

Ξενόγλωσσος όρος	**Ελληνικός Όρος**
Machine Learning	Μηχανική Μάθηση
Regression	Παλινδρόμηση
Neural Networks	Νευρωνικά Δίκτυα
K-Nearest Neighbors	Κ-Πλησιέστερων Γειτόνων
Decision Trees	Δένδρα Αποφάσεων
Clustering	Συσταδοποίηση
Genetic Algorithms	Γενετικοί Αλγόριθμοι
Support Vector Machines	Μηχανές Διανυσματικής Στήριξης
Data Warehouses	Αποθήκες Δεδομένων
Deep Learning	Βαθιά Μάθηση
Artificial Intelligence	Τεχνίτη Νοημοσύνη
Classification	Κατηγοριοποίηση
Linear Regression	Γραμμική Παλινδρόμηση
Logistic Regression	Λογιστική Παλινδρόμηση
Association Rules	Κανόνων Συσχέτισης
Time Series Analysis	Ανάλυση Χρονολογικών Σειρών
Supervised Learning	Επιβλεπόμενη Μάθηση
Unsupervised Learning	Μη Επιβλεπόμενη Μάθηση
Semi-Supervised Learning	Ημι-επιβλεπόμενη Μάθηση
Active Learning	Ενεργή Μηχανική Μάθηση
Training Data	Δεδομένα Εκπαίδευσης
Test Data	Δεδομένα Δοκιμών
Classifier	Ταξινομιτής
Self-training	Αυτό-εκπαίδευση
Co-training	Συνεκπαίδευση
Tri-training	Τρι-εκπαίδευση
Democratic Co-training	Δημοκρατική Συνεκπαίδευση
Tri-training with Editing	Τρι-εκπαίδευση με Επεξεργασία
Instance Based learning	μάθηση Βασισμένη-σε-Περίπτωση
Lazy Learning	Οκνηρή Εκμάθηση
Support Vector Machines	Μηχανές Διανυσμάτων Υποστήριξης
Step Function	Βηματική Συνάρτηση
Sigmoid Function	Σιγμοειδής Συνάρτηση
Logistic Function	Λογιστική Συνάρτηση
Receiver Operating Characteristic	Καμπύλες Διαχείρισης Λειτουργικών Χαρακτηριστικών
Data Mining	Εξόρυξη Δεδομένων
Accuracy	Ακρίβεια
Confusion Matrix	Πίνακας Σύγχυσης
Matching Matrix	Πίνακας Ταιριάσματος
Precision	Ακρίβεια Προσέγγισης
Recall	Ανάκληση
Area Under the Curve	Περιοχή Κάτω από την Καμπύλη
Precision Recall Curves	Καμπύλες Ανάκλησης Ακρίβειας
Dataset	Σύνολο Δεδομένων

ΣΥΝΤΜΗΣΕΙΣ – ΑΡΚΤΙΚΟΛΕΞΑ – ΑΚΡΩΝΥΜΙΑ

RASCO	Random Subspaces for Co-training
Rel-RASCO	Relevant Random Subspaces for Co-training
KNN	K-Nearest Neighbors
SVM	Support Vector Machines
ROC	Receiver Operating Characteristic
FN	False Negative
FP	False Positive
TP	True Positive
TN	True Negative
TPR	True Positive Rate
FPR	False Positive Rate
AUC	Area Under the Curve
PR Curves	Precision Recall Curves
IDE	Integrated Development Environment
ReLU	Rectified Linear Unit

ΠΑΡΑΡΤΗΜΑ - ΚΩΔΙΚΑΣ

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

import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from matplotlib import rcParams
import re
from sklearn.preprocessing import StandardScaler


%matplotlib inline

from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC, LinearSVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.linear_model import Perceptron
from sklearn.linear_model import SGDClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LinearRegression

from keras.models import Sequential
from keras.layers import Dense, Activation, Dropout
import keras
from tensorflow.keras.optimizers import Adam

import graphviz
!pip3 install ann_visualizer
from ann_visualizer.visualize import ann_viz

from sklearn.metrics import r2_score
from sklearn.metrics import roc_curve
from sklearn import metrics


rcParams['figure.figsize'] = 10,8


%matplotlib inline
```

```
df_train = pd.read_csv("train.csv")
df_test = pd.read_csv("test.csv")
```

```
print(df_train.info())
print(df_test.info())
```

```
df_train.describe()
```

```
print(df_train.head())
```

```
df_train['Title'] = df_train.Name.apply(lambda x: re.search(' ([A-
Z][a-z]+)\.', x).group(1))

plt.figure(figsize=(12,5))

sns.countplot(x='Title', data=df_train, palette="hls")
plt.xlabel("Title", fontsize=16)
plt.ylabel("Count", fontsize=16)
plt.title("Title Name Count", fontsize=20)
plt.xticks(rotation=45)
plt.show()
```

```
df_test['Title'] = df_test.Name.apply(lambda x: re.search(' ([A-Z][a-
z]+)\.', x).group(1))
```

```
Title_Dictionary = {
    "Capt": "Officer",
    "Col": "Officer",
    "Major": "Officer",
    "Dr": "Officer",
    "Rev": "Officer",
    "Jonkheer": "Royalty",
    "Don": "Royalty",
    "Sir": "Royalty",
    "the Countess": "Royalty",
    "Dona": "Royalty",
    "Lady": "Royalty",
    "Mme": "Mrs",
    "Ms": "Mrs",
    "Mrs": "Mrs",
    "Mlle": "Miss",
    "Miss": "Miss",
    "Mr": "Mr",
    "Master": "Master"
}

df_train['Title'] = df_train.Title.map(Title_Dictionary)
df_test['Title'] = df_test.Title.map(Title_Dictionary)
```

```
print("Chances to survive based on titles: ")
print(df_train.groupby("Title")["Survived"].mean())

plt.figure(figsize=(12,5))

sns.countplot(x='Title', data=df_train, palette="hls",
              hue="Survived")
plt.xlabel("Titles", fontsize=16)
plt.ylabel("Count", fontsize=16)
plt.title("Title Grouped Count", fontsize=20)
plt.xticks(rotation=45)
plt.show()
```

```
age_high_zero_died = df_train[(df_train["Age"] > 0) &
                              (df_train["Survived"] == 0)]
age_high_zero_surv = df_train[(df_train["Age"] > 0) &
                              (df_train["Survived"] == 1)]

plt.figure(figsize=(10,5))

sns.distplot(age_high_zero_surv["Age"], bins=24, color='g')
sns.distplot(age_high_zero_died["Age"], bins=24, color='r')
plt.title("Distribuition and density by Age",fontsize=20)
plt.xlabel("Age",fontsize=15)
plt.ylabel("Distribuition Died and Survived",fontsize=15)
plt.show()
```

```
age_group = df_train.groupby(["Sex","Pclass","Title"])["Age"]

print(age_group.median())
```

```
df_train.loc[df_train.Age.isnull(), 'Age'] =
df_train.groupby(['Sex','Pclass','Title']).Age.transform('median')
```

```
plt.figure(figsize=(12,5))

sns.distplot(df_train["Age"], bins=24)
plt.title("Distribuition and density by Age")
plt.xlabel("Age")
plt.show()
```

```
plt.figure(figsize=(12,5))

g = sns.FacetGrid(df_train, col='Survived',size=5)
g = g.map(sns.distplot, "Age")
plt.show()
```

```
interval = (0, 5, 12, 18, 25, 35, 60, 120)

cats = ['babies', 'Children', 'Teen', 'Student', 'Young', 'Adult',
'Senior']

df_train["Age_cat"] = pd.cut(df_train.Age, interval, labels=cats)

df_train["Age_cat"].head()
```

```
interval = (0, 5, 12, 18, 25, 35, 60, 120)

cats = ['babies', 'Children', 'Teen', 'Student', 'Young', 'Adult',
'Senior']

df_test["Age_cat"] = pd.cut(df_test.Age, interval, labels=cats)
```

```
print(pd.crosstab(df_train.Age_cat, df_train.Survived))

plt.figure(figsize=(12,10))

plt.subplot(2,1,1)
sns.countplot("Age_cat",data=df_train,hue="Survived", palette="hls")
plt.ylabel("Count", fontsize=18)
plt.xlabel("Age Categories", fontsize=18)
plt.title("Age Distribution ", fontsize=20)

plt.subplot(2,1,2)
sns.swarmplot(x='Age_cat',y="Fare",data=df_train,
              hue="Survived", palette="hls", )
plt.ylabel("Fare Distribution", fontsize=18)
plt.xlabel("Age Categories", fontsize=18)
plt.title("Fare Distribution by Age Categories ", fontsize=20)

plt.subplots_adjust(hspace = 0.5, top = 0.9)

plt.show()
```

```
Age_fare = ['Pclass', 'Age_cat'] #seting the desired

cm = sns.light_palette("green", as_cmap=True)
pd.crosstab(df_train[Age_fare[0]], df_train[Age_fare[1]],
            values=df_train['Fare'], aggfunc=['mean']).style.back-
ground_gradient(cmap = cm)
```

```
plt.figure(figsize=(12,5))

sns.distplot(df_train[df_train.Survived == 0]["Fare"],
             bins=50, color='r')
sns.distplot(df_train[df_train.Survived == 1]["Fare"],
             bins=50, color='g')
plt.title("Fare Distribuition by Survived", fontsize=20)
plt.xlabel("Fare", fontsize=15)
plt.ylabel("Density",fontsize=15)
plt.show()
```

```
df_train.Fare = df_train.Fare.fillna(-0.5)

quant = (-1, 0, 8, 15, 31, 600)

label_quants = ['NoInf', 'quart_1', 'quart_2', 'quart_3', 'quart_4']

df_train["Fare_cat"] = pd.cut(df_train.Fare, quant, labels=la-
bel_quants)

print(pd.crosstab(df_train.Fare_cat, df_train.Survived))

plt.figure(figsize=(12,5))

sns.countplot(x="Fare_cat", hue="Survived", data=df_train, pal-
ette="hls")
plt.title("Count of survived x Fare expending",fontsize=20)
plt.xlabel("Fare Cat",fontsize=15)
plt.ylabel("Count",fontsize=15)

plt.show()
```

```
df_test.Fare = df_test.Fare.fillna(-0.5)

quant = (-1, 0, 8, 15, 31, 1000)
label_quants = ['NoInf', 'quart_1', 'quart_2', 'quart_3', 'quart_4']

df_test["Fare_cat"] = pd.cut(df_test.Fare, quant, labels=label_quants)
```

```
del df_train["Fare"]
del df_train["Ticket"]
del df_train["Age"]
del df_train["Cabin"]
del df_train["Name"]

del df_test["Fare"]
del df_test["Ticket"]
del df_test["Age"]
del df_test["Cabin"]
del df_test["Name"]
```

```
print(pd.crosstab(df_train.Pclass, df_train.Embarked))

plt.figure(figsize=(12,5))

sns.countplot(x="Embarked", data=df_train, hue="Pclass",palette="hls")
plt.title('Embarked x Pclass Count', fontsize=20)
plt.xlabel('Embarked with PClass',fontsize=17)
plt.ylabel('Count', fontsize=17)

plt.show()
```

```
df_train["Embarked"] = df_train["Embarked"].fillna('S')

print(pd.crosstab(df_train.Survived, df_train.Embarked))

plt.figure(figsize=(12,5))

sns.countplot(x="Embarked", data=df_train, hue="Survived",pal-
ette="hls")
plt.title('Class Distribuition by survived or not',fontsize=20)
plt.xlabel('Embarked',fontsize=17)
plt.ylabel('Count', fontsize=17)

plt.show()

print(pd.crosstab(df_train.Survived, df_train.Pclass))

plt.figure(figsize=(12,5))

sns.countplot(x="Pclass", data=df_train, hue="Survived",palette="hls")
plt.xlabel('PClass',fontsize=17)
plt.ylabel('Count', fontsize=17)
plt.title('Class Distribuition by Survived or not', fontsize=20)

plt.show()
```

```
g = sns.factorplot(x="SibSp",y="Survived",data=df_train,
                   kind="bar", height = 5, aspect= 1.6, palette =
"hls")
g.set_ylabels("Survival Probability", fontsize=15)
g.set_xlabels("SibSp", fontsize=15)

plt.show()

g  = sns.factorplot(x="Parch",y="Survived",data=df_train, kind="bar",
size = 6,palette = "hls")

g = g.set_ylabels("Survival Probability")
g.set_xlabels("Parch", fontsize=15)

plt.show()
```

```
df_train["FSize"] = df_train["Parch"] + df_train["SibSp"] + 1

df_test["FSize"] = df_test["Parch"] + df_test["SibSp"] + 1

print(pd.crosstab(df_train.FSize, df_train.Survived))
sns.factorplot(x="FSize",y="Survived", data=df_train,
kind="bar",size=6, aspect=1.6)
plt.show()

del df_train["SibSp"]
del df_train["Parch"]

del df_test["SibSp"]
del df_test["Parch"]
```

```
print(pd.crosstab(df_train.Survived, df_train.Sex))

plt.figure(figsize=(12,5))
sns.countplot(x="Sex", data=df_train, hue="Survived",palette="hls")
plt.title('Sex Distribution by Survived or not', fontsize=20)
plt.xlabel('Sex Distribution',fontsize=17)
plt.ylabel('Count', fontsize=17)

plt.show()
```

```
print("Total of Survived or not: ")
print(df_train.groupby("Survived")["PassengerId"].count())

plt.figure(figsize=(12,5))

sns.countplot(x="Survived", data=df_train,palette="hls")
plt.title('Total Distribuition by survived or not', fontsize=22)
plt.xlabel('Target Distribuition', fontsize=18)
plt.ylabel('Count', fontsize=18)

plt.show()
```

```
df_train.head()
```

```
df_train = pd.get_dummies(df_train, columns=["Sex","Em-
barked","Age_cat","Fare_cat","Title"],\
                          prefix=["Sex","Emb","Age","Fare","Prefix"],
drop_first=True)

df_test = pd.get_dummies(df_test, columns=["Sex","Em-
barked","Age_cat","Fare_cat","Title"],\
                         prefix=["Sex","Emb","Age","Fare","Prefix"],
drop_first=True)

plt.figure(figsize=(15,12))
plt.title('Correlation of Features for Train Set')
sns.heatmap(df_train.astype(float).corr(),vmax=1.0,  annot=True)
plt.show()
```

```
train = df_train.drop(["Survived","PassengerId"],axis=1)
train_ = df_train["Survived"]

test_ = df_test.drop(["PassengerId"],axis=1)

X_train = train.values
y_train = train_.values

X_test = test_.values
X_test = X_test.astype(np.float64, copy=False)

sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.fit_transform(X_test)

df_submission = pd.read_csv("gender_submission.csv")
submission_ = df_submission["Survived"]
Y_test = submission_.values
```

```
# Decision Tree
decision_tree = DecisionTreeClassifier()
decision_tree.fit(X_train, y_train)
Y_pred = decision_tree.predict(X_test)
acc_decision_tree = round(decision_tree.score(X_train, y_train) * 100,
2)
```

```
# Gaussian Naive Bayes
gaussian = GaussianNB()
gaussian.fit(X_train, y_train)
Y_pred = gaussian.predict(X_test)
acc_gaussian = round(gaussian.score(X_train, y_train) * 100, 2)
```

```
# Logistic Regression
logreg = LogisticRegression()
logreg.fit(X_train, y_train)
Y_pred = logreg.predict(X_test)
acc_log = round(logreg.score(X_train, y_train) * 100, 2)
```

```
# KNN
knn = KNeighborsClassifier(n_neighbors = 3)
knn.fit(X_train, y_train)
Y_pred = knn.predict(X_test)
acc_knn = round(knn.score(X_train, y_train) * 100, 2)
```

```
# Support Vector Machines
svc = SVC()
svc.fit(X_train, y_train)
Y_pred = svc.predict(X_test)
acc_svc = round(svc.score(X_train, y_train) * 100, 2)
```

```
# Perceptron
perceptron = Perceptron()
perceptron.fit(X_train, y_train)
Y_pred = perceptron.predict(X_test)
acc_perceptron = round(perceptron.score(X_train, y_train) * 100, 2)
```

```
#Neural Network
#Neural Network Structure
model = Sequential()

model.add(Dense(18,
                activation='relu',
                input_dim=20,
                kernel_initializer='uniform'))

model.add(Dropout(0.2))

model.add(Dense(60,
                kernel_initializer='uniform',
                activation='relu'))

model.add(Dropout(0.7))

model.add(Dense(1,
                kernel_initializer='uniform',
                activation='sigmoid'))

model.summary()

#Neural Network Compile & Fit
model.compile(optimizer = Adam(learning_rate=0.001),
                   loss = 'binary_crossentropy',
                   metrics = ['accuracy'])

pred_NN = model.fit(X_train, y_train, validation_data=(X_test,Y_test),
               batch_size = 32,
               epochs = 200, verbose=1, use_multiprocessing=True)

#Neural Network Score
scores = model.evaluate(X_train, y_train, batch_size=25)
acc_NN = round(scores[1]*100, 2)

ann_viz(model, title="Neural Network")
```

```
fig, ax_arr = plt.subplots(nrows = 4, ncols = 2, figsize = (15,20))

#DT
probs = decision_tree.predict_proba(X_test)
preds_tree = probs[:,1]
fprdtree, tprdtree, thresholddtree = roc_curve(Y_test, preds_tree)
roc_aucdtree = auc(fprdtree, tprdtree)

ax_arr[0,0].plot(fprdtree, tprdtree, 'b', label = 'AUC = %0.2f' %
roc_aucdtree)
ax_arr[0,0].plot([0, 1], [0, 1],'r--')
ax_arr[0,0].set_title('ROC Decision Tree ',fontsize=20)
ax_arr[0,0].set_ylabel('True Positive Rate',fontsize=20)
ax_arr[0,0].set_xlabel('False Positive Rate',fontsize=15)
ax_arr[0,0].legend(loc = 'lower right', prop={'size': 16})

#NB
probs = gaussian.predict_proba(X_test)
preds_NB = probs[:,1]
fprNB, tprNB, thresholdNB = roc_curve(Y_test, preds_NB)
roc_aucNB = auc(fprNB, tprNB)

ax_arr[0,1].plot(fprNB, tprNB, 'b', label = 'AUC = %0.2f' % roc_aucNB)
ax_arr[0,1].plot([0, 1], [0, 1],'r--')
ax_arr[0,1].set_title('ROC Gaussian Naive Bayes ',fontsize=20)
ax_arr[0,1].set_ylabel('True Positive Rate',fontsize=20)
ax_arr[0,1].set_xlabel('False Positive Rate',fontsize=15)
ax_arr[0,1].legend(loc = 'lower right', prop={'size': 16})

#LReg
probs = logreg.predict_proba(X_test)
preds_logreg = probs[:,1]
fprlogreg, tprlogreg, thresholdlogreg = roc_curve(Y_test,
preds_logreg)
roc_auclogreg = auc(fprlogreg, tprlogreg)

ax_arr[1,0].plot(fprlogreg, tprlogreg, 'b', label = 'AUC = %0.2f' %
roc_auclogreg)
ax_arr[1,0].plot([0, 1], [0, 1],'r--')
ax_arr[1,0].set_title('ROC Logistic Regression ',fontsize=20)
ax_arr[1,0].set_ylabel('True Positive Rate',fontsize=20)
ax_arr[1,0].set_xlabel('False Positive Rate',fontsize=15)
ax_arr[1,0].legend(loc = 'lower right', prop={'size': 16})
```

```
#KNN
probs = knn.predict_proba(X_test)
preds_KNN = probs[:,1]
fprkNN, tprkNN, thresholdkNN = roc_curve(Y_test, preds_KNN)
roc_auckNN = auc(fprkNN, tprkNN)

ax_arr[1,1].plot(fprkNN, tprkNN, 'b', label = 'AUC = %0.2f' %
roc_auckNN)
ax_arr[1,1].plot([0, 1], [0, 1],'r--')
ax_arr[1,1].set_title('ROC k-Nearest Neighbors ',fontsize=20)
ax_arr[1,1].set_ylabel('True Positive Rate',fontsize=20)
ax_arr[1,1].set_xlabel('False Positive Rate',fontsize=15)
ax_arr[1,1].legend(loc = 'lower right', prop={'size': 16})

#SVC
probs = svc.predict_proba(X_test)
preds_SVC = probs[:,1]
fprsvc, tprsvc, thresholdsvc = roc_curve(Y_test, preds_SVC)
roc_aucsvc = auc(fprsvc, tprsvc)

ax_arr[2,0].plot(fprsvc, tprsvc, 'b', label = 'AUC = %0.2f' %
roc_aucsvc)
ax_arr[2,0].plot([0, 1], [0, 1],'r--')
ax_arr[2,0].set_title('ROC Support Vector Machine',fontsize=20)
ax_arr[2,0].set_ylabel('True Positive Rate',fontsize=20)
ax_arr[2,0].set_xlabel('False Positive Rate',fontsize=15)
ax_arr[2,0].legend(loc = 'lower right', prop={'size': 16})


#Perceptron
probs = perceptron.decision_function(X_test)
preds_prtn = probs
fprprtn, tprprtn, thresholdprtn = roc_curve(Y_test, preds_prtn)
roc_aucprtn = auc(fprprtn, tprprtn)

ax_arr[2,1].plot(fprprtn, tprprtn, 'b', label = 'AUC = %0.2f' %
roc_aucprtn)
ax_arr[2,1].plot([0, 1], [0, 1],'r--')
ax_arr[2,1].set_title('ROC Perceptron ',fontsize=20)
ax_arr[2,1].set_ylabel('True Positive Rate',fontsize=20)
ax_arr[2,1].set_xlabel('False Positive Rate',fontsize=15)
ax_arr[2,1].legend(loc = 'lower right', prop={'size': 16})
```

```
#NN
probs = model.predict_proba(X_test)
preds_NN = probs
fprNN, tprNN, thresholdNN = roc_curve(Y_test, preds_NN)
roc_aucNN = auc(fprNN, tprNN)

ax_arr[3,0].plot(fprNN, tprNN, 'b', label = 'AUC = %0.2f' % roc_aucNN)
ax_arr[3,0].plot([0, 1], [0, 1],'r--')
ax_arr[3,0].set_title('ROC Neural Network ',fontsize=20)
ax_arr[3,0].set_ylabel('True Positive Rate',fontsize=20)
ax_arr[3,0].set_xlabel('False Positive Rate',fontsize=15)
ax_arr[3,0].legend(loc = 'lower right', prop={'size': 16})

#ALL
ax_arr[3,1].plot(fprdtree, tprdtree, 'b', label = 'Decision Tree',
color='black')
ax_arr[3,1].plot(fprNB, tprNB, 'b', label = 'Gaussian Naive Bayes',
color='blue')
ax_arr[3,1].plot(fprlogreg, tprlogreg, 'b', label = 'Logistic Regres-
sion', color='brown')
ax_arr[3,1].plot(fprkNN, tprkNN, 'b', label = 'k-Nearest Neighbors',
color='green')
ax_arr[3,1].plot(fprsvc, tprsvc, 'b', label = 'Support Vector Ma-
chine', color='grey')
ax_arr[3,1].plot(fprprtn, tprprtn, 'b', label = 'Perceptron',
color='red')
ax_arr[3,1].plot(fprNN, tprNN, 'b', label = 'Neural Network',
color='orange')
ax_arr[3,1].set_title('ROC ',fontsize=20)
ax_arr[3,1].set_ylabel('True Positive Rate',fontsize=20)
ax_arr[3,1].set_xlabel('False Positive Rate',fontsize=15)
ax_arr[3,1].legend(loc = 'lower right', prop={'size': 16})

plt.subplots_adjust(wspace=0.2)
plt.tight_layout()
```

```
fig, ax_arr = plt.subplots(nrows = 4, ncols = 2, figsize = (15,20))

#DT
prdtree, redtree, thsld_tree = precision_recall_curve(Y_test,
preds_tree)

ax_arr[0,0].plot(prdtree, redtree, 'b', label = 'AUC = %0.2f' %
roc_aucdtree)
ax_arr[0,0].set_title('PR Decision ',fontsize=20)
ax_arr[0,0].set_ylabel('Precision',fontsize=20)
ax_arr[0,0].set_xlabel('Recall',fontsize=15)
ax_arr[0,0].legend(loc = 'lower left', prop={'size': 16})

#NB
prNB, reNB, thsld_NB = precision_recall_curve(Y_test, preds_NB)

ax_arr[0,1].plot(prNB, reNB, 'b', label = 'AUC = %0.2f' % roc_aucNB)
ax_arr[0,1].set_title('PR Gaussian Naive Bayes ',fontsize=20)
ax_arr[0,1].set_ylabel('Precision',fontsize=20)
ax_arr[0,1].set_xlabel('Recall',fontsize=15)
ax_arr[0,1].legend(loc = 'lower left', prop={'size': 16})

#LReg
prlogreg, relogreg, thsld_logreg = precision_recall_curve(Y_test,
preds_logreg)

ax_arr[1,0].plot(prlogreg, relogreg, 'b', label = 'AUC = %0.2f' %
roc_auclogreg)
ax_arr[1,0].set_title('PR Logistic Regression ',fontsize=20)
ax_arr[1,0].set_ylabel('Precision',fontsize=20)
ax_arr[1,0].set_xlabel('Recall',fontsize=15)
ax_arr[1,0].legend(loc = 'lower left', prop={'size': 16})

#KNN
prkNN, rekNN, thsld_kNN = precision_recall_curve(Y_test, preds_KNN)

ax_arr[1,1].plot(prkNN, rekNN, 'b', label = 'AUC = %0.2f' %
roc_auckNN)
ax_arr[1,1].set_title('PR k-Nearest Neighbors ',fontsize=20)
ax_arr[1,1].set_ylabel('Precision',fontsize=20)
ax_arr[1,1].set_xlabel('Recall',fontsize=15)
ax_arr[1,1].legend(loc = 'lower left', prop={'size': 16})

#SVC
prsvc, resvc, thsld_SVC = precision_recall_curve(Y_test, preds_SVC)
```

```
ax_arr[2,0].plot(prsvc, resvc, 'b', label = 'AUC = %0.2f' %
roc_aucsvc)
ax_arr[2,0].set_title('PR Support Vector Machine',fontsize=20)
ax_arr[2,0].set_ylabel('Precision',fontsize=20)
ax_arr[2,0].set_xlabel('Recall',fontsize=15)
ax_arr[2,0].legend(loc = 'lower left', prop={'size': 16})


#Perceptron
prprtn, reprtn, thsld_prtn = precision_recall_curve(Y_test,
preds_prtn)

ax_arr[2,1].plot(prprtn, reprtn, 'b', label = 'AUC = %0.2f' %
roc_aucprtn)
ax_arr[2,1].set_title('PR Perceptron ',fontsize=20)
ax_arr[2,1].set_ylabel('Precision',fontsize=20)
ax_arr[2,1].set_xlabel('Recall',fontsize=15)
ax_arr[2,1].legend(loc = 'lower left', prop={'size': 16})


#NN
prNN, reNN, thsld_NN = precision_recall_curve(Y_test, preds_NN)

ax_arr[3,0].plot(prNN, reNN, 'b', label = 'AUC = %0.2f' % roc_aucNN)
ax_arr[3,0].set_title('PR Neural Network ',fontsize=20)
ax_arr[3,0].set_ylabel('Precision',fontsize=20)
ax_arr[3,0].set_xlabel('Recall',fontsize=15)
ax_arr[3,0].legend(loc = 'lower left', prop={'size': 16})

#ALL
ax_arr[3,1].plot(prdtree, redtree, 'b', label = 'Decision Tree',
color='black')
ax_arr[3,1].plot(prNB, reNB, 'b', label = 'Gaussian Naive Bayes',
color='blue')
ax_arr[3,1].plot(prlogreg, relogreg, 'b', label = 'Logistic Regres-
sion', color='brown')
ax_arr[3,1].plot(prkNN, rekNN, 'b', label = 'k-Nearest Neighbors',
color='green')
ax_arr[3,1].plot(prsvc, resvc, 'b', label = 'Support Vector Machine',
color='grey')
ax_arr[3,1].plot(prprtn, reprtn, 'b', label = 'Perceptron',
color='red')
ax_arr[3,1].plot(prNN, reNN, 'b', label = 'Neural Network', color='or-
ange')
ax_arr[3,1].set_title('PR ',fontsize=20)
ax_arr[3,1].set_ylabel('Precision',fontsize=20)
ax_arr[3,1].set_xlabel('Recall',fontsize=15)
ax_arr[3,1].legend(loc = 'lower left', prop={'size': 16})

plt.subplots_adjust(wspace=0.2)
plt.tight_layout()
```

```
plt.plot(pred_NN.history['accuracy'])
plt.plot(pred_NN.history['val_accuracy'])
plt.title('Model Accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
```

```
plt.plot(pred_NN.history['loss'])
plt.plot(pred_NN.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
```

```
models = pd.DataFrame({
    'Model': ['SVM', 'KNN', 'Logistic Regression',
              'Naive Bayes', 'Perceptron', 'Decision Tree', 'Neural
Network'],
    'Score by Accuracy %': [acc_svc, acc_knn, acc_log,
                            acc_gaussian, acc_perceptron,
                            acc_decision_tree, acc_NN]})
sorted_score = models.sort_values(by='Score by Accuracy %', ascend-
ing=False)
print(sorted_score)

plt.figure(figsize=(12, 5))

g = sns.factorplot(x="Model", y="Score by Accuracy %",
data=sorted_score,
                   kind="bar", height=5, aspect=1.8, palette="hls",
hue="Score by Accuracy %", dodge=False)
g.set_ylabels("Score by Accuracy %", fontsize=15)
g.set_xlabels("Model", fontsize=15)

plt.show()
```

ΒΙΒΛΙΟΓΡΑΦΙΚΕΣ ΑΝΑΦΟΡΕΣ

[1] A History of Bayes' Theorem, http://lesswrong.com/lw/774/a_history_of_bayes_theorem/

[2] The Growing Timeline of AI Milestones, https://achievements.ai/

[3] Alpaydin, E. (2009). Introduction to machine learning. MIT press.

[4] Carbonell, J. G., & Gil, Y. (1987). 'Learning by Experimentation. Machine Learning, 256-266.

[5] Mitchell, T. M. (1997). Machine learning. 1997. Burr Ridge, IL: McGraw Hill, 45(37), 870-877.

[6] Witten, I. H., Frank, E., Hall, M. A., & Pal, C. J. (2016). Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann.

[7] Margaret Dunham, Data Mining, Εισαγωγικά και Προηγμένα Θέματα Εξόρυξης Γνώσης Από Δεδομένα, Επιμέλεια Ελληνικής Έκδοσης Βασίλης Βερύκιος & Γιάννης Θεοδωρίδης, 2004.

[8] Fisher, R. A. (1936). "The Use of Multiple Measurements in Taxonomic Problems". Annals of Eugenics. 7 (2): 179–188

[9] Alan O. Sykes, "An Introduction to Regression Analysis" (Coase-Sandor Institute for Law & Economics Working Paper No. 20, 1993).

[10] Everitt, B. S., Landau, S., Leese, M. and Stahl, D. (2011) Index, in Cluster Analysis, 5th Edition, John Wiley & Sons, Ltd, Chichester, UK. doi: 10.1002/9780470977811.index

[11] Agrawal, R.; Imieliński, T.; Swami, A. (1993). "Mining association rules between sets of items in large databases". Proceedings of the 1993 ACM SIGMOD international conference on Management of data - SIGMOD '93. p. 207.

[12] Introduction to Time Series Analysis, http://www.itl.nist.gov/div898/handbook/pmc/section4/pmc4.htm

[13] Zhu, X., & Goldberg, A. B. (2009). Introduction to semi-supervised learning. Synthesis lectures on artificial intelligence and machine learning, 3(1), 1-130.

[14] Tan, P. N., Steinbach, M., & Kumar, V. (2013). Data mining cluster analysis: basic concepts and algorithms. Introduction to data mining.

[15] Kotsiantis, S. B., Zaharakis, I., & Pintelas, P. (2007). Supervised machine learning: A review of classification techniques.

[16] Witten, I. H., Frank, E., Hall, M. A., & Pal, C. J. (2016). Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann.

[17] Yarowsky, D. (1995). Unsupervised word sense disambiguation rivaling supervised methods. In Proceedings of the 33rd annual meeting on Association for Computational Linguistics (pp. 189-196). Association for Computational Linguistics.

[18] Blum, A., & Mitchell, T. (1998). Combining labeled and unlabeled data with cotraining. In Proceedings of the eleventh annual conference on Computational learning theory (pp. 92-100). ACM.

[19] Zhou, Z. H., & Li, M. (2005). Tri-training: Exploiting unlabeled data using three classifiers. IEEE Transactions on knowledge and Data Engineering, 17(11), 1529- 1541.

[20] Breiman, L. (1996). Bagging predictors. Machine learning, 24(2), 123-140.

[21] Hady, M. F. A., & Schwenker, F. (2010). Combining committee-based semi-supervised learning and active learning. Journal of Computer Science and Technology, 25(4), 681-698.

[22] Zhou, Y., & Goldman, S. (2004). Democratic co-learning. In Tools with Artificial Intelligence, 2004. ICTAI 2004. 16th IEEE International Conference on (pp. 594-602). IEEE.

[23] Deng, C., & Guo, M. (2006). Tri-training and data editing based semi-supervised clustering algorithm. MICAI 2006: Advances in Artificial Intelligence, 641-651.

[24] Wang, J., Luo, S. W., & Zeng, X. H. (2008). A random subspace method for cotraining. In Neural Networks, 2008. IJCNN 2008. (IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on (pp. 195-200). IEEE.

[25] Yaslan, Y., & Cataltepe, Z. (2009). Random relevant and non-redundant feature subspaces for co-training. In International Conference on Intelligent Data Engineering and Automated Learning (pp. 679-686). Springer, Berlin, Heidelberg.
[26] Dasgupta, S. (2011). Two faces of active learning. Theoretical computer science, 412(19), 1767-1781.
[27] Roiger, J. R., & Geatz, M. W. (2003). Data Mining: A tutorial-based primer. NY: Pearson Education.
[28] Hanson, R., Stutz, J., & Cheeseman, P. (1991). Bayesian classification theory.
[29] Simple Linear Regression - Oxford Journals, http://www.oxfordjournals.org/our_journals/tropej/online/ma_chap2.pdf
[30] Altman, Naomi S. (1992). "An introduction to kernel and nearest-neighbor nonparametric regression"
[31] N Giannakeas, PS Karvelis, DI Fotiadis, “A classification-based segmentation of cDNA microarray images using support vector machines,”, in Proc of the 30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, pp. 875-878, 2008.
[32] Kevin Gurney, An introduction to neural networks, University of Sheffield, Taylor & Francis e-Library, 2004.
[33] Activation Functions, https://en.wikibooks.org/wiki/Artificial_Neural_Networks/Activation_Functions
[34] McCulloch, W.S. & Pitts, W. Bulletin of Mathematical Biophysics (1943) 5: 115. doi:10.1007/BF02478259
[35] Steel, R. G. D.; Torrie, J. H. (1960). Principles and Procedures of Statistics with Special Reference to the Biological Sciences.
[36] Δρ. Μιχαήλ Ε. Φιλιππάκης. (2016).ΣΤΑΤΙΣΤΙΚΕΣ ΜΕΘΟΔΟΙ & ΑΝΑΛΥΣΗ ΠΑΛΙΝΔΡΟΜΗΣΗΣ για τις ΝΕΕΣ ΤΕΧΝΟΛΟΓΙΕΣ.
[37] Spackman, 1989, Signal detection theory: valuable tools for evaluating inductive learning, Proceedings of the sixth international workshop on Machine learning
[38] FAWCETT T. (2006): An Introduction to ROC Analysis,” Pattern Recognition Letters 27 (8), 861-874.
[39] DOMINGOS P. (1999): Metacost: A General Method for Making Classifiers Cost-sensitive, In Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 155-164, San Diego, CA, ACM Press.
[40] Stehman, 1997, Selecting and interpreting measures of thematic classification accuracy
[41] HAIBO H. and GARCIA E.A. (2009): Learning from Imbalanced Data, IEEE Transactions on Knowledge and Data Engineering 21 (9)
[42] Swets, 1988, Measuring the Accuracy of Diagnostic Systems
[43] Bradley, 1997, The use of the area under the ROC curve in the evaluation of machine learning algorithms
[44] Davis and Goadrich, 2006, The Relationship Between Precision-Recall and ROC Curves, Conference: Proceedings of the 23rd International Conference on Machine Learning
[45] C Drummond, RC Holte - Machine learning, 2006.Cost curves: An improved method for visualizing classifier performance