ΠΑΝΕΠΙΣΤΗΜΙΟ
ΘΕΣΣΑΛΙΑΣ

Πανεπιστήμιο Θεσσαλίας

Πρόγραμμα Μεταπτυχιακών Σπουδών

«Πληροφορική και Υπολογιστική Βιοϊατρική»

Ασφάλεια Υπολογιστικών και Τηλεπικοινωνιακών Συστημάτων, Διαχείριση Μεγάλου Όγκου Δεδομένων και Προσομοίωση

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Συγκριτική Αποτίμηση Αλγορίθμων Μηχανικής Μάθησης και Συνδυασμός τους για την Ανίχνευση Φαινομένων

Τσούκας Ε. Βασίλειος

Επιβλέπων Καθηγητής: κ. Σταμούλης Γεώργιος

Επιστημονικός Σύμβουλος: κ. Κολομβάτσος Κωνσταντίνος

Λαμία 2018

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Συγκριτική Αποτίμηση Αλγορίθμων Μηχανικής Μάθησης και Συνδυασμός τους για την Ανίχνευση Φαινομένων

Τσούκας Ε. Βασίλειος

ΑΕΜ: 00328

Επιβλέπων Καθηγητής: κ. Σταμούλης Γεώργιος

Επιστημονικός Σύμβουλος: κ. Κολομβάτσος Κωνσταντίνος

ΠΕΡΙΛΗΨΗ

Η ραγδαία ανάπτυξη της τεχνολογίας τόσο σε επίπεδο hardware αλλά και software είναι σε θέση πλέον να βοηθήσει σε προβλήματα που παλιότερα η λύση τους φάνταζε μη εφικτή. Η Ελλάδα, χώρα με πανέμορφα πλούσια δάση και ως γνωστόν από τους καλύτερους τουριστικούς προορισμούς, μαστίζεται κάθε χρόνο από τεράστιες καταστροφές που οφείλονται σε πυρκαγιά η οποία προέρχεται από την ασυνειδησία ορισμένων, σπανίως λόγω των καιρικών συνθηκών και κυρίως από εμπρησμούς προς όφελος κάποιων προνομιούχων που επιζητούν περισσότερη γη για εκμετάλλευση.

Η μηχανική μάθηση μία σχετικά νέα τεχνολογία με μεγάλη ανάπτυξη τα τελευταία χρόνια, ήδη έχει κάνει βήματα στο να εξαλείψει μικρού μεγέθους προβλήματα και είμαι αισιόδοξος πως μέσα στα επόμενα χρόνια θα μπορέσει αν όχι να αποτελέσει λύση για μεγαλύτερης εμβέλειας κινδύνους σε όλους τους τομείς, να είναι σε θέση τουλάχιστον να τους μειώσει κατά ένα μεγάλο ποσοστό. Η ερώτηση που προέκυψε και η συγκεκριμένη διπλωματική προσπαθεί να δώσει απάντηση είναι αν μπορεί να δημιουργηθεί μια εφαρμογή η οποία θα μπορεί να προβλέψει πότε υπάρχει κίνδυνος να ξεσπάσει πυρκαγιά σε περιοχές της Ελλάδας έτσι ώστε οι κρατικοί μηχανισμοί να είναι έτοιμοι να την αντιμετωπίσουν άμεσα.

Η διπλωματική εργασία αποσκοπεί στην ανάπτυξη μίας εφαρμογής που θα δέχεται ένα αρχείο το οποίο θα περιέχει στοιχεία όπως η θερμοκρασία, η υγρασία, το επίπεδο του οξυγόνου και του διοξειδίου του άνθρακα και μέσα από ένα σύνολο αλγορίθμων θα πραγματοποιεί classification αναλόγως με την πιθανότητα ξεσπάσει πυρκαγιά σε κλάσεις από το ένα, ως τη μικρότερη πιθανότητα, έως την κλάση δέκα, ως τη μεγαλύτερη.

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον κύριο Κωνσταντίνο Κολομβάτσο, Διδάσκων στο Μεταπτυχιακό Πρόγραμμα σπουδών, που μου έδωσε την ευκαιρία να ασχοληθώ με ένα τόσο ενδιαφέρον θέμα, για την άμεση απόκριση του στα ερωτήματα μου, αλλά και για το χρόνο που αφιέρωσε για τη διεκπεραίωση της διπλωματικής μου εργασίας.

Πίνακας Περιεχομένων

Εικόνες

Πίνακες

Κεφάλαιο 1

Εισαγωγή

Εισαγωγή

Η παραγωγή δεδομένων γίνεται διαρκώς, εικοσιτέσσερις ώρες το εικοσιτετράωρο, επτά ημέρες την εβδομάδα. Η συγγραφή ενός βιβλίου είναι νέα δεδομένα. Ένα μέσο βιβλίο, για παράδειγμα 300 σελίδων αν μετατραπεί σε ψηφιακή μορφή θα έχει μέγεθος περίπου ένα megabyte. Αν υποθετικά μετατρέπαμε τα δέντρα ενός μικρού δάσους σε βιβλία και αυτά με τη σειρά τους σε ψηφιακή μορφή θα μιλούσαμε για δεδομένα μεγέθους περίπου 80 gigabyte. Στην περίπτωση που επιχειρούσαμε να κάνουμε το ίδιο για το δάσος του Αμαζονίου τότε το αποτέλεσμα θα ήταν δεδομένα ενός terabyte. Οι συγκεκριμένες αναλογίες φανερώνουν πόσα πολλά δεδομένα μπορούν να υπάρξουν σαν ψηφιακή μορφή όταν μιλάμε για terabytes. Σύμφωνα με την IDC's Digital Universe Study από την αρχή της δημιουργίας του πλανήτη έχουν παραχθεί 130 Exabytes δεδομένων. Το 2010 ο αριθμός αυξήθηκε στα 1200 exabytes, το 2015 σχεδόν επταπλασιάστηκε στα 7900 exabytes και για το 2020 αναμένεται να εκτοξευτεί και να αγγίξει τα 41000 exabytes.

Συνεπώς υπάρχει ένα δαιδαλώδες πλήθος δεδομένων τα οποία με τη σωστή εκμετάλλευση μπορούν να αξιοποιηθούν σε πολλούς τομείς, όπως για παράδειγμα η αύξηση κέρδους σε εταιρείες, κάτι που κάνει ήδη το Facebook με τις διαφημίσεις προϊόντων σύμφωνα με τα θέλω των χρηστών, η καλύτερη πρόγνωση ασθένειας στην ιατρική μέσω προβλέψεων από τα στοιχεία των εξετάσεων, η δυναμικότητα και οι κινήσεις στον οικονομικό τομέα για την πρόβλεψη οικονομικής κρίσης, οι προβλέψεις για φυσικές και μη καταστροφές, η πρόβλεψη ενός τροχαίου ατυχήματος και η αποφυγή του μέσω ειδικών μηχανισμών στα μέσα μεταφοράς, ακόμα και η πρόβλεψη για οικονομική απάτη σε μια απλή συναλλαγή.

1.1 Αντικείμενο της Διπλωματικής

Το αντικείμενο της διπλωματικής εργασίας είναι η εκμετάλλευση των δεδομένων και της πληροφορίας καθώς και η πλήρης εκμετάλλευση της τεχνολογίας της μηχανικής μάθησης προκειμένου να μπορεί να αξιοποιεί τα σωστά δεδομένα και να κάνει ,ακριβώς αυτό που επαναλαμβάνω πιο πάνω, προβλέψεις. Μία εφαρμογή σαν αυτή που υλοποίησα θα μπορεί να δέχεται δεδομένα από συγκεκριμένες περιοχές, να διακρίνει τη θερμοκρασία, την υγρασία, τα επίπεδα οξυγόνου και διοξειδίου του άνθρακα και να μπορεί να προβλέπει τον κίνδυνο να ξεσπάσει μια πυρκαγιά. Γίνεται χρήση έξι αλγορίθμων μηχανικής μάθησης τους οποίους συγκρίνουμε με την εισαγωγή διάφορων dataset αναμεσά τους και του κύριου dataset για την φωτιά, προκειμένου να βρούμε το

βέλτιστο αλγόριθμο με την υψηλότερη ακρίβεια και φυσικά υψηλότερο precision και recall που είναι σε θέση να πραγματοποιήσει το classification στα δεδομένα. Συνεπώς κύριο μέλημά μου είναι η δημιουργία ενός ensemble scheme το οποίο προκύπτει από τον συνδυασμό των αλγορίθμων αυτών και η αποτίμησή του μέσω συγκριτικών τεστ από ένα σύνολο διαφορετικών dataset.

1.2 Στόχος της Διπλωματικής

Η παρούσα διπλωματική είναι αρχικά η υλοποίηση μίας εφαρμογής η οποία εκμεταλλευόμενη κάποιους αλγόριθμους μηχανικής μάθησης και ένα σύνολο dataset, στοχεύει στη συγκριτική αποτίμηση των αλγορίθμων μέσα από ένα πλήθος πειραμάτων τα οποία θα έχουν σαν αποτέλεσμα την ανάδειξη του ιδανικού αλγορίθμου για το κάθε dataset ξεχωριστά και κυρίως του dataset με τις περιβαλλοντικές μετρήσεις για το classification του βαθμού επικινδυνότητας πυρκαγιάς. Επιπρόσθετα στοχεύουμε σε έναν συνδυασμό τον αλγορίθμων που θα παράξει το ensemble scheme υπεύθυνο για τις προβλέψεις και την λήψη αποφάσεων. Τέλος απαραίτητη είναι η τελική αποτίμηση του ensemble scheme για το dataset της φωτιάς αλλά και επιπλέον πειράματα για τον τρόπο αντίδρασης του σε διαφορετικά είδη δεδομένων και μεταβλητών.

Η εφαρμογή αυτή θα πρέπει να είναι σε θέση να μπορεί να προειδοποιήσει τα πυροσβεστικά σώματα για το ποσοστό κινδύνου να ξεσπάσει μία πυρκαγιά σύμφωνα με περιβαλλοντικούς μεταβλητές όπως η θερμοκρασία σε συνάρτηση με το επίπεδο του οξυγόνου στην ατμόσφαιρα. Σε δεύτερη φάση στόχος είναι να μπορέσει να εξελιχθεί και να ενσωματωθεί σε μία συσκευή η οποία θα μπορούσε να αποτελέσει μέρος του εξοπλισμού του πυροσβεστικού σώματος στις επικίνδυνες τουλάχιστον περιοχές.

1.3 Δομή της Διπλωματικής

Στα κεφάλαια που ακολουθούν γίνεται περιγραφή της μηχανικής μάθησης, ο τρόπος λειτουργίας και των εφαρμογών της, περιγραφή και ανάλυση των αλγορίθμων που χρησιμοποιήθηκαν, τα στάδια του προγράμματος με τον κώδικα και τα αντίστοιχα αποτελέσματα του, οι βιβλιοθήκες που χρησιμοποιήθηκαν, το περιβάλλον εργασίας και

τα εργαλεία για την ανάπτυξη της εφαρμογής, μελέτη και πειράματα και τέλος τα συμπεράσματα και μελλοντικές χρήσεις

2ο Κεφάλαιο: Θεωρητικό Υπόβαθρο

3ο Κεφάλαιο: Αλγόριθμοι Μηχανικής Μάθησης

4ο Κεφάλαιο: Δεδομένα και Ποιοτική Αποτίμηση Αλγορίθμων

5ο Κεφάλαιο: Το Πρόγραμμα

6ο Κεφάλαιο: Συμπεράσματα και Μελλοντικές Επεκτάσεις

Κεφάλαιο 2

Θεωρητικό Υπόβαθρο

Θεωρητικό Υπόβαθρο

2.1 Μηχανική Μάθηση

Η αρχή γίνεται με τον Alan Turing στο ερώτημα του αν οι μηχανές είναι ικανές να σκεφτούν. Ο Arthur Samuel το 1959 όρισε τη Μηχανική μάθηση ως πεδίο μελέτης που δίνει την ικανότητα στους υπολογιστές να μαθαίνουν χωρίς να έχουν προγραμματιστεί ρητά. Αργότερα ο Tom M Mitchel προτείνει έναν πιο μαθηματικό ορισμό ο οποίος αναφέρει πως ένα πρόγραμμα μαθαίνει από εμπειρία Ε ως προς μία κλάση εργασιών Τ και ένα μέτρο επίδοσης Ρ, αν η επίδοση του σε εργασίες της κλάσης Τ, όπως αποτιμάται από το μέτρο Ρ, βελτιώνεται με την εμπειρία Ε. Η μηχανική μάθηση με πιο απλά λόγια είναι μία μελέτη κατά την οποία κατασκευάζεται ένα σύνολο αλγορίθμων το οποίο μπορεί να κάνει προβλέψεις ή και να εξάγει αποφάσεις ως ένα αποτέλεσμα σύμφωνα από τα δεδομένα που δέχεται σαν είσοδο.

Η μηχανική μάθηση πολλές φορές συγχέεται με άλλους δύο επιστημονικούς τομείς των υπολογιστών. Την τεχνητή νοημοσύνη και την εξόρυξη γνώσεων.

Όσον αφορά το πρώτο σκέλος, η μηχανική μάθηση αναπτύχθηκε αναζητώντας τρόπους κατασκευής μηχανών που θα μάθαιναν από τα δεδομένα και θα μπορούσαν να καταστήσουν τα υπολογιστικά συστήματα αυτόνομα στον τομέα της τεχνητής νοημοσύνης. Τα προβλήματα όμως ήταν τόσο πρακτικού αλλά και θεωρητικού τύπου που είχε σαν αποτέλεσμα να δυσκολεύει την στατιστική φύση την μηχανικής μάθησης κάτι που την έκανε στην περίοδο , τέλος της δεκαετίας του 1980 και αρχές του 1990 να αναδιοργανωθεί ως ένα ξεχωριστό επιστημονικό πεδίο.

Για το δεύτερο σκέλος, η μηχανική μάθηση και η εξόρυξη γνώσεων είναι δύο τομείς οι οποίοι χρησιμοποιούν ίδιες μεθόδους, η μεν πρώτη χρησιμοποιεί για παράδειγμα την μη επιτηρούμενη μάθηση η οποία είναι μια μέθοδος που χρησιμοποιείται για την εξόρυξη δεδομένων και από την άλλη πλευρά στην εξόρυξη μπορεί να χρησιμοποιηθούν μέθοδοι μηχανικής μάθησης όπως αλγόριθμοι για την επεξεργασία των δεδομένων.

Οι εργασίες μηχανικής μάθησης χωρίζονται σε δύο βασικές κατηγορίες, supervised learning και unsupervised learning.

2.1.1 Supervised learning

Στη συγκεκριμένη κατηγορία απαιτείται ένας επιστήμονας ο οποίος θα παρέχει στο πρόγραμμα την είσοδο και το αποτέλεσμα προκειμένου ο αλγόριθμος να μπορέσει να μάθει να αντιστοιχεί τα αποτελέσματα με τις εισόδους. Για το λόγο του ότι χρειάζονται δεδομένα εκπαίδευσης με σκοπό να μάθει το πρόγραμμα, ονομάστηκε η συγκεκριμένη τεχνική επιβλεπόμενη μάθηση. Σύμφωνα με τα δεδομένα εισόδου μπορεί να γίνει διαχωρισμός σε 3 επιπλέον κατηγορίες:

1) Semi-supervised learning
2) Reinforcement learning
3) Active learning

Στην πρώτη κατηγορία θα δοθεί ένα training set από το οποίο θα λείπει ένα, περισσότερα ή και πολλές φορές σχεδόν όλα τα επιθυμητά αποτελέσματα εξόδου.

Στη δεύτερη κατηγορία το πρόγραμμα αλληλοεπιδρά με ένα περιβάλλον χωρίς να γνωρίζει αν είναι κοντά στο στόχο που έχει θέσει ο επιστήμονας. Για παράδειγμα να εκπαιδευτεί το πρόγραμμα σε ένα παιγνίδι σκάκι με αντίπαλο έναν κανονικό παίχτη και να προσπαθήσει να μάθει από τις κινήσεις του.

Τέλος στην τελευταία κατηγορία το πρόγραμμα θα κάνει ερωτήματα και θα ζητήσει περισσότερα δεδομένα από την πηγή προκειμένου να βγάλει τα επιθυμητά αποτελέσματα.

2.1.2 Unsupervised learning

Τα δεδομένα που παρέχονται στο πρόγραμμα είναι unlabeled και ο στόχος του αλγορίθμου είναι να μπορέσει να βρει τη δομή των δεδομένων εισόδου χωρίς να υπάρχει κάποιο δείγμα ή έλεγχος για την ακρίβεια.

2.2 Εφαρμογές Μηχανικής Μάθησης

2.2.1 Regression

Είναι μία στατιστική μέθοδος η οποία επιτρέπει την αναζήτηση σχέσης μεταξύ δύο η περισσότερων μεταβλητών. Κατά τη διάρκεια της συγκεκριμένης ανάλυσης ο ερευνητής μπορεί να καταλάβει ποιοι παράγοντες επηρεάζουν το αποτέλεσμα, με ποιον τρόπο το κάνουν και φυσικά πώς ένας παράγοντας μπορεί να επηρεάσει έναν άλλον. Για να επιτευχθεί η παραπάνω υπόθεση χωρίζουμε τους όρους σε δύο κατηγορίες:

- Εξαρτημένη μεταβλητή : η μεταβλητή για την οποία θέλουμε να γίνει η πρόβλεψη

- Ανεξάρτητες μεταβλητές: οι μεταβλητές που ελέγχουμε το αν και πώς μπορούν να επηρεάσουν την εξαρτημένη μεταβλητή μας.

Συνεπώς για να πραγματοποιήσουμε μία ανάλυση αυτού του τύπου θα πρέπει να έχουμε ένα dataset με μεταβλητές κατά τις οποίες θα υπάρχει σχέση μεταξύ τους και η μία από αυτές θα επηρεάζεται άμεσα από τις υπόλοιπες. Για παράδειγμα το πώς μπορεί να επηρεαστεί ο μισθός ενός υπαλλήλου αναλόγως με τα χρόνια προϋπηρεσίας του σε μία εταιρεία.

Εικόνα 2.1: Παράδειγμα Single Linear Regression

Στο παραπάνω διάγραμμα παρατηρούμε πως υπάρχουν δύο μεταβλητές. Στον άξονα x η μεταβλητή Salary η οποία είναι η εξαρτημένη μεταβλητή για την οποία θέλουμε να μάθουμε περισσότερα και να γίνει η πρόβλεψη και στον άξονα y η μεταβλητή Years of Experience η οποία είναι η ανεξάρτητη μεταβλητή. Είναι από τα πιο απλά παραδείγματα και αμέσως μπορούμε να δούμε τη σχέση μεταξύ των δύο, η οποία είναι πως με τα χρόνια προϋπηρεσίας αυξάνεται και ο μισθός.

Τύποι Regression

Αναλόγως με το είδος του προβλήματος, τη φύση των δεδομένων αλλά και την σχέση που έχουν μεταξύ τους η Regression analysis χωρίζεται στις κατηγορίες που αναγράφονται αναφορικά:

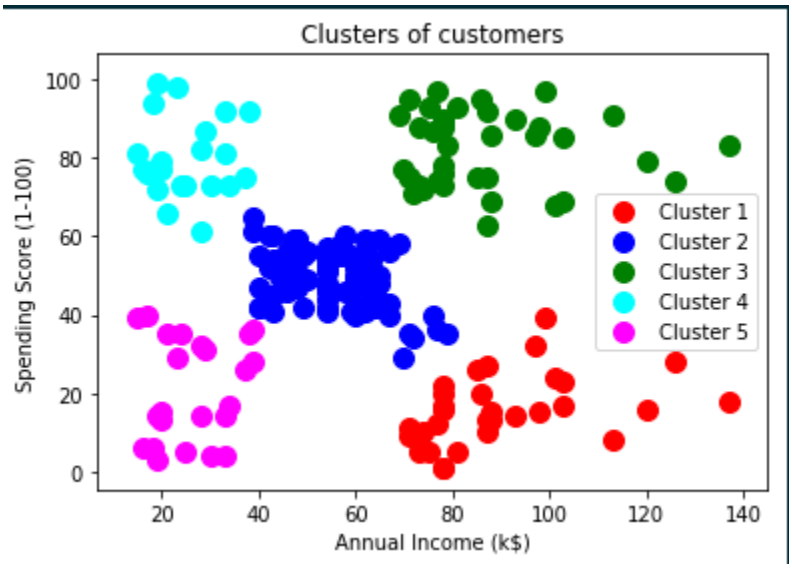
A/A	Κατηγορία
1	Lineal Regression
2	Multiple Linear Regression
3	Polynomial Regression
4	Support Vector Regression
5	Decision Tree Regression
6	Random Forest Regression
7	Non Linear Regression

Πίνακας 2.1: Τύποι Regression

2.2.2 Cluster Analysis

Στην ανάλυση κατά συστάδες ο σκοπός είναι η αξιοποίηση των πληροφοριών στις μεταβλητές των δεδομένων προκειμένου αυτά να ομαδοποιηθούν. Συνεπώς θα προκύψουν ομάδες οι οποίες θα έχουν ομοιογενή δεδομένα. Οι προσεγγίσεις για την ανάλυση είναι οι ιεραρχικές μέθοδοι, οι στατιστικές μέθοδοι και η προσέγγιση K- Means.

Ένα παράδειγμα θα ήταν να χρησιμοποιήσουμε την ανάλυση κατά συστάδες για να πάρουμε πληροφορίες για τους πελάτες ενός mall.



Εικόνα 2.2: Cluster Analysis Visualization

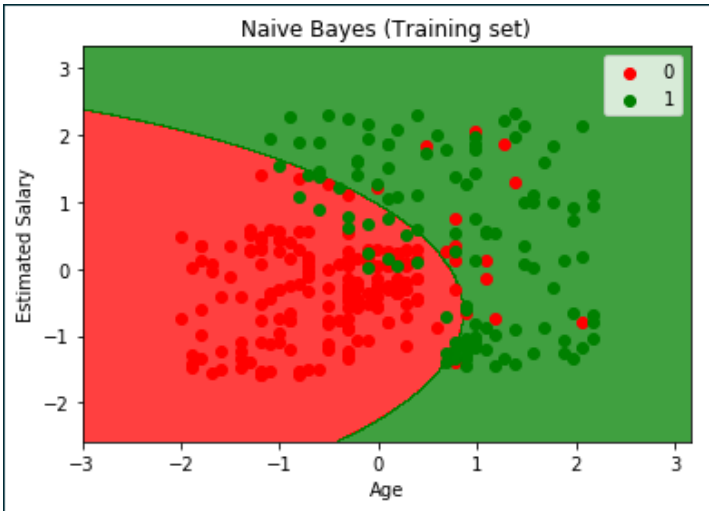
Εδώ ο στόχος μας ήταν να δούμε τη σχέση που έχει ο μισθός με το ποσό που ξοδεύουν οι πελάτες. Η ομαδοποίηση δημιούργησε πέντε διαφορετικά groups αναλόγως με το ετήσιο εισόδημα των πελατών και τους πόντους που αποκομίζουν από το πολυκατάστημα κατά τη διάρκεια της χρονιάς σε συνάρτηση με το ποσό που διαθέτουν για τις αγορές τους.

2.2.3 Classification

Είναι ο διαχωρισμός των δεδομένων εισόδου σε δύο η περισσότερες κλάσεις. Θεωρείται μία από τις βασικές μεθόδους επιβλεπόμενης μάθησης και μπορεί να δώσει απαντήσεις στα παρακάτω ερωτήματα:

1) Είναι spam ή όχι το συγκεκριμένο email;
2) Αυτή η συναλλαγή είναι έμπιστη ή μπορεί να αποτελέσει οικονομική απάτη;
3) Ποια είναι τα φρούτα που αγοράζονται περισσότερο σε ένα μανάβικο;
4) Πόσα αυτοκίνητα θα πωληθούν την επόμενη χρονιά;
5) Μπορεί ένας παίχτης να έχει μεγάλη εξέλιξη στον χ αθλητικό σύλλογο;

Στο παρακάτω διάγραμμα φαίνονται οι μεταβλητές του μισθού και της ηλικίας κατά τις οποίες γίνεται και η ομαδοποίηση σε δύο κλάσεις, την κλάση 1 και την κλάση 2. Αυτό που προκύπτει είναι ότι τα άτομα μεγαλύτερης ηλικίας αμείβονται με περισσότερα χρήματα εν αντιθέσει με τα νεαρότερα που έχουν και μικρότερο μέσο όρο μισθού.



Εικόνα 2.3: Naïve Bayes Visualization

Κεφάλαιο 3

Αλγόριθμοι Μηχανικής Μάθησης

3 Αλγόριθμοι μηχανικής μάθησης

Στο πρόγραμμα που υλοποιήθηκε στα πλαίσια της διπλωματικής εργασίας, έγινε χρήση συνολικά έξι αλγορίθμων. Οι αλγόριθμοι αυτοί είναι οι εξής:

1) Logistic Regression
2) Linear Discriminant Analysis
3) K Nearest Neighbors Classifier
4) Decision Tree Classifier
5) Naive Bayes classifier
6) Support vector machine

Ακολουθεί μία σύντομη περιγραφή τους, απλά παραδείγματα για τον τρόπο που λειτουργούν και τις προβλέψεις τους, τη μαθηματική τους αναπαράσταση, τμήμα κώδικα τους που έχω υλοποιήσει στην γλώσσα Python καθώς και αναπαράσταση των αποτελεσμάτων τους.

3.1 Logistic Regression

Ο συγκεκριμένος αλγόριθμος χρησιμοποιείται για τον υπολογισμό διακριτών τιμών οι οποίες βασίζονται σε ένα σετ ανεξάρτητων μεταβλητών που λαμβάνει από τον χρήστη. Τα αποτελέσματα συνήθως είναι Ναι ή Όχι, Αληθές ή Ψευδές και 0 ή 1. Ενώ το όνομα του περιλαμβάνει τη λέξη Regression δεν είναι ένας αλγόριθμος για Regression αλλά για Classification.

Ένα σύντομο παράδειγμα μπορεί να είναι ένα τεστ γνώσεων που δίνεται σε παιδιά δημοτικού και μέσα από τις απαντήσεις τους θέλουμε να γνωρίζουμε αν θα έχουν ποσοστό σωστών απαντήσεων πάνω από τη βάση. Οπότε το τελικό αποτέλεσμα μπορεί να έχει μόνο δύο εκδοχές, ΝΑΙ ή ΟΧΙ. Οπότε μετά από ένα σύνολο τεστ διαφορετικών θεμάτων θα μπορεί να βγει ένα ποσοστό βασισμένο στις απαντήσεις το οποίο θα δείχνει το πόσο καλός είναι ο μαθητής σε συγκεκριμένα μαθήματα. Για παράδειγμα Ο Βασίλης έχει 80% πιθανότητες να περάσει ένα τεστ το οποίο αφορά την

πληροφορική, 60% για τεστ φυσικής, 50% για τεστ μαθηματικών και 20% για το θεωρητικό τεστ Γλώσσας. Τα αποτελέσματα αυτά μπορούν να βοηθήσουν το μαθητή να καταλάβει την κλίση του και την κατεύθυνση που θα πρέπει να επιλέξει στο μέλλον.

Πλεονεκτήματα:

1) Πρόκειται για έναν εύκολο στην υλοποίηση αλγόριθμο ο οποίος μπορεί να χρησιμοποιηθεί σε Big Data

2) Εμφανίζει την πιθανότητα για τα αποτελέσματα, κάτι που τον καθιστά χρήσιμο σε προβλήματα στατιστικής και πιθανοτήτων

3) Αποτελεσματικό για λίγες μεταβλητές και πιθανότατα η πρώτη επιλογή ενός ερευνητή ο οποίος επιθυμεί να δουλέψει με ένα μικρό δείγμα μεταβλητών

Μειονεκτήματα:

1) Δεν χρησιμοποιείται όταν υπάρχουν πέντε οι περισσότερες κατηγορίες μεταβλητών, τα αποτελέσματα και οι προβλέψεις δεν θα είναι σωστά και αποτελεί πρόβλημα στο οποίο δίνει λύση ο αλγόριθμος Decision Tree.

<u>Μαθηματική αναπαράσταση:</u>

$$\hat{p} = \frac{\exp\left(B_0 + B_1 X\right)}{1 + \exp\left(B_0 + B_1 x\right)} = \frac{e^{B_0 + B_1 x}}{1 + e^{B_0 + B_1 x}}$$

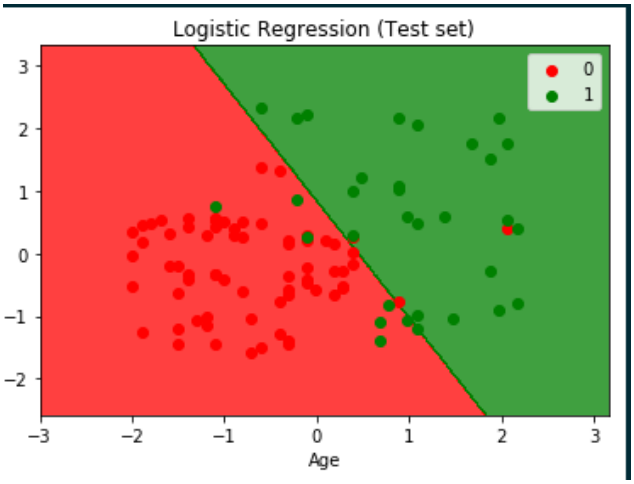
(1)

Υλοποίηση σε Python:

```
# Fitting Logistic Regression to the Training set
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression(random_state = 0)
classifier.fit(X_train, y_train)

# Predicting the Test set results
y_pred = classifier.predict(X_test)

# Making the Confusion Matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)

# Visualising the Training set results
from matplotlib.colors import ListedColormap
X_set, y_set = X_train, y_train
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].max() + 1, step = 0.01),
                     np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:, 1].max() + 1, step = 0.01))
plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
             alpha = 0.75, cmap = ListedColormap(('red', 'green')))
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                c = ListedColormap(('red', 'green'))(i), label = j)
plt.title('Logistic Regression (Training set)')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```

Εικόνα 3.1: Τμήμα κώδικα γλώσσας Python για Logistic Regression

Αναπαράσταση Αποτελεσμάτων:



Εικόνα 3.2: Logistic Regression Visualization

3.2 Linear Discriminant Analysis

Είναι ο αλγόριθμος ο οποίος προσπαθεί να βρει τις ομοιότητες ή τις διαφορές ανάμεσα σε δύο η περισσότερες κλάσεις αντικειμένων. Η Linear Discriminant Analysis έχει αρκετές ομοιότητες με τον αλγόριθμο Logistic Regression και χρησιμοποιείται στα ίδια ερωτήματα. Οι βασικές τους διαφορές είναι μόλις δύο και αυτές περιορίζονται στο μικρό αριθμό από samples που είναι το ιδανικό για καλύτερα αποτελέσματα στην Discriminant Analysis αλλά και στις υποθέσεις και τους περιορισμούς που πρέπει να ορίσουμε στον συγκεκριμένο αλγόριθμο. Τα αποτελέσματα που θα πάρουμε από τα Training Sets θα είναι σαφώς καλύτερα στον αλγόριθμο της ανάλυσης συγκριτικά με τον αλγόριθμο Logistic Regression, αλλά οι περιορισμοί που προανέφερα τον καθιστούν αρκετές φορές πιο δύσχρηστο και μη υλοποιήσιμο για συγκεκριμένα δεδομένα ωθώντας τους Data Scientists στη χρήση του Logistic Regression.

Σαν παράδειγμα μπορούμε να δώσουμε την χρήση του στην Ιατρική όπου μπορεί να υπολογιστεί η σοβαρότητα στην κατάσταση υγείας ενός ασθενούς. Οι ασθενείς χωρίζονται σε ομάδες ανάλογα με την αρχική ανάλυση της ασθένειας, ήπια, μέτρια, σοβαρή, θανάσιμη, γίνεται η σύγκριση με τα αποτελέσματα των εξετάσεων και των εργαστηρίων και με τη χρήση του αλγορίθμου προσπαθούμε να κάνουμε την ίδια κατηγοριοποίηση σε μελλοντικούς ασθενείς.

Μαθηματική Αναπαράσταση:

$$P(y = k|X) = \frac{P(X|y = k)P(y = k)}{P(X)} = \frac{P(X|y = k)P(y = k)}{\sum_l P(X|y = l) \cdot P(y = l)} \quad (2)$$

Πλεονεκτήματα:

1) Μπορεί να διαχειριστεί πολλές κατηγορίες μεταβλητών και χρησιμοποιείται σε προβλήματα όπου αλγόριθμοι όπως ο Logistic Regression είναι αναποτελεσματικοί, όπως για παράδειγμα σε εφαρμογές Face Recognition το οποίο στην ουσία αποτελεί ένα template από ένα μεγάλο πλήθος pixels.

Μειονεκτήματα:

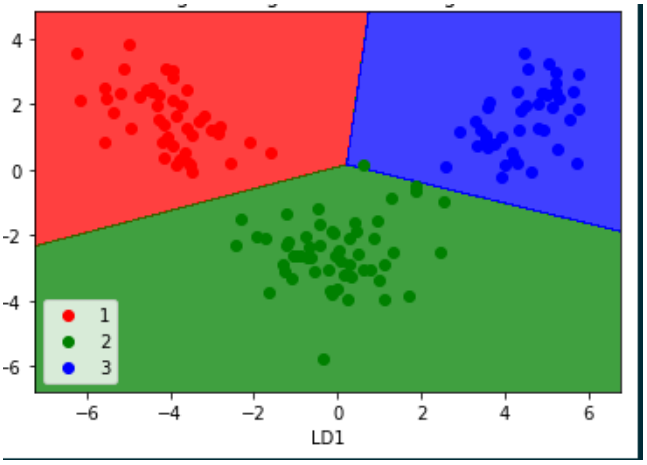
1) Αρκετά παλιότερη μέθοδος η οποία απαιτεί να ικανοποιούνται αρκετά κριτήρια προκειμένου να προχωρήσει στο απαραίτητο classification, ένα από αυτά είναι το πλήθος των μεταβλητών μέσα σε ένα group θα πρέπει να είναι ακριβώς το ίδιο με το άλλο group στο οποίο θα γίνει η σύγκριση. Είναι αρκετά δύσκολο να ικανοποιηθούν όλες οι απαιτήσεις του αλγορίθμου LDA και για αυτό προτιμάται τις περισσότερες φορές ο αλγόριθμος Logistic Regression.

Υλοποίηση σε Python:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

# Importing the dataset
dataset = pd.read_csv('Wine.csv')
X = dataset.iloc[:, 0:13].values
y = dataset.iloc[:, 13].values

# Splitting the dataset into the Training set and Test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 0)

# Feature Scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

# Applying LDA
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA
lda = LDA(n_components = 2)
X_train = lda.fit_transform(X_train, y_train)
X_test = lda.transform(X_test)

# Fitting Logistic Regression to the Training set
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression(random_state = 0)
classifier.fit(X_train, y_train)
```

Εικόνα 3.3: Τμήμα κώδικα γλώσσας Python για Linear Discriminant Analysis

Αναπαράσταση Αποτελεσμάτων:



Εικόνα 3.4: Linear Discriminant Analysis Visualization

3.3 K Nearest Neighbors Classifier

Πρόκειται για ένα σχετικά απλό αλγόριθμο ο οποίος ελέγχει όλες τις περιπτώσεις και ταξινομεί τις καινούριες σύμφωνα με τα γειτονικά στοιχεία. Για να γίνει η ταξινόμηση αυτή χρησιμοποιείται μία συνάρτηση απόστασης. Για την περίπτωση που έχουμε κατηγορικές μεταβλητές χρησιμοποιείται η συνάρτηση Hamming ενώ για όλες τις άλλες περιπτώσεις οι συναρτήσεις που χρησιμοποιούνται είναι η Euclidean, Chebyshev, Minkowski και Manhattan. Ο KNN αλγόριθμος ικανοποιεί προβλήματα classification αλλά και προβλήματα regression ωστόσο είναι αρκετά πιο σύνηθες να τον βλέπουμε σε περιπτώσεις που απαιτείται classification.

Απλά παραδείγματα στην πραγματική ζωή για την λειτουργία του αλγορίθμου είναι όταν κάποιος αιτείται μια θέση εργασίας σε μια εταιρεία και η εταιρεία αυτή έρχεται σε επαφή με τους προηγούμενους εργοδότες και συναδέλφους που ήταν κοντά του για να μάθει περισσότερα για το εργασιακά του καθήκοντα και τις αποδόσεις του, ή πολύ πιο απλό παράδειγμα μπορεί να θεωρηθεί η Ελληνίδα μητέρα ή ακόμα καλύτερα γιαγιά η οποία μόλις μάθει πως το εγγόνι της έχει μία καινούρια σχέση αμέσως θα ξεκινήσει ερωτήσεις που αφορούν τους γονείς, τις δουλείες τους, τα αδέρφια και γενικότερα όλους τους κοντινούς ανθρώπους προκειμένου να μάθει περισσότερα για το συγκεκριμένο άτομο. Θεωρώ πως το συγκεκριμένο παράδειγμα καλύπτει 100% το πώς λειτουργεί ο KNN.

Πλεονεκτήματα:

1) Αποτελεί ίσως τον πιο απλό στην κατανόηση αλλά και στην υλοποίηση αλγόριθμο μηχανικής μάθησης

2) Συγκριτικά με τους υπόλοιπους αλγορίθμους χρειάζεται ελάχιστο χρόνο για την εκπαίδευση του

3) Είναι ιδανικός για datasets με πολλά attributes και μεταβλητές στα οποία θα πρέπει να δημιουργηθούν περισσότερες από δύο κλάσεις

Μειονεκτήματα:

1) Το πρώτο βασικό πρόβλημα του αλγορίθμου έχει να κάνει με την πλευρά του hardware του μηχανήματος που θα τρέξει γιατί αποτελεί έναν ιδιαίτερα υπολογιστικά δαπανηρό αλγόριθμο

2) Το δεύτερο πρόβλημα που τον χαρακτηρίζει είναι η ασυνέπεια σε μεγάλου μεγέθους δεδομένα κατά τα οποία μπορούν να εμφανιστούν προβλήματα με την ακρίβεια

Μαθηματική Αναπαράσταση:

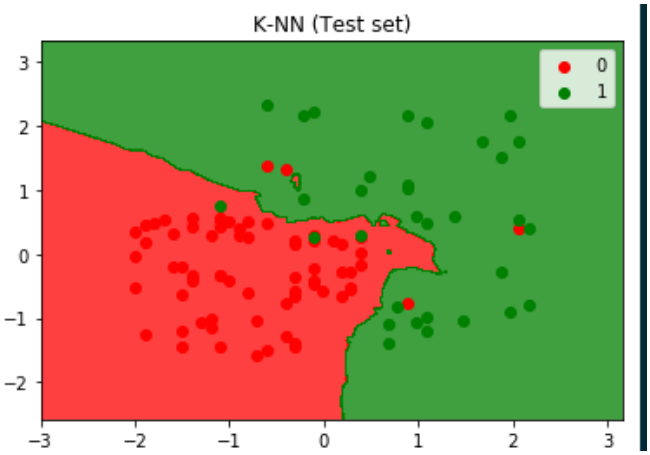
$$P(y = j|X = x) = \frac{1}{K} \sum_{i \in \mathcal{A}} I(y^{(i)} = j) \quad (3)$$

Υλοποίηση σε Python:

```
X = dataset.iloc[:, [2, 3]].values
y = dataset.iloc[:, 4].values

# Splitting the dataset into the Training set and Test set
from sklearn.cross_validation import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state = 0)

# Feature Scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

# Fitting K-NN to the Training set
from sklearn.neighbors import KNeighborsClassifier
classifier = KNeighborsClassifier(n_neighbors = 5, metric = 'minkowski', p = 2)
classifier.fit(X_train, y_train)

# Predicting the Test set results
y_pred = classifier.predict(X_test)

# Making the Confusion Matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
```

Εικόνα 3.5: Τμήμα κώδικα γλώσσας Python για K Nearest Neighbors

Αναπαράσταση Αποτελεσμάτων:

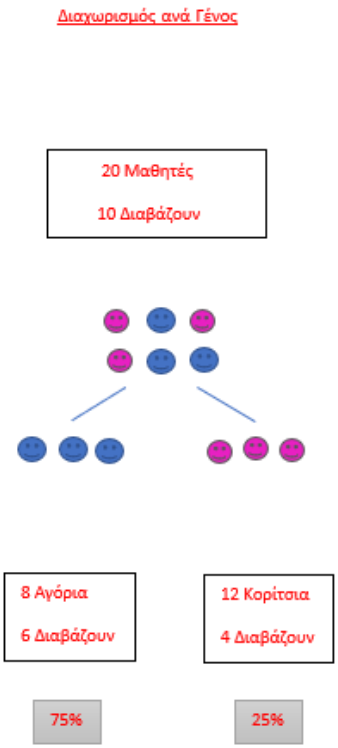


Εικόνα 3.6: K Nearest Neighbors Visualization

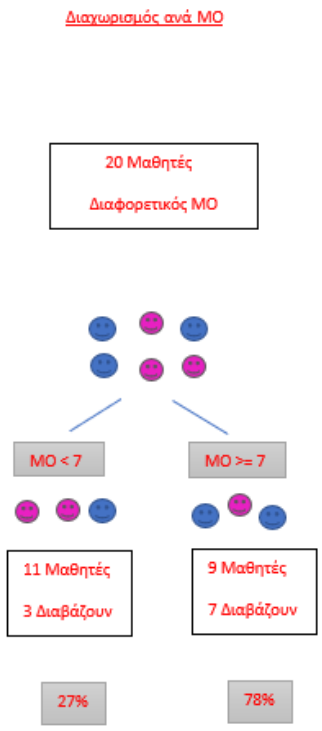
3.4 Decision Tree Classifier

Ο αλγόριθμος χρησιμοποιείται για classification προβλήματα και η βασική του λειτουργία είναι να χωρίζει το πλήθος των δεδομένων σε δύο ή περισσότερα ομογενή σετ. Χωρίζονται σε δύο τύπους Categorical Variable Decision Tree όπου έχουμε categorical variables και συνήθως παίρνουμε απαντήσεις όπως ΝΑΙ ή ΟΧΙ και σε Continuous Variable Decision Tree όπου δουλεύουμε με συνεχείς μεταβλητές.

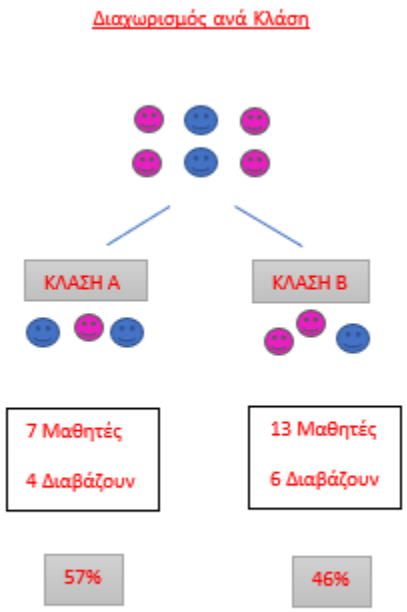
Σαν παράδειγμα μπορούμε να δούμε μία τάξη 20 μαθητών και σαν μεταβλητές το γένος, το Μέσο όρο βαθμολογίας τους και μία κλάση για το διαχωρισμό. Σαν μια εξωσχολική δραστηριότητα – εργασία για το διάστημα των διακοπών του καλοκαιριού τους ανατέθηκε να διαβάσουν ένα βιβλίο. Σύμφωνα με μία έρευνα που έκανε ο καθηγητής, στο 50% των μαθητών δηλαδή στους 10 από τους 20 αρέσει το διάβασμα, διαβάζουν στον ελεύθερο τους χρόνο και έχουν τελειώσει τουλάχιστον ένα βιβλίο μέσα στους τρεις τελευταίους μήνες . Το μοντέλο που επιθυμούμε να κατασκευαστεί θέλουμε να προβλέπει πόσοι από αυτούς τους μαθητές θα ασχοληθούν με το διάβασμα ενός βιβλίου και πόσοι όχι. Αρχικά χωρίζουμε τους μαθητές ανά γένος και με το αν διαβάζουν βιβλία. Ακολουθεί ο ΜΟ βαθμολογίας και η σχέση με το διάβασμα και τέλος δύο κλάσεις που θα έχουν μαθητές ανεξαρτήτων βαθμολογιών και γένους που διαβάζουν βιβλία.



Εικόνα 3.7: Decision Tree για διαχωρισμό ανά γένος



Εικόνα 3.8: Decision Tree για διαχωρισμό ανά ΜΟ



Εικόνα 3.09: Decision Tree για διαχωρισμό ανά κλάση

Πλεονεκτήματα:

1) Ο πιο εύκολος στην επεξήγηση αλγόριθμος και από τους πιο γρήγορος στην παραγωγή αποτελεσμάτων σε μεγάλο πλήθος δεδομένων

2) Είναι αλγόριθμος που μπορεί να σχεδιαστεί εύκολα στο χαρτί κάτι που θα βοηθήσει αργότερα το χρήστη στην υλοποίηση του αλλά και στην προετοιμασία των δεδομένων

3) Καλό για λίγες κατηγορίες μεταβλητών σε προβλήματα classification, με γρήγορα αποτελέσματα και χωρίς το φόβο των διαφόρων παραμέτρων όπως για παράδειγμα ισχύει στον αλγόριθμο Support Vector Machine.

Μειονεκτήματα:

1) Πολλές φορές ο ερευνητής μπορεί να δημιουργήσει αρκετά περίπλοκα δέντρα στα οποία δεν πραγματοποιείται ο σωστός διαχωρισμός των δεδομένων, ένα φαινόμενο το οποίο είναι γνωστό με τον όρο overfitting

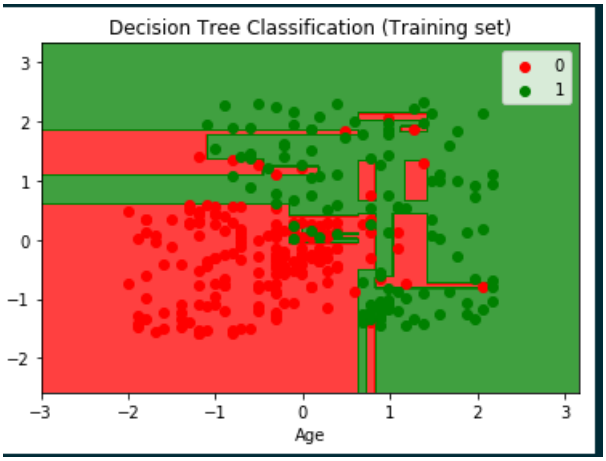
2) Στην περίπτωση που υπάρχει μικρή ποικιλομορφία στα δεδομένα είναι δυνατή η δημιουργία ενός δέντρου μη χρήσιμου το οποίο θα έχει σαν αντίκτυπο να επηρεάσει αρνητικά το αποτέλεσμα

Υλοποίηση σε Python:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

# Importing the dataset
dataset = pd.read_csv(test.csv')
X = dataset.iloc[:, [2, 3]].values
y = dataset.iloc[:, 4].values

# Splitting the dataset into the Training set and Test set
from sklearn.cross_validation import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state = 0)

# Feature Scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

# Fitting Decision Tree Classification to the Training set
from sklearn.tree import DecisionTreeClassifier
classifier = DecisionTreeClassifier(criterion = 'entropy', random_state = 0)
classifier.fit(X_train, y_train)

# Predicting the Test set results
y_pred = classifier.predict(X_test)
```

Εικόνα 3.10: Τμήμα κώδικα γλώσσας Python για Decision Tree

Αναπαράσταση Αποτελεσμάτων:



Εικόνα 3.11: Decision Tree Visualization

3.5 Naive Bayes

Είναι ένας αλγόριθμος για classification ο οποίος βασίζεται σύμφωνα με το θεώρημα του Bayes στη μοναδικότητα των δεδομένων. Δηλαδή στο ότι κάποιες ιδιότητες και χαρακτηριστικά είναι μοναδικά σε συγκεκριμένα αντικείμενα και μέσα από αυτά μπορούμε να καθορίσουμε το είδος και να τα διαχωρίσουμε όπως για παράδειγμα στα φρούτα, ένα φρούτο μπορεί να θεωρηθεί ότι είναι μία μπανάνα αν έχει κίτρινο χρώμα, είναι μακρόστενο και έχει μήκος 15 εκατοστά.

Σαν παράδειγμα μπορούμε να υποθέσουμε πως σχεδιάζουμε να πάμε μία εκδρομή. Ρωτάμε όλα τα άτομα που θα συμμετάσχουν ποια μέρα μπορούν ανάμεσα σε Παρασκευή, Σάββατο, Κυριακή και Δευτέρα για τις επόμενες τέσσερις εβδομάδες. Ετοιμάζουμε το dataset και μετά το χωρίζουμε σε πίνακα συχνότητας και πιθανότητας προκειμένου μέσω του Naïve Bayes να βρούμε τα αποτελέσματα για κάθε κλάση. Η κλάση η οποία θα έχει το υψηλότερο ποσοστό θα είναι και η πρόβλεψη για το αποτέλεσμά μας.

Ημέρα	Εκδρομή
Παρασκευή	ΝΑΙ
Σάββατο	ΝΑΙ
Κυριακή	ΟΧΙ
Δευτέρα	ΝΑΙ
Παρασκευή	ΝΑΙ
Σάββατο	ΟΧΙ
Κυριακή	ΝΑΙ
Δευτέρα	ΝΑΙ
Παρασκευή	ΝΑΙ
Σάββατο	ΟΧΙ
Κυριακή	ΟΧΙ
Παρασκευή	ΟΧΙ
Σάββατο	ΟΧΙ
Κυριακή	ΝΑΙ

Πίνακας 3.1: Πίνακας με τις διαθέσιμες ημέρες για εκδρομή και την αντίστοιχη απόφαση πλειοψηφίας

Πίνακας Συχνότητας		
Ημέρα	ΝΑΙ	ΟΧΙ
Παρασκευή	3	1
Σάββατο	1	3
Κυριακή	2	2
Δευτέρα	2	0

Πίνακας 3.2: Πίνακας Συχνότητας Ημέρας/Απόφασης

Πίνακας Πιθανότητας			
Ημέρα	ΝΑΙ	ΟΧΙ	
Παρασκευή	3	1	0.29
Σάββατο	1	3	0.29
Κυριακή	2	2	0.29
Δευτέρα	2	0	0.15
Αποτέλεσμα	0.57	0.43	

Πίνακας 3.3: Πίνακας Πιθανότητας

Για την πιθανότητα της Παρασκευής θα ισχύει:

P(ΝΑΙ | Παρασκευή) = P (Παρασκευή | ΝΑΙ) * P(ΝΑΙ) / P(Παρασκευή)

P(Παρασκευή | ΝΑΙ) = 0.37

P(Παρασκευή) = 0.29

P(ΝΑΙ) = 0.57

Άρα: P(ΝΑΙ | Παρασκευή) = 0.37 * 0.57 / 0.29

P(ΝΑΙ | Παρασκευή) = 0.72

Για την πιθανότητα του Σαββάτου θα ισχύει:

P(ΝΑΙ | Σάββατο) = P (Σάββατο | ΝΑΙ) * P(ΝΑΙ) / P(Σάββατο)

P(Σάββατο | ΝΑΙ) = 0.12

P(Σάββατο) = 0.29

P(ΝΑΙ) = 0.57

Άρα: P(ΝΑΙ | Σάββατο) = 0.12 * 0.57 / 0.29

P(ΝΑΙ | Σάββατο) = 0.23

Για την πιθανότητα της Κυριακής θα ισχύει:

P(ΝΑΙ | Κυριακή) = P (Κυριακή | ΝΑΙ) * P(ΝΑΙ) / P(Κυριακή)

P(Κυριακή | ΝΑΙ) = 0.25

P(Κυριακή) = 0.29

P(ΝΑΙ) = 0.57

Άρα: P(ΝΑΙ | Κυριακή) = 0.25 * 0.57 / 0.29

P(ΝΑΙ | Κυριακή = 0.49

Για την πιθανότητα της Δευτέρας θα ισχύει:

P(ΝΑΙ | Δευτέρας) = P (Δευτέρας | ΝΑΙ) * P(ΝΑΙ) / P(Δευτέρας)

P(Δευτέρας | ΝΑΙ) = 0.25

P(Δευτέρας) = 0.14

P(ΝΑΙ) = 0.57

Άρα: P(ΝΑΙ | Δευτέρας) = 0.25 * 0.57 / 0.15

P(ΝΑΙ | Δευτέρας) = 0.95

Τα αποτελέσματα που παίρνουμε είναι τα ακόλουθα:

Παρασκευή	**0.65**
Σάββατο	0.23
Κυριακή	0.49
Δευτέρα	0.95

Πίνακας 3.4: Πίνακας αποτελεσμάτων

Οπότε το πιθανότερο να γίνει η συγκεκριμένη εκδρομή είναι την ημέρα της Δευτέρας.

Μαθηματική Αναπαράσταση:

$$p(C_k|\vec{x}) = \frac{p(\vec{x}|C_k)p(C_k)}{p(\vec{x})} = \frac{p(x_1, \ldots, x_n|C_k)p(C_k)}{p(x_1, \ldots, x_n)} \quad (4)$$

Πλεονεκτήματα:

1) Πολύ απλός στην κατανόηση αλγόριθμος ο οποίος βασίζεται στο θεώρημα του Bayes για τη μοναδικότητα των ιδιοτήτων που χαρακτηρίζουν το κάθε αντικείμενο

2) Αποτελεσματικό για λίγες κατηγορίες μεταβλητών κατά τις οποίες το classification πραγματοποιείται πολύ γρήγορα και αυτό αποτελεί ένα από τα κύρια χαρακτηριστικά του μετά την απλότητα του φυσικά την ταχύτητα δηλαδή στους υπολογισμούς

3) Ένα ακόμα από τα πλεονεκτήματα του αλγορίθμου είναι ότι θεωρείται από τους αλγορίθμους μπαλαντέρ καθώς μπορεί να χρησιμοποιηθεί σε πλήθος προβλημάτων τα οποία μπορεί να είναι αρκετά εύκολα ή δύσκολα, ακόμα και σε Big Data χωρίς να είναι ιδιαιτέρως υπολογιστικά δαπανηρός

Μειονεκτήματα:

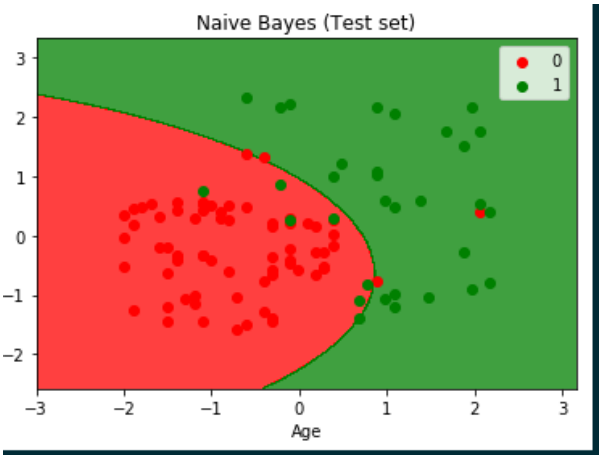
1) Ένα από τα βασικά προβλήματα που αντιμετωπίζει ο αλγόριθμος είναι ότι πολλές φορές αποτυγχάνει στην πρόβλεψη της πιθανότητας για μία κλάση πράγμα που τον καθιστά μη ικανό σε εφαρμογές κυρίως πρόβλεψης και υπολογισμού ποσοστών

Υλοποίηση σε Python:

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

# Fitting Naive Bayes to the Training set
from sklearn.naive_bayes import GaussianNB
classifier = GaussianNB()
classifier.fit(X_train, y_train)

# Predicting the Test set results
y_pred = classifier.predict(X_test)

# Making the Confusion Matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
```

Εικόνα 3.12: Τμήμα κώδικα γλώσσας Python για Naive Bayes

Αναπαράσταση Αποτελεσμάτων:

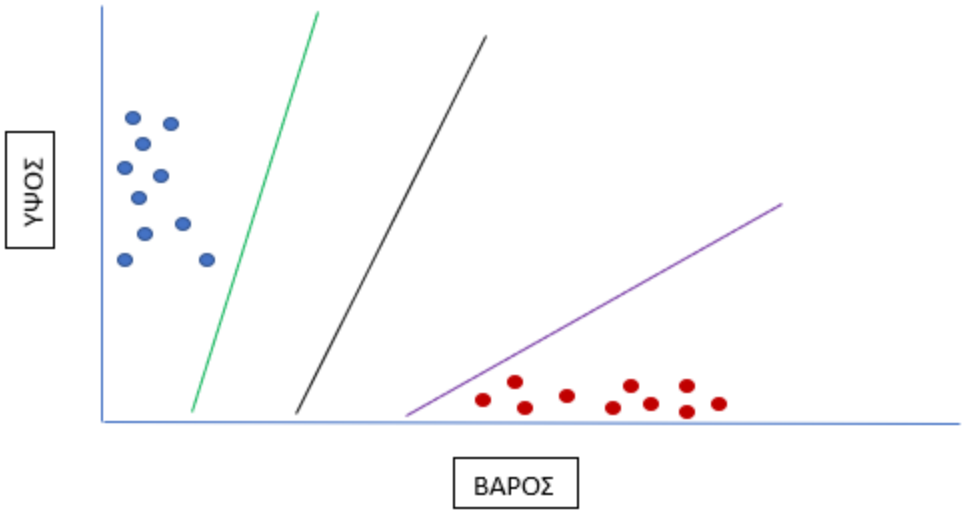


Εικόνα 3.13: Naïve Bayes Visualization

3.6 Support vector machine

Ο τελευταίος από τους αλγορίθμους που χρησιμοποίησα και ίσως ο πιο δύσκολος στο να περιγραφεί και να εξηγηθεί. Είναι ένας ακόμα αλγόριθμος που χρησιμοποιείται για classification. Τα δεδομένα χωρίζονται σε όμοιους πληθυσμούς για να μπορέσουν αργότερα να δημιουργηθούν οι κλάσεις και ορίζονται συντεταγμένες, τοποθετώντας γραμμές σαν σύνορα για να χωρίσουν τα δεδομένα με την μικρότερη απόσταση από το πιο κοντινό αντικείμενο του κάθε group.

Ένα σχετικά απλό παράδειγμα είναι να έχουμε ένα πλήθος μαθητών όπου οι μισοί είναι μαθητές 1^{ης} Δημοτικού και οι υπόλοιποι μαθητές 3^{ης} Λυκείου. Χρησιμοποιώντας ένα δείγμα του πληθυσμού και ορίζοντας κανόνες θέλουμε ο υπολογιστής να μπορέσει να τους διαχωρίσει αναλόγως με την ηλικία τους. Για να το πετύχουμε αυτό μπορούμε να πάρουμε σαν χαρακτηριστικά το βάρος και το ύψος των μαθητών.



Εικόνα 3.14: Παράδειγμα SVM με κλάσεις ύψους και βάρους

Γνωρίζουμε ότι:

Α) Οι μαθητές 1^{ης} Δημοτικού έχουν χαμηλότερο ΜΟ βάρους

Β) Οι μαθητές 3^{ης} Λυκείου έχουν υψηλότερο ΜΟ ύψους

Οπότε αν ένας μαθητής έχει ύψος 175 εκατοστά και βάρος 74 κιλά η πρόβλεψη θα είναι ότι ο συγκεκριμένος μαθητής πηγαίνει στην 3^{η} Λυκείου.

Πλεονεκτήματα:

1) Είναι ένας αλγόριθμος ο οποίος χρησιμοποιείται για προβλήματα που δεν μπορούν να βγουν αποτελέσματα με τη χρήση του αλγορίθμου Logistic Regression αφού χαρακτηρίζεται από την υψηλή του ακρίβεια σε dataset με μη γραμμικά δεδομένα

2) Αποτελεί τον ιδανικό αλγόριθμο για εφαρμογές όπου γίνεται classification κειμένου λόγω της ιδιότητας του να παρέχει υψηλή ακρίβεια

Μειονεκτήματα:

1) Δεν συνιστάται για προβλήματα χωρίς παραδείγματα απαραίτητα για την εκπαίδευση των αλγορίθμων, είναι ένας αλγόριθμος που χαρακτηρίζεται από τη δύσκολη εκπαίδευση του

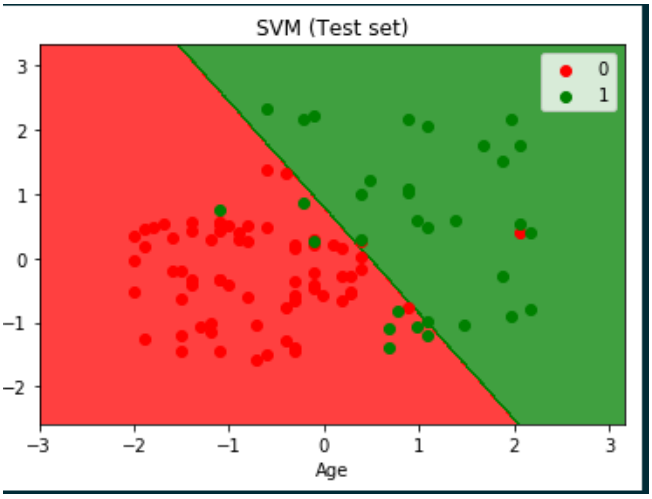
2) Είναι ένας υπολογιστικά δαπανηρός αλγόριθμος καθώς απαιτεί αρκετούς πόρους συστήματος κυρίως μνήμης

Υλοποίηση σε Python:

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

# Fitting SVM to the Training set
from sklearn.svm import SVC
classifier = SVC(kernel = 'linear', random_state = 0)
classifier.fit(X_train, y_train)

# Predicting the Test set results
y_pred = classifier.predict(X_test)

# Making the Confusion Matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
```

Εικόνα 3.15: Τμήμα κώδικα γλώσσας Python για Support Vector Machines

Αναπαράσταση Αποτελεσμάτων:



Εικόνα 3.19: SVM Visualization

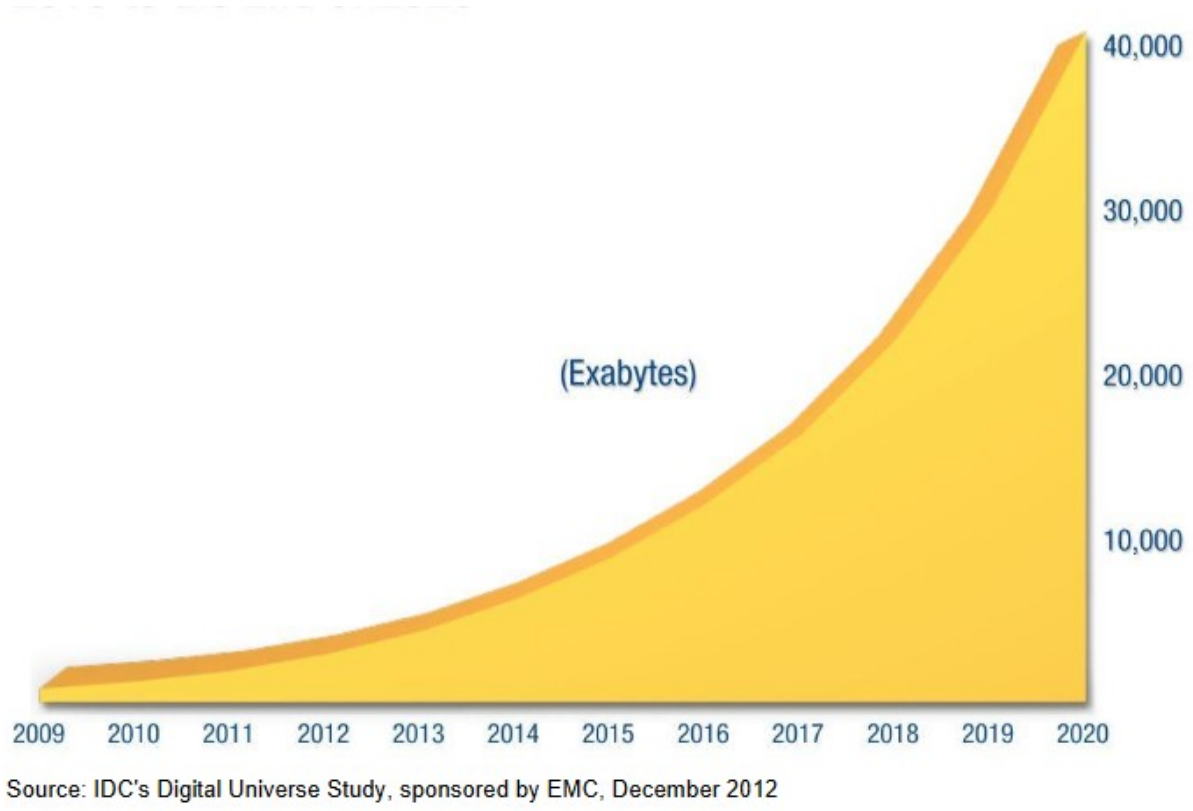
Κεφάλαιο 4

Δεδομένα και Ποιοτική Αποτίμηση Αλγορίθμων

4 Δεδομένα και Ποιοτική Αποτίμηση Αλγορίθμων

4.1 Δεδομένα

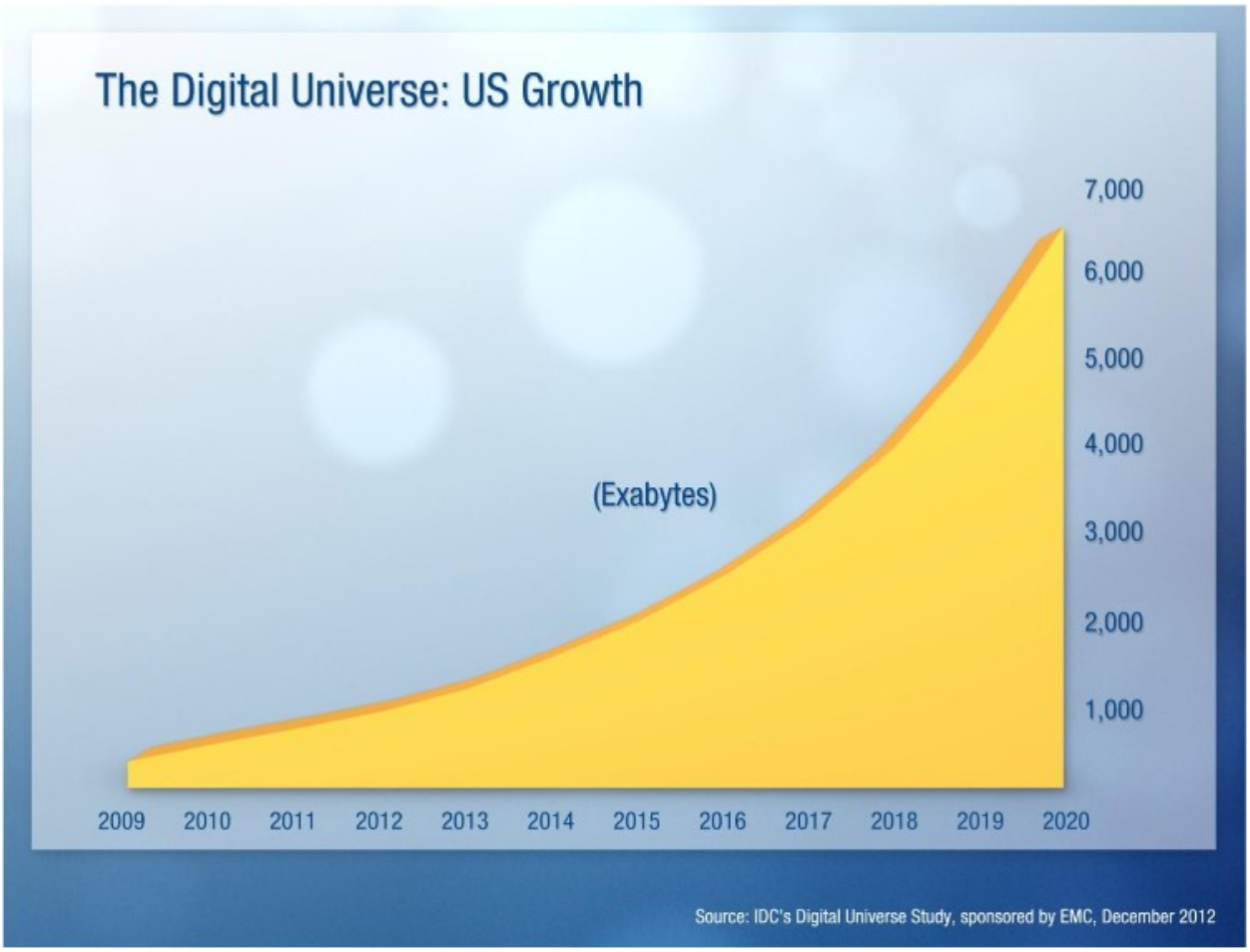
Όπως έχει προαναφερθεί από την αρχή του κόσμου έως το 2005 τα δεδομένα που δημιουργήθηκαν ήταν της τάξης των 130 exabytes.Με τεράστια αύξηση μέσα σε μία δεκαετία όπου οι αριθμοί αγγίζουν τα 1200 exabytes το 2010, τα 7900 exabytes το 2015 και με την εκτίμηση πως ο αριθμός θα είναι το 2020 σχεδόν 41000 exabytes, όλοι οι αριθμοί είναι σύμφωνα με την IDC's Digital Universe Study το 2012.



Εικόνα 4.1: Αύξηση των δεδομένων από το έτος 2009 έως το 2020

Από τα δεδομένα αυτά όπως αναφέρει η έρευνα μόνο ένα ποσοστό της τάξης του 25% μπορεί να αναλυθεί και να είναι χρήσιμο σήμερα και εκτιμάται πως το 2020 το αντίστοιχο ποσοστό θα αυξηθεί στο 33%.

Σύμφωνα με την ίδια έρευνα στις Ηνωμένες Πολιτείες Αμερικής υπάρχει αύξηση περίπου 25% το χρόνο για τα δεδομένα που δημιουργούνται με αποτέλεσμα από το 2012 που υπήρξαν 898 exabytes να αυξηθούν στα 6.6 zettabytes το 2020.



Εικόνα 4.2: Τα δεδομένα στις ΗΠΑ από το 2009 έως το 2020

Κάποιοι από τους λόγους αυτής της αύξησης είναι η χρήση των social media, τα κινητά τηλέφωνα και τα tablets, η καλύτερη ποιότητα σε ταινίες και βιντεοπαιχνίδια, η ψηφιακή τηλεόραση και κυρίως το γεγονός πως πλέον σχεδόν κάθε σπίτι έχει έναν ηλεκτρονικό υπολογιστή με σύνδεση στο διαδίκτυο. Από τα συγκεκριμένα παραδείγματα μπορούμε να αναλύσουμε δεδομένα τα οποία θα είναι χρήσιμα όπως τι είναι αυτό που θα ψάξει κάποιος στα social media και θα του αρέσει ή δεδομένα τα οποία δεν έχουν κάποια ιδιαίτερη σημασία, όπως για παράδειγμα πάλι τα social media, όπου δύο άνθρωποι ανταλλάσσουν την ίδια φωτογραφία πολλές φορές σε ένα chat μεταξύ τους προσπαθώντας απλώς να περάσουν την ώρα τους.

Η όλη ουσία δηλαδή είναι να μπορέσουμε να διαχωρίσουμε το πλήθος όλων αυτών των δεδομένων, να τα αναλύσουμε και να αποκομίσουμε κάτι από αυτά.

4.2 Υιοθετούμενα Σύνολα Δεδομένων

Προκειμένου να γίνουν πειράματα για τον καλύτερο έλεγχο των αλγορίθμων και της εφαρμογής σαν σύνολο χρησιμοποιήθηκαν datasets τα οποία είναι διαθέσιμα στην ιστοσελίδα https://archive.ics.uci.edu/ml/index.php και του βασικού dataset της εφαρμογής το οποίο περιέχει τις μετρήσεις για τη φωτιά. Οι αλγόριθμοι της εφαρμογής είναι Logistic Regression, Linear Discriminant Analysis, K Nearest Neighbors Classifier, Decision Tree Classifier, Naive Bayes classifier και Support vector machine.

Η συγκεκριμένη επιλογή έγινε μετά από δοκιμές σε ένα μεγάλο σύνολο αλγορίθμων μηχανικής μάθησης για classification και οι προαναφερόμενοι πέτυχαν τα υψηλότερα ποσοστά ακρίβειας σε παραπλήσια datasets αλλά και στο συγκεκριμένο που αφορά τη φωτιά και απαιτείται για την υλοποίηση της διπλωματικής εργασίας.

Dataset Φωτιάς

Για τη βασική μελέτη η οποία θα υλοποιήσει το dataset της εφαρμογής χρησιμοποιούνται περιβαλλοντικές μετρήσεις οι οποίες μετά το classification θα δημιουργήσουν κλάσεις οι οποίες θα φανερώνουν το βαθμό επικινδυνότητάς να ξεσπάσει πυρκαγιά. Περιέχει στοιχεία όπως η θερμοκρασία, η υγρασία, το επίπεδο του οξυγόνου και του διοξειδίου του άνθρακα και μέσα από ένα σύνολο αλγορίθμων θα πραγματοποιεί classification αναλόγως με την πιθανότητα ξεσπάσει πυρκαγιά σε κλάσεις από το ένα, ως τη μικρότερη πιθανότητα, έως την κλάση δέκα, ως τη μεγαλύτερη.

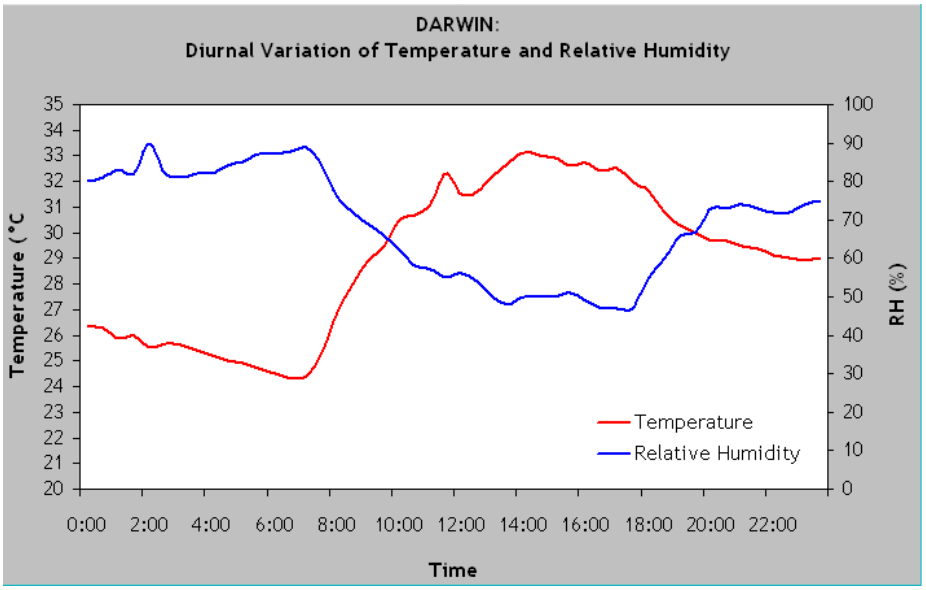
Θερμοκρασία	Σχετική Υγρασία	CO2	Επίπεδο O2
20	0.17	410	0.21
26	0.15	410	0.21
32	0.12	410	0.21
19	0.18	408	0.21
16	0.19	407	0.21
22	0.18	409	0.21
40	0.11	411	0.21
11	0.31	405	0.19
15	0.22	403	0.21
31	0.13	409	0.21
23	0.15	410	0.21
25	0.15	410	0.21
27	0.11	410	0.21
14	0.16	409	0.21
16	0.21	408	0.21
18	0.17	409	0.21
33	0.15	410	0.21

Πίνακας 4.1 : Το dataset με τις μετρήσεις

Οι μετρήσεις έχουν γίνει, αποθηκεύτηκαν σε ένα αρχείο CSV και φορτώθηκαν στο πρόγραμμα προκειμένου να αναλυθούν από τους αλγορίθμους. Για το πώς μπορεί η μηχανική μάθηση ακολουθούν λίγα λόγια για επεξήγηση.

Η θερμοκρασία είναι ο σημαντικότερος παράγοντας που επηρεάζει την φωτιά. Οι ακτίνες του ηλίου αυξάνουν τη θερμοκρασία στην επιφάνεια της γης, κάτι που έχει σαν αποτέλεσμα την ευκολότερη ανάφλεξη και την αύξηση της ταχύτητας διάδοσης της φωτιάς. Έχει παρατηρηθεί πώς η διάδοση πυρκαγιάς και το ύψος της είναι σαφώς υψηλότερα νωρίς το απόγευμα, δηλαδή ώρες όπου η θερμοκρασία είναι στην υψηλότερη τιμή της για την ημέρα. Επίσης η θερμοκρασία είναι ο παράγοντας που επηρεάζει την σχετική υγρασία η οποία είναι η δεύτερη σημαντικότερη μέτρηση μας.

Η σχετική υγρασία είναι η ποσότητα των υδρατμών που εμπεριέχεται στην ατμόσφαιρα προς το βάρος των υδρατμών που μπορεί να συμπεριλάβει μέχρις ότου αυτός να καθίσταται κεκορεσμένος. Η μέτρηση μπορεί να γίνει με έναν υγρογράφο ή ένα υγρόμετρο. Όσο χαμηλότερη είναι η σχετική υγρασία τόσο πιο εύκολο είναι να πραγματοποιηθεί μία ανάφλεξη. Σχετικά με τη θερμοκρασία είναι αντιστρόφως ανάλογα μεγέθη, δηλαδή η αύξηση της θερμοκρασίας προκαλεί μείωση της σχετικής υγρασίας.



Εικόνα 4.3: Θερμοκρασία σε αναλογία με σχετική υγρασία, source http://learnline.cdu.edu.au

Συνεπώς σύμφωνα με το παραπάνω παράδειγμα οι δύο από τους βασικότερους παράγοντες που θα αναλύσουν οι αλγόριθμοι προκειμένου να κατατάξουν τα αποτελέσματα σε κλάσεις είναι η θερμοκρασία και η σχετική υγρασία.

Ακολουθεί εικόνα από το αρχείο CSV στο οποίο μετά την ανάλυση εμφανίζονται και οι σχετικές κλάσεις που προέκυψαν από το classification.

25	0.15	410	0.21	6
27	0.16	410	0.21	7
8	0.32	405	0.21	6
1	0.89	403	0.17	5
5	0.65	406	0.18	5
28	0.19	409	0.2	7
30	0.19	410	0.21	7
14	0.19	407	0.21	6
8	0.32	405	0.21	5
1	0.89	403	0.17	5
5	0.65	406	0.18	5
28	0.19	409	0.2	7
30	0.19	410	0.21	7
39	0.08	411	0.21	9

Εικόνα 4.4: Το αρχείο CSV με τις κλάσεις

Iris Dataset

Αποτελεί ένα από τα πιο γνωστά dataset που είναι διαθέσιμα στο διαδίκτυο από τον R.A Fisher, το οποίο έχει τα μήκη και πλάτη του φύλλου και του πέταλου ενός λουλουδιού και τρείς κλάσεις για τον διαχωρισμό τους.

Blood Transfusion Dataset

Τα δεδομένα προέρχονται από κέντρο αιμοδοσίας της πόλης Hsin-Chu στην Ταιβάν. Τα δεδομένα αυτά περιλαμβάνουν τους μήνες που έχουν περάσει από την τελευταία δωρεά, το συνολικό αριθμό δωρεών, την ποσότητα του αίματος σε cc, τους μήνες που έχουν περάσει από την πρώτη δωρεά και τέλος αν έχει δώσει αίμα τον μήνα Μάρτιο του 2007.

Lenses Dataset

Το συγκεκριμένο dataset περιέχει την ηλικία του ασθενή η οποία διαχωρίζεται σε 1 για νεαρή, 2 για προ πρεσβυωπική, 3 για πρεσβυωπική, την συνταγή γυαλιών με 1 για μυωπία και 2 για υπερμετρωπία, αστιγματισμό με 1 για όχι και 2 για ναι, παραγωγή δακρύων με 1 για μειωμένη και 2 για κανονική και τρείς κλάσεις όπου στην πρώτη ο ασθενής χρειάζεται σκληρούς φακούς επαφής, στη δεύτερη χρειάζεται μαλακούς φακούς επαφής και στην τρίτη και τελευταία κλάση ο ασθενής δεν πρέπει να φορέσει φακούς επαφής.

4.3 Μετρικές Απόδοσης

Μετά τη φόρτωση του κάθε dataset στην εφαρμογή πραγματοποιείται σύγκριση των έξι αλγορίθμων προκειμένου να παραχθούν τα αποτελέσματα του classification, με την αποτίμηση τους να βασίζεται στην αρχική εκτιμώμενη ακρίβεια, την ακρίβεια του validation set, το precision και το recall.

4.3.1 Εκτιμώμενη Ακρίβεια

Προκειμένου να βρούμε την εκτιμώμενη ακρίβεια πρέπει να δημιουργηθεί ένα validation dataset. Τα δεδομένα θα χωριστούν σε δύο μέρη με το μεγαλύτερο ποσοστό να αξιοποιείται για την εκπαίδευση των αλγορίθμων και το μικρότερο για το validation dataset. Όταν λέμε για ποσοστά συνήθως είναι 70 με 80% για το training και 20 με 30% για το validation. Εν συνεχεία πραγματοποιείται cross validation όπου εμφανίζεται η αναλογία των σωστών προβλέψεων σε συνάρτηση με τον αριθμό των συνολικών εγγραφών του dataset ώστε να δώσει ένα ποσοστό εκτιμώμενης ακρίβειας προβλέψεων.

4.3.2 Ακρίβεια

Ονομάζεται ο έλεγχος ενός μοντέλου αλγορίθμου μηχανικής μάθησης σε βαθμονομημένα δεδομένα στα οποία πραγματοποιείται σύγκριση της εξόδου τους με το output που παρήγαγε ο αλγόριθμος αυτός. Το ποσοστό της ομοιότητας των δύο εξόδων ονομάζεται ακρίβεια, αποτελεί δηλαδή μία ένδειξη για το πόσο ποιοτικό και ακριβές είναι το μοντέλο που δημιουργήσαμε. Ο τρόπος υπολογισμού εξαρτάται από το πώς έχει αναγνωριστεί το κάθε δεδομένο, δηλαδή αν είναι True Positive και θεωρείται True Positive, False Negative αν είναι negative και έχει χαρακτηριστεί ως positive, True Negative, αν είναι negative και έχει χαρακτηριστεί ως negative kai τέλος False Negative αν είναι positive και έχει αναγνωριστεί ως negative.

Η φόρμουλα υπολογισμού είναι, ακρίβεια = (TP + TN)/(TP + TN + FP + FN) οπότε αν για παράδειγμα έχουμε τον ακόλουθο πίνακα αποτελεσμάτων με 5 TP, 10 FN, 30 FP και 50 FN θα ισχύσει:

Ακρίβεια = (5 + 50) / (5 + 50 + 30+ 10)

Ακρίβεια = 55 / 95

Άρα η ακρίβεια είναι 57.8%

	Labeled Positive	Labeled Negative
Class -> Positive	5	10
Class -> Negative	30	50

Πίνακας 4.2: Αποτελέσματα αναγνώρισης δεδομένων

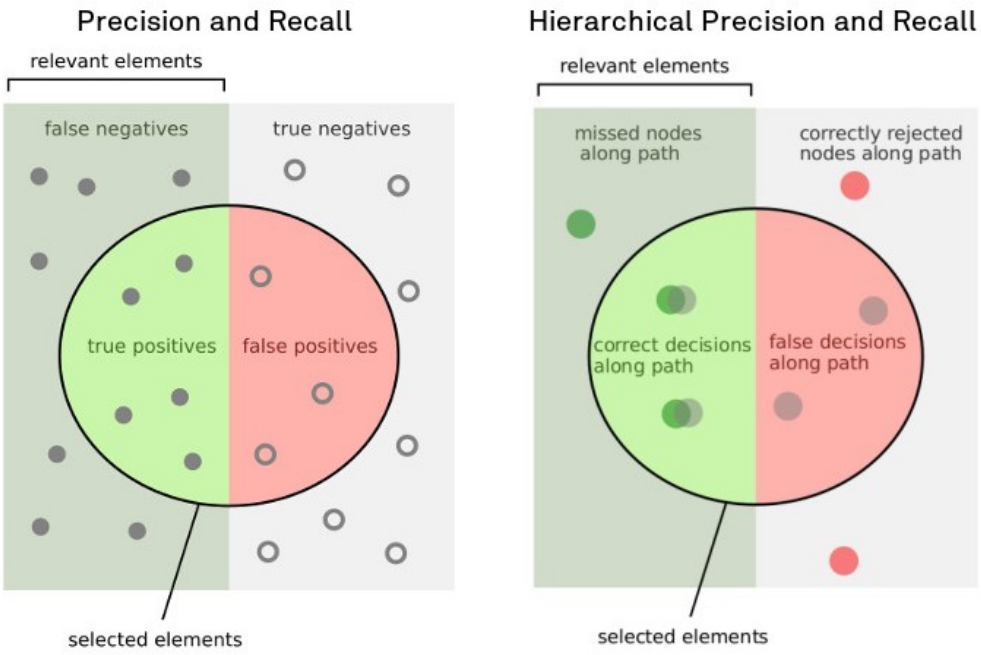
4.3.3 Precision & Recall

Οι μετρικές Precision και Recall είναι άμεσα συνδεδεμένες με την ακρίβεια καθώς ελέγχουν τα True Positives, False Negatives, True Negatives, False Positives. Το Precision αφορά τις σχετικές εγγραφές από το σύνολο των εγγραφών που ανακτήθηκαν, ενώ το Recall τις σχετικές εγγραφές από το σύνολο των σχετικών εγγραφών. Συνεπώς ισχύουν οι ακόλουθες σχέσεις:

Precision = TP / TP + FP

Recall = TP/ TP + FN

Ως παράδειγμα έχουμε ένα σύνολο από πορτοκάλια και μήλα και ο στόχος του προγράμματος είναι να αναγνωρίσει τα μήλα. Αν υποθέσουμε ότι από τα 20 φρούτα τα 11 είναι μήλα και το πρόγραμμα καταφέρει να αναγνωρίσει 10 φρούτα ως μήλα αλλά τα 2 από αυτά είναι πορτοκάλια τότε ισχύει ότι έχουμε 8 True Positives και 2 False Positives. Συνεπώς το Precision της εφαρμογής είναι 8/10 και το Recall είναι 8/11.



Εικόνα 4.5. :Precision & Recall source https://idalab.de/blog/data-science/hierarchical-metrics

4.4 Αποτίμηση Μέσω Προσομοιώσεων

Ακολουθούν οι πίνακες με την αρχική εκτιμώμενη ακρίβεια, την ακρίβεια του validation set, το precision και το recall για κάθε έναν από τους αλγορίθμους σε όλα τα dataset.

Fire Dataset

	Εκτιμώμενη Ακρίβεια	Ακρίβεια	Precision	Recall
LR	71%	62%	60%	62%
LDA	81%	93%	100%	100%
KNN	82%	93%	95%	94%
DT	90%	100%	100%	100%
NB	71%	93%	95%	94%
SVM	84%	93%	100%	100%

Πίνακας 4.3 : Σύγκριση Αλγορίθμων dataset φωτιάς

Τα αποτελέσματα δείχνουν πως ο ιδανικός αλγόριθμος είναι ο Decision Tree με εκτιμώμενη ακρίβεια 90% και ποσοστό 100% σε ακρίβεια, precision και recall.

Iris Dataset

	Εκτιμώμενη Ακρίβεια	Ακρίβεια	Precision	Recall
LR	95%	80%	83%	80%
LDA	97%	96%	97%	97%
KNN	98%	90%	90%	82%
DT	97%	86%	87%	87%
NB	96%	83%	84%	83%
SVM	99%	93%	94%	93%

Πίνακας 4.4 : Σύγκριση Αλγορίθμων iris dataset

Τα αποτελέσματα δείχνουν πως ο ιδανικός αλγόριθμος είναι ο Support Vector Machine με εκτιμώμενη ακρίβεια 99%, μετά από δοκιμή και των έξι αλγορίθμων όμως ο αλγόριθμος Linear Discriminant Analysis ο οποίος είχε εκτιμώμενη ακρίβεια 97%, έχει σαφώς καλύτερα αποτελέσματα με ακρίβεια 96% και ποσοστό 97% σε precision και recall.

Blood Transfusion Dataset

	Εκτιμώμενη Ακρίβεια	Ακρίβεια	Precision	Recall
LR	45%	87%	77%	88%
LDA	45%	87%	77%	88%
KNN	38%	62%	73%	68%
DT	67%	50%	70%	50%
NB	55%	62%	91%	62%
SVM	58%	87%	77%	88%

Πίνακας 4.5: Σύγκριση Αλγορίθμων blood transfusion dataset

Τα αποτελέσματα δείχνουν πως ο ιδανικός αλγόριθμος είναι ο Support Vector Machine με εκτιμώμενη ακρίβεια 67%, μετά από δοκιμή και των έξι αλγορίθμων όμως οι αλγόριθμοι Linear Discriminant Analysis, Logistic Regression και Support Vector Machine έχουν καλύτερα αποτελέσματα με ακρίβεια 87% ,ποσοστό 77% σε precision και 88% σε recall.

Lenses Dataset

	Εκτιμώμενη Ακρίβεια	Ακρίβεια	Precision	Recall
LR	80%	20%	5%	20%
LDA	63%	80%	90%	80%
KNN	69%	40%	17%	40%
DT	75%	60%	87%	60%
NB	63%	100%	100%	100%
SVM	75%	20%	4%	20%

Πίνακας 4.6 : Σύγκριση Αλγορίθμων lenses dataset

Τα αποτελέσματα δείχνουν πως ο ιδανικός αλγόριθμος είναι ο Logistic Regression με εκτιμώμενη ακρίβεια 67%, μετά από δοκιμή και των έξι αλγορίθμων όμως ο αλγόριθμος Naive Bayes έχει ακρίβεια 100% ,ποσοστό 100% σε precision και 100% σε recall.

Κεφάλαιο 5

Το Πρόγραμμα

5 Το Πρόγραμμα

5.1 Περιγραφή

Σαν μια σύντομη περιγραφή το πρόγραμμα αρχικά φορτώνει τις απαραίτητες βιβλιοθήκες, εμφανίζει στον χρήστη ένα μήνυμα που ζητάει να γίνει φόρτωση του csv αρχείου με τα δεδομένα και πραγματοποιεί την φόρτωση. Στη συνέχεια εμφανίζει κάποιες λεπτομέρειες για τα δεδομένα όπως μια σύνοψη των στοιχείων, μια προεπισκόπηση και το σύνολο των στοιχείων ανά κλάση. Σαν τέλος του αρχικού σταδίου παρουσιάζει τα δεδομένα με διαγράμματα τύπου Box and Whisker αλλά και Histograms. Στη συνέχεια κάνει διαχωρισμό των δεδομένων, εισάγει τα μοντέλα των αλγορίθμων και τα συγκρίνει για να βγάλει το ποσοστό επιτυχίας του καθενός πριν κάνει και την γραφική αναπαράσταση. Τέλος ελέγχει την ακρίβεια, τα λάθη και εμφανίζει και το τελικό Classification Report για κάθε ένα από τα μοντέλα.

5.2 Οι βιβλιοθήκες

Η γλώσσα προγραμματισμού Python μαζί με την γλώσσα R θεωρείται από τις καλύτερες, ευέλικτες και πιο δυνατές γλώσσες για να ασχοληθεί ένας ερευνητής που τον ενδιαφέρει το πεδίο της μηχανικής μάθησης. Για να επιτευχθεί όμως αυτό απαιτούνται κάποιες βιβλιοθήκες οι οποίες θα ενεργοποιήσουν τον απαραίτητο κώδικα στην Python. Στο πρόγραμμα που υλοποίησα χρησιμοποιώ τις βιβλιοθήκες Pandas, Matplotlib και Scikit-learn.

```
import pandas as pd
from pandas.plotting import scatter_matrix
import matplotlib.pyplot as plt
from sklearn import model_selection
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
```

Εικόνα 5.1: Οι βιβλιοθήκες του συστήματος

5.2.1 Pandas

Είναι μία Open source βιβλιοθήκη που δημιουργήθηκε από τον Wes McKinney και η ίσως η πιο απαραίτητη για όποιον θέλει να ασχοληθεί με την ανάλυση των δεδομένων μέσω της Python. Αυτό που κάνει η συγκεκριμένη βιβλιοθήκη είναι ότι παίρνει ένα αρχείο TSV, μία ολόκληρη βάση δεδομένων SQL ή πιο συχνά ένα αρχείο CSV και το μετατρέπει σε ένα data frame, δηλαδή ένα αντικείμενο της Python που αποτελείται από γραμμές και στήλες.

Οι κύριοι λόγοι που θα χρησιμοποιηθεί αυτή η βιβλιοθήκη είναι για να διαβάσει το αρχείο και να κάνει τις απαραίτητες μετατροπές και είναι αυτοί που ακολουθούν:

1) Μετατροπή μιας λίστας ή λεξικού σε data frame
2) Άνοιγμα ενός τοπικού αρχείου από τον υπολογιστή
3) Άνοιγμα ενός αρχείου από μια ιστοσελίδα ή server

Στο πρόγραμμα που υλοποίησα χρησιμοποίησα το Pandas για να διαβάσω το CSV αρχείο και για το visualization.

```
import pandas as pd
from pandas.plotting import scatter_matrix
```

Εικόνα 5.2: Φόρτωση της βιβλιοθήκης Pandas

```
print("Απαιτείται φόρτωση αρχείου")
filename = input('Προορισμός αρχείου: ')
print("")

# fortwsi arxeiou
dataset = pd.read_csv(filename)
```

Εικόνα 5.3: Μήνυμα προς το χρήστη και φόρτωση του αρχείου CSV

5.2.2 Matplotlib

Η δεύτερη βιβλιοθήκη δημιουργήθηκε τον John D. Hunter και είναι μια βιβλιοθήκη απαραίτητη για το visualization των αποτελεσμάτων. Προσφέρει τη δυνατότητα στον χρήστη να δημιουργεί plots για τις εφαρμογές του μέσω ενός API που αξιοποιεί Toolkits όπως το Tkinter. Η Matplotlib είναι συνδεδεμένη με το MATLAB γιατί η δημιουργία της αποσκοπεί στο να δίνεται στον χρήστη ένα Interface παρόμοιο του MATLAB όσο κάνει χρήση της γλώσσας Python. Στο πρόγραμμα έγινε χρήση της για το visualization.

5.2.3 Scikit-learn

Πρωτοεμφανίστηκε το 2007 σαν project για το Google Summer of code από τον David Cournapeau. Είναι μια δωρεάν βιβλιοθήκη η οποία καλείται να δημιουργήσει αλγορίθμους για Regression, Classification και Clustering. Είναι αρκετά εύκολη στη χρήση και δημιουργώντας ένα μοντέλο αλγορίθμου με μερικές αλλαγές και παραμετροποιήσεις μπορείς πολύ γρήγορα να φτιάξεις και τα υπόλοιπα μοντέλα. Στο πρόγραμμα μου τη χρησιμοποίησα για να εισάγω του αλγορίθμους, για το classification report, το accuracy score και το confusion matrix.

```
from sklearn import model_selection
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
```

Εικόνα 5.4: Οι βιβλιοθήκες του πακέτου Scikit-learn

5.3 Τα Στάδια του Προγράμματος

5.3.1 Στάδιο 1

Αρχικά οι πρώτες γραμμές κώδικα εμφανίζουν των αριθμό των εγγραφών από το CSV αρχείο και τα attributes για τις εγγραφές αυτές. Αριστερά είναι το τμήμα του κώδικα και δεξιά τα αποτελέσματα.

```
# to sunolo twn stoixeiwn kai twn attributes
print('-------------------------')
print('Instances και Attributes')
print('-------------------------')
print("")
print(dataset.shape)
print("")
```

Εικόνα 5.5: Κώδικας για Instances-Attributes

```
-------------------------
Instances και Attributes
-------------------------

(79, 5)
```

Εικόνα 5.6: Εμφάνιση Instances-Attributes

Στη συνέχεια γίνεται μία προεπισκόπηση των δέκα πρώτων στοιχείων σαν μια δεύτερη επιβεβαίωση για τον χρήστη ότι τα στοιχεία έχουν φορτωθεί στο πρόγραμμα και εμφανίζονται σωστά. Ακολουθεί ο κώδικας και το αποτέλεσμα.

```
# proepiskopoisi
print('--------------------------------')
print('Προεπισκόπηση 10 πρώτων στοιχείων')
print('--------------------------------')
print("")
print(dataset.head(10))
print("")
```

Εικόνα 5.7: Κώδικας για την προεπισκόπηση των στοιχείων

```
--------------------------------
Προεπισκόπηση 10 πρώτων στοιχείων
--------------------------------

   temperature  relative humidity  CO2  O2 level  CLASS
0           20               0.17  410      0.21      6
1           26               0.15  410      0.21      7
2           32               0.12  410      0.21      8
3           19               0.18  408      0.21      6
4           16               0.19  407      0.21      6
5           22               0.18  409      0.21      6
6           40               0.11  411      0.21      9
7           11               0.31  405      0.19      6
8           15               0.22  403      0.21      6
9           31               0.13  409      0.21      8
```

Εικόνα 6.8: Εμφάνιση των 10 πρώτων στοιχείων

Ακολουθεί μία σύνοψη των στοιχείων με πληροφορίες όπως ο μέσος όρος, το μέγιστο, το ελάχιστο και ποσοστιαίες τιμές.

```
# plirofories
print("")
print('-----------------')
print('Σύνοψη στοιχείων')
print('-----------------')
print(dataset.describe())
print("")
```

Εικόνα 5.9: Κώδικας για περισσότερες πληροφορίες των στοιχείων

```
-----------------
Σύνοψη στοιχείων
-----------------
       temperature  relative humidity         CO2   O2 level      CLASS
count    79.000000          79.000000   79.000000  79.000000  79.000000
mean     22.177215           0.257848  408.265823   0.202532   6.607595
std      12.324418           0.218706    2.654019   0.013725   1.304998
min       1.000000           0.080000  402.000000   0.160000   5.000000
25%      10.500000           0.130000  406.000000   0.200000   6.000000
50%      26.000000           0.160000  410.000000   0.210000   6.000000
75%      31.000000           0.310000  410.000000   0.210000   8.000000
max      40.000000           0.890000  411.000000   0.210000   9.000000
```

Εικόνα 5.10: Εμφάνιση πληροφοριών των στοιχείων

Και τέλος ο κώδικας που θα αντιστοιχήσει τα στοιχεία με τις κλάσεις και το αποτέλεσμα που βγάζει το πρόγραμμα στην οθόνη του χρήστη.

```
# oi klaseis
print('-------------------------')
print('Σύνολο Στοιχείων ανα κλάση')
print('-------------------------')
print("")
print(dataset.groupby('CLASS').size())
print("")
```

Εικόνα 5.11: Κώδικας για κατάταξη στοιχείων ανά κλάση

```
-------------------------
Σύνολο Στοιχείων ανα κλάση
-------------------------

CLASS
5    18
6    25
7    15
8    12
9     9
```

Εικόνα 5.12: Εμφάνιση στοιχείων ανά κλάση

5.3.2 Στάδιο 2

Στο επόμενο στάδιο το πρόγραμμα δημιουργεί τους πίνακες για τα δεδομένα και τα χωρίζει σε train και validation sets προκειμένου να γίνει η εκπαίδευση των αλγορίθμων.

```
# diaxwrismos dedomenwn
array = dataset.values
X = array[:,:-1]
Y = array[:,4]
validation_size = 0.20
seed = 7
X_train, X_validation, Y_train, Y_validation = model_selection.train_test_split(X, Y, test_size=validation_size, random_state=seed)
```

Εικόνα 5.13: Δημιουργία των dataframes

Στη συνέχεια εισάγονται οι αλγόριθμοι Logistic Regression, Linear Discriminant Analysis, K-Neighbors Classifier, Decision Tree Classifier, Naïve Bayes και Support Vector Machines.

```
models = []
models.append(('Logistic Regression', LogisticRegression()))
models.append(('Linear Discriminant Analysis', LinearDiscriminantAnalysis()))
models.append(('K-Neighbors Classifier', KNeighborsClassifier()))
models.append(('Decision Tree Classifier', DecisionTreeClassifier()))
models.append(('Gaussian NB', GaussianNB()))
models.append(('SVM', SVC()))
```

Εικόνα 5.14: Τα μοντέλα των αλγορίθμων

Αμέσως μετά την εισαγωγή τα δεδομένα θα δοκιμαστούν σε κάθε έναν από τους αλγορίθμους για να γίνει η σύγκριση μεταξύ τους.

```
results = []
names = []
for name, model in models:
    kfold = model_selection.KFold(n_splits=6, random_state=seed)
    cv_results = model_selection.cross_val_score(model, X_train, Y_train, cv=kfold, scoring=scoring)
    results.append(cv_results)
    names.append(name)
    msg = "%s: %f (%f)" % (name, cv_results.mean(), cv_results.std())
    print(msg)
```

Εικόνα 5.15: Σύγκριση των αλγορίθμων

Παρακάτω εμφανίζονται τα αποτελέσματα της σύγκρισης.

```
--------------------
Έλεγχος Αλγορίθμων
--------------------

Logistic Regression: 0.716667 (0.156619)
Linear Discriminant Analysis: 0.810606 (0.076676)
K-Neighbors Classifier: 0.827273 (0.083154)
Decision Tree Classifier: 0.906061 (0.052661)
Gaussian NB: 0.715152 (0.135824)
SVM: 0.843939 (0.085991)
```

Εικόνα 5.16: Τα αποτελέσματα της σύγκρισης των αλγορίθμων

Άρα τα στοιχεία που λαμβάνουμε είναι τα εξής:

Logistic Regression → 71%

Linear Discriminant Analysis → 81%

K-Neighbors Classifier → 82%

Decision Tree Classifier → 90%

Naïve Bayes → 71%

Support Vector Machines → 84%

Συνεπώς ο βέλτιστος αλγόριθμος για τα δεδομένα που χρησιμοποιήσαμε είναι ο Decision Tree Classifier με ποσοστό 90% και ακολουθούν ο Support Vector Machines

με 84% και ο K-Neighbors Classifier 82%.

5.3.3 Στάδιο 3

Στο τελευταίο στάδιο του προγράμματος, πραγματοποιούνται οι προβλέψεις στο validation dataset me όλους τους αλγορίθμους και εμφανίζεται η ακρίβεια, ο έλεγχος των λαθών με το confusion matrix και τέλος το classification report.

```
svm = SVC()
svm.fit(X_train, Y_train)
predictions = svm.predict(X_validation)
print("")
print('-----------------------------------------------')
print('Accuracy score με SVM στο validation dataset')
print('-----------------------------------------------')
print("")
print(accuracy_score(Y_validation, predictions))
print("")
print('----------------------------------')
print('Έλεγχος λαθών με Confusion Matrix')
print('----------------------------------')
print("")
print(confusion_matrix(Y_validation, predictions))
print("")
print('----------------------')
print('Classification Report')
print('----------------------')
print("")
print(classification_report(Y_validation, predictions))
```

Εικόνα 5.17: Οι προβλέψεις στο validation dataset

```
-----------------------------------------
Accuracy score με SVM στο validation dataset
-----------------------------------------

0.9375
```

Εικόνα 5.18: Το accuracy score

```
----------------------------------
Έλεγχος λαθών με Confusion Matrix
----------------------------------

[[3 0 0 0 0]
 [0 4 0 0 0]
 [0 1 3 0 0]
 [0 0 0 3 0]
 [0 0 0 0 2]]
```

Εικόνα 5.19: Confusion Matrix

```
----------------------
Classification Report
----------------------

             precision    recall  f1-score   support

        5.0       1.00      1.00      1.00         3
        6.0       0.80      1.00      0.89         4
        7.0       1.00      0.75      0.86         4
        8.0       1.00      1.00      1.00         3
        9.0       1.00      1.00      1.00         2

avg / total       0.95      0.94      0.94        16
```

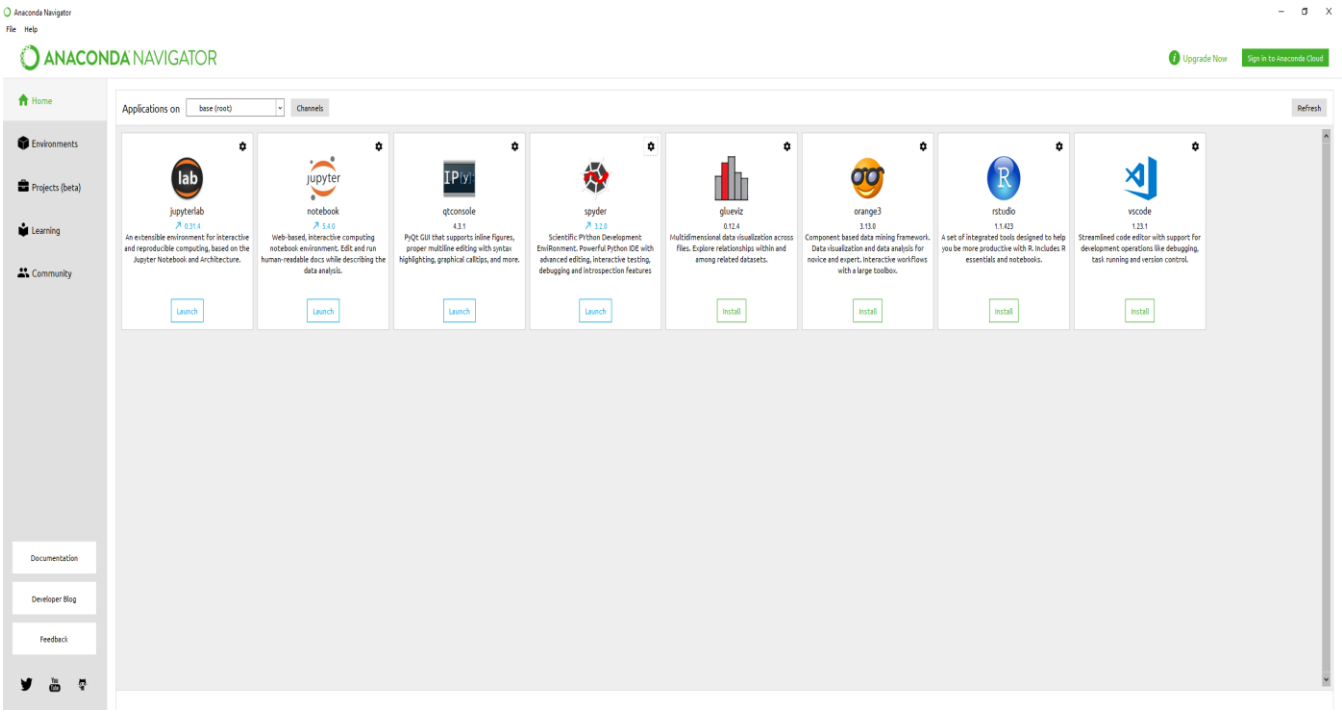
Εικόνα 5.20: Classification Report

5.4 Εργαλεία και Περιβάλλον Εργασίας

5.4.1 Anaconda

Το Anaconda αποτελεί το πρώτο και σημαντικότερο εφόδιο για τον ερευνητή που θέλει να ασχοληθεί με τη μηχανική μάθηση και γενικότερα με την επιστήμη των δεδομένων μέσω της γλώσσας Python και ανεξαρτήτως λειτουργικού συστήματος. Τα χαρακτηριστικά που κάνουν το Anaconda να ξεχωρίζει είναι η αυτόματη εγκατάσταση της γλώσσας στο μηχάνημα, το πλήθος των προ εγκατεστημένων packages αλλά και περισσότερα από 1500 επιπλέον που μπορεί να κατεβάσει κάποιος δωρεάν και τέλος η ενσωμάτωση του πακέτου Conda το οποίο βοηθά στην εγκατάσταση επιπλέον complex περιβαλλόντων όπως το Scikit-learn, το οποίο χρησιμοποιήθηκε και για την ανάπτυξη του προγράμματος της διπλωματικής εργασίας.

Επιπρόσθετα υπάρχουν ήδη προ εγκατεστημένα δύο επιπλέον λογισμικά τα οποία μάλιστα χρησιμοποίησα το Spyder και το Jupiter Notebooks.

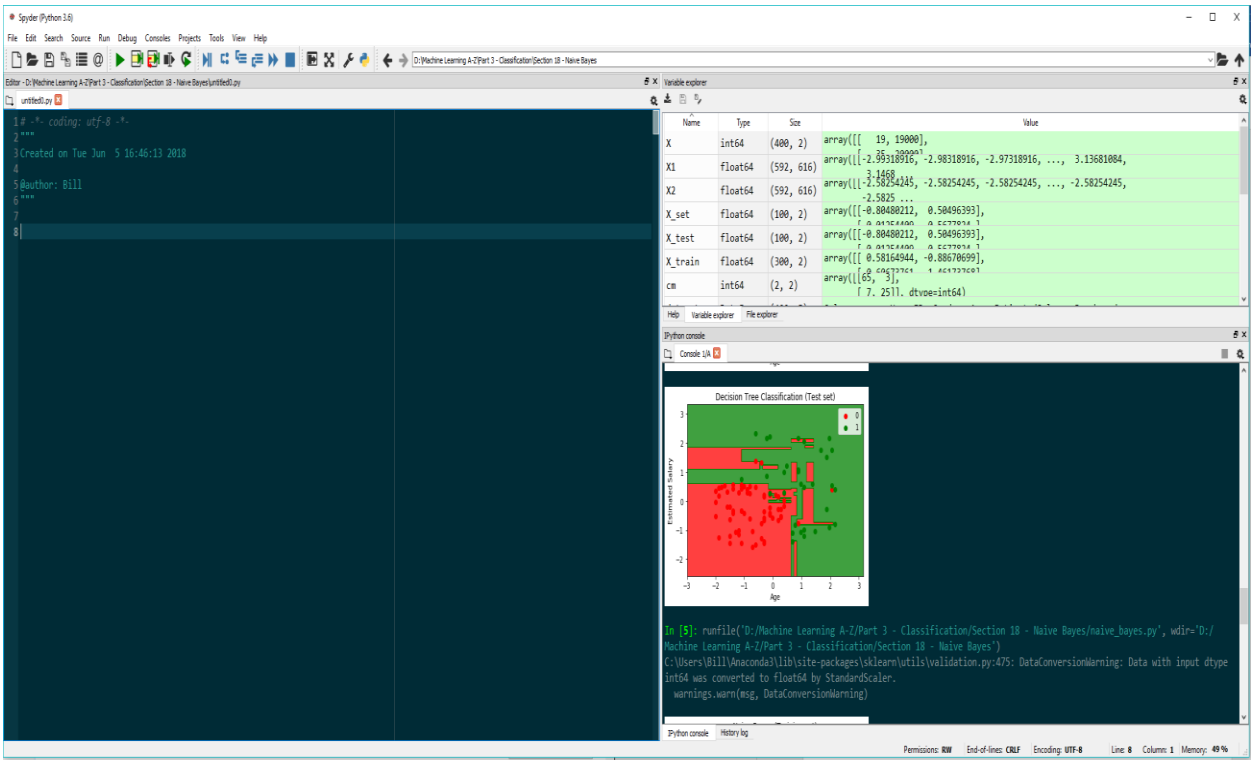


Εικόνα 5.21: Η αρχική οθόνη του Anaconda

5.4.2 Spyder

Αποτέλεσε το βασικό περιβάλλον ανάπτυξης της εφαρμογής μου. Είναι μία σουίτα πλήρως εξοπλισμένη με plugins απαραίτητα για πειράματα μηχανικής μάθησης και ενσωματώνει τα βασικότερα τα οποία είναι το SciPy, το iPython και το Matplotlib. Κάποια από τα στοιχεία που το κάνουν να ξεχωρίζει είναι:

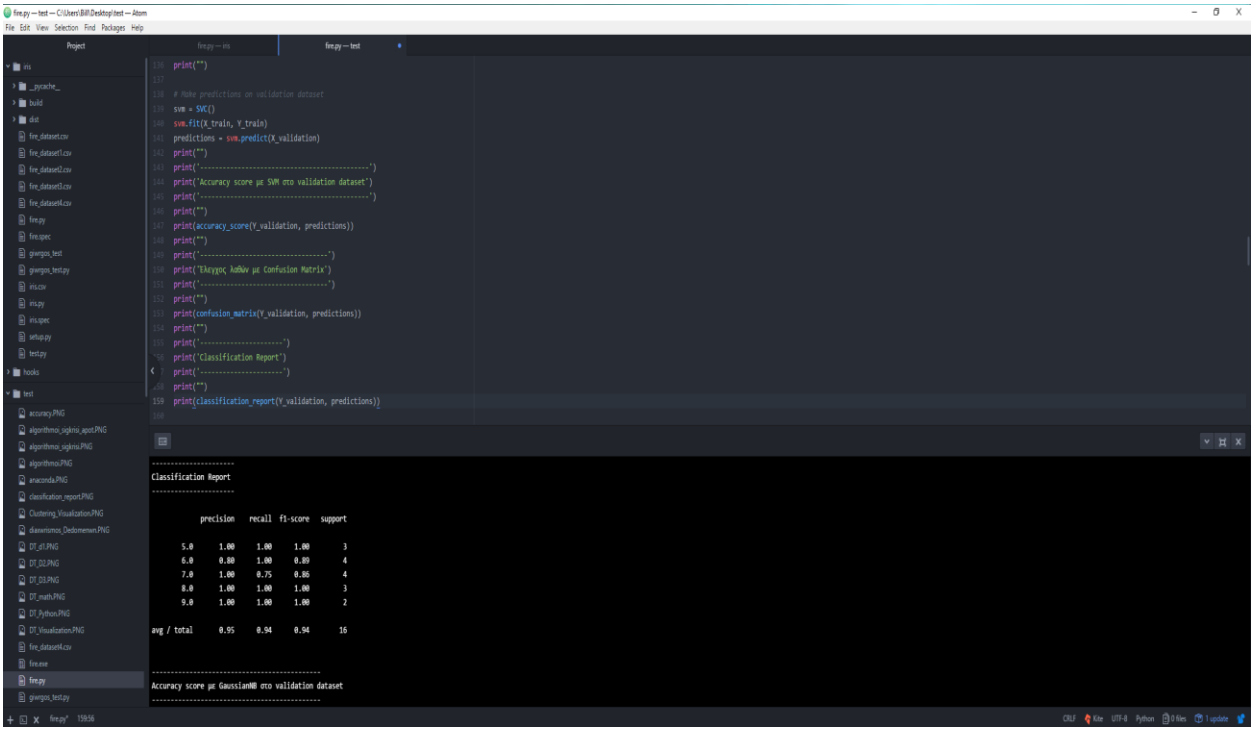
1) Το syntax highlighting στον editor
2) Ο File Explorer για την άμεση εναλλαγή μεταξύ working directories
3) Η πληθώρα υποστήριξης κονσόλων και κυρίως της iPython, απαραίτητη για το visualization των αποτελεσμάτων
4) Το παράθυρο βοήθειας με το πλούσιο documentation για όλα τα εγκατεστημένα plugins αλλά και για αυτά που θα περάσει ο χρήστης
5) Ο Variable Explorer που εμφανίζει τα dataframes που δημιουργήθηκαν



Εικόνα 5.22: Το περιβάλλον εργασίας του Spyder

5.4.3 Atom

Το τελευταίο εργαλείο που χρησιμοποίησα είναι ο text editor Atom, με τον οποίο διαμόρφωσα καλύτερα τον κώδικα μου, έκανες τις απαραίτητες αλλαγές όπου αυτό ήταν θεμιτό και compile προκειμένου να δημιουργηθεί το executable αρχείο. Ένας από τους καλύτερους editors για τη γλώσσα Python, “hackable” όπως συνηθίζουν να τον αποκαλούν λόγω της ατέλειωτης παραμετροποίησης που προσφέρει στον χρήστη, από την εγκατάσταση νέων packages και themes έως το πλήρες customization του UI με πείραγμα της HTML και της JavaScript του προγράμματος.



Εικόνα 5.23: Το περιβάλλον εργασίας του Atom

Κεφάλαιο 6

Συμπεράσματα και Μελλοντική Επέκταση

6.1 Συμπεράσματα

Συμπερασματικά καθημερινά παράγεται ένα τεράστιο πλήθος δεδομένων από τα οποία ένα μικρό ποσοστό μπορεί να αξιοποιηθεί προκειμένου να βοηθήσει στην επίλυση μικρών ή και μεγαλύτερων προβλημάτων.

Στην παρούσα διπλωματική εργασία γίνεται προσπάθεια εκμετάλλευσης περιβαλλοντικών μετρήσεων αλλά η εκμετάλλευση της τεχνολογίας της μηχανικής μάθησης προκειμένου να μπορεί να κάνει προβλέψεις για την πρόβλεψη και την πρόληψη φωτιάς. Μία εφαρμογή σαν αυτή που υλοποίησα θα μπορεί να δέχεται δεδομένα από συγκεκριμένες περιοχές, να διακρίνει τη θερμοκρασία, την υγρασία, τα επίπεδα οξυγόνου και διοξειδίου του άνθρακα και να μπορεί να προβλέπει τον κίνδυνο να ξεσπάσει μια πυρκαγιά.

Στην εφαρμογή υπάρχει ένα σύνολο έξι αλγορίθμων για τους οποίους γίνονται συγκριτικά τεστ προκειμένου να βρεθεί ο ιδανικός σύμφωνα πάντα με τα δεδομένα, το είδος αλλά και το πλήθος πριν το πρόγραμμα προβεί στο απαραίτητο classification.

6.2 Μελλοντική Επέκταση

Η εφαρμογή στην παρούσα φάση είναι σε αρκετά πρώιμο στάδιο. Το πρώτο βήμα για την εξέλιξη της είναι να αποκτήσει ένα γραφικό περιβάλλον για να γίνει πιο φιλική προς τον χρήστη, μιας και αυτή τη στιγμή ανοίγει και τρέχει σε command prompt. Επίσης εμφανίζει όλα τα αποτελέσματα καθώς αυτά τρέχουν από τον κώδικα όπου σε αυτό το σημείο και με την υλοποίηση ενός UI θα ήθελα να προσθέσω κουμπιά και τη δυνατότητα στον χρήστη να εμφανίζει ότι επιθυμεί αυτός, όποτε το επιθυμεί και για όσες φορές χρειαστεί.

Σε δεύτερο στάδιο και μετά την υλοποίηση του γραφικού περιβάλλοντος οι σκέψεις μου είναι να αποκτήσει υποστήριξη για Mac OS και Linux, καθώς τη δεδομένη χρονική στιγμή το μόνο λειτουργικό σύστημα που υποστηρίζει είναι τα Windows. Ένα βήμα ακόμα είναι και η υποστήριξη από συσκευές κινητών και η μετατροπή της σε mobile application για υποστήριξη iOS και Android.

Το τελικό στάδιο είναι η δημιουργία μίας συσκευής για την ενσωμάτωση της εφαρμογής μου. Με την επίτευξη αυτού του έργου το αμέσως επόμενο σχέδιο είναι η προσπάθεια εξέλιξης τόσο της συσκευής αλλά και της εφαρμογής. Η συσκευή θα πρέπει να είναι φορητή συνεπώς θα πρέπει να λειτουργεί μέσω μπαταρίας αλλά και να έχει σύνδεση στο ίντερνετ προκειμένου να τραβάει τα δεδομένα από κάποιον server ή βάση δεδομένων. Αντίστοιχα η εφαρμογή θα πρέπει να βελτιωθεί για να μην απαιτεί μεγάλη υπολογιστική ισχύ κάτι που θα είχε επίπτωση στη μπαταρία της συσκευής και συνεπώς στη διάρκεια λειτουργίας

Σύμφωνα με όλα τα παραπάνω θα ικανοποιούσα το στόχο της δημιουργίας μιας mobile συσκευής πρόβλεψης κινδύνου πυρκαγιάς η οποία θα μπορούσε να αποτελέσει μέρος του εξοπλισμού του πυροσβεστικού σώματος για πρόβλεψη και άμεση αντιμετώπιση εάν και όταν αυτό κριθεί επιθυμητό.

Βιβλιογραφία

[1] Weather Elements that affect Fire Behavior.(n.d). Διαθέσιμο στο http://www.auburn.edu/academic/forestry_wildlife/fire/weather_elements.html. Ανακτήθηκε 6 Ιουνίου 2018

[2] Fire Fundamentals: Fire Weather .(n.d). Διαθέσιμο στο http://learnline.cdu.edu.au/units/env207/fundamentals/weather.html. Ανακτήθηκε 5 Ιουνίου 2018

[3] THE DIGITAL UNIVERSE IN 2020 :Big Data , Bigger Digital Shadows , and Biggest Growth in the Far East—United States .(2013). Διαθέσιμο στο https://www.emc.com/collateral/analyst-reports/idc-digital-universe-united-states.pdf. Ανακτήθηκε 2 Ιουνίου 2018

[4] Hurwitz, J. Kirsch, Daniel(2018), Machine Learning for dummies, Hoboken NJ: John Wiley & Sons Inc

[5] Mitchell, T.(1997), Machine Learning, McGraw

[6] THE DIGITAL UNIVERSE IN 2020.(2012). Διαθέσιμο στο https://www.emc.com/collateral/analyst-reports/idc-the-digital-universe-in-2020.pdf. Ανακτήθηκε 11 Μαΐου 2018

[7] Swaminathan, S. Logistic Regression — Detailed Overview.(2015). Διαθέσιμο στο https://towardsdatascience.com/logistic-regression-detailed-overview-46c4da4303bc . Ανακτήθηκε 5 Μαΐου 2018

[8]Jason Brownlee, Machine Learning Algorithms in Python(n.d). Machine Learning Mastery, Διαθέσιμο στο https://machinelearningmastery.com/machine-learning-with-python/, Ανακτήθηκε 15 Ιουνίου 2018

[9] Linear and Quadratic Discriminant Analysis.(n.d). Διαθέσιμο στο http://scikit-learn.org/stable/modules/lda_qda.html . Ανακτήθηκε 8 Μαΐου 2018

[10] A Complete Guide to K-Nearest-Neighbors with Applications in Python and R.(2016). Διαθέσιμο στο https://kevinzakka.github.io/2016/07/13/k-nearest-neighbor/ . Ανακτήθηκε 2 Μαΐου 2018

[11] Blood Transfusion Service Center Data Set(n.d). Διαθέσιμο στο https://archive.ics.uci.edu/ml/datasets/Blood+Transfusion+Service+Center . Ανακτήθηκε 9 Ιουνίου 2018

[12] Iris Data Set (n.d). Διαθέσιμο στο https://archive.ics.uci.edu/ml/datasets/Iris. Ανακτήθηκε 9 Ιουνίου 2018

[13] Lenses Data Set (n.d). Διαθέσιμο στο https://archive.ics.uci.edu/ml/datasets/Lenses. Ανακτήθηκε 9 Ιουνίου 2018

[14] Decision Trees in Machine Learning (2017). Διαθέσιμο στο https://towardsdatascience.com/decision-trees-in-machine-learning-641b9c4e8052. Ανακτήθηκε 20 Ιουνίου 2018