ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ
Πρόγραμμα Μεταπτυχιακών Σπουδών
“Μηχανική Λογισμικού για Διαδικτυακές & Φορητές Εφαρμογές”

Τίτλος Διπλωματικής

Ενσωμάτωση και χρήση Μηχανικής Μάθησης σε σύστημα διαχείρισης Έξυπνου Σπιτιού

Βαρσάμης Ζοζέφ

Επιβλέπων: *<Σάββας Ηλίας, Καθηγητής>*

ΛΑΡΙΣΑ 2021

«Εγώ ο <Βαρσάμης Ζοζέφ>, δηλώνω υπεύθυνα ότι η παρούσα Πτυχιακή Εργασία με τίτλο < ***Ενσωμάτωση και χρήση Μηχανικής Μάθησης σε σύστημα διαχείρισης Έξυπνου Σπιτιού****> είναι δική μου και βεβαιώνω ότι:*

- *Σε όσες περιπτώσεις έχω συμβουλευτεί δημοσιευμένη εργασία τρίτων, αυτό επισημαίνεται με σχετική αναφορά στα επίμαχα σημεία.*
- *Σε όσες περιπτώσεις μεταφέρω λόγια τρίτων, αυτό επισημαίνεται με σχετική αναφορά στα επίμαχα σημεία. Με εξαίρεση τέτοιες περιπτώσεις, το υπόλοιπο κείμενο της πτυχιακής αποτελεί δική μου δουλειά.*
- *Αναφέρω ρητά όλες τις πηγές βοήθειας που χρησιμοποίησα.*
- *Σε περιπτώσεις που τμήματα της παρούσας πτυχιακής έγιναν από κοινού με τρίτους, αναφέρω ρητά ποια είναι η δική μου συνεισφορά και ποια των τρίτων.*
- *Γνωρίζω πως η λογοκλοπή αποτελεί σοβαρότατο παράπτωμα και είμαι ενήμερος(-η) για την επέλευση των νομίμων συνεπειών»*

< υπογραφή >

< Βαρσάμης Ζοζέφ>

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή

Τόπος: .....

Ημερομηνία: .....

ΕΠΙΤΡΟΠΗ ΑΞΙΟΛΟΓΗΣΗΣ

1. .....
2. .....
3. .....

Περίληψη

Ενσωμάτωση και χρήση Μηχανικής Μάθησης με σκοπό την βελτίωση του τρόπου λειτουργίας του συγκεκριμένου προ υπάρχοντος συστήματος διαχείρισης σπιτιού. Σκοπός μας είναι η βελτίωση της λειτουργίας του προγράμματος τόσο σε θέματα αυτοματοποίησης και λειτουργικότητας όσο και σε βελτιστοποίηση της ταχύτητας εκτέλεσης του. Το υπάρχον σύστημα, όσον αφορά το Hardware, αποτελείται από ένα 'Raspberry Pi model 3', ένα 'Grove Shield' μαζί με κάποιους αισθητήρες και μία IR-Camera. Όσον αφορά το Software, το πρόγραμμα είναι σε γλώσσα προγραμματισμού 'Python 2.7', η ιστοσελίδα διαχείρισης 'τρέχει' σε 'local web-server' και συγκεκριμένα στον 'Apache2' και έχουν χρησιμοποιηθεί οι γλώσσες προγραμματισμού 'PHP', 'HTML' και 'JavaScript'.

Επίσης έχουν χρησιμοποιηθεί μερικά API's, όπως 'DarkSky', 'ThingSpeak' και 'Zapier', τόσο για απεικόνιση των δεδομένων που λαμβάνουμε από τους αισθητήρες όσο και για την αυτοματοποίηση κάποιων ενεργειών. Το πεδίο με το οποίο θα ασχοληθούμε, είναι αυτό της επεξεργασίας των δεδομένων που λαμβάνουμε από τους αισθητήρες. Είναι σημαντικό να αναφερθεί πως για να μπορέσουμε να κάνουμε χρήση Μηχανικής Μάθησης, θα πρέπει να έχουμε μεγάλο όγκο δεδομένων προς επεξεργασία. Στο συγκεκριμένο σύστημα υπάρχουν δεδομένα καταγραφής από τους αισθητήρες, από τον Οκτώβριο του 2018. Ειδικότερα, θα ασχοληθούμε μόνο με το θέμα της Θέρμανσης/Ψύξης(Καλοριφέρ/Air-Condition), με σκοπό την εξάλειψη του περιθωρίου λάθους του συστήματος. Δηλαδή, με βάση τα δεδομένα των αισθητήρων και τις ενέργειες του χρήστη, να ελέγξουμε σε ποιες περιπτώσεις το σύστημα 'έσφαλε' και ενεργοποίησε ή απενεργοποίησε την Θέρμανση/Ψύξη και ο χρήστης το 'διόρθωσε'. Οι τεχνικές Μηχανικής Μάθησης που θα χρησιμοποιήσουμε είναι: Regression και Classification. Επίσης χρησιμοποιήθηκε και το Rapid Miner Studio για την απεικόνιση των μοντέλων (Decision Tree & Linear Regression).

Ευχαριστίες

Ευχαριστώ τους γονείς μου και τους καθηγητές μου που με βοήθησαν να καταλάβω πόσο σημαντικό ήταν να συνεχίσω τις σπουδές μου και σε Μεταπτυχιακό επίπεδο και ήταν αρωγοί στην προσπάθεια μου μέχρι το τέλος.

Βαρσάμης Ζοζέφ

ημερομηνία

Περιεχόμενα

1 Εισαγωγή

Τον Οκτώβριο του 2018, παραδόθηκε και παρουσιάστηκε πτυχιακή εργασία με τίτλο: «Ανάπτυξη εφαρμογής «Εξυπνου» Σπιτιού με χρήση Raspberry Pi, Grove Pi και την τεχνολογία 'Internet of Things'» από τον φοιτητή Βαρσάμη Ζοζέφ, δηλαδή εμού του ιδίου. Υπάρχουν αναφορές σε καίρια σημεία της πτυχιακής εργασίας, οι οποίες είναι απαραίτητες τόσο για την κατανόηση του τρόπου λειτουργίας του προ υπάρχοντος συστήματος, όσο και για τον τρόπο με τον οποίο θα γίνει η αναβάθμισή του συστήματος ώστε να χρησιμοποιηθεί η Μηχανική Μάθηση. Τα 'νέα' κεφάλαια στα οποία αναφέρεται και περιγράφεται αναλυτικά η ανάπτυξη και χρήση της Μηχανικής Μάθησης στο σύστημα είναι τα Μηχανική Μάθηση (Machine Learning – ML) και Ανάπτυξη συστήματος Μηχανικής Μάθησης (Machine Learning)

1.1 Μηχανική Μάθηση (Machine Learning – ML)

1.1.1 Ορισμός

Η Μάθηση (Learning) είναι μία από τις θεμελιώδεις ιδιότητες της νοήμονος συμπεριφοράς του ανθρώπου. Παρά τις μελέτες και τις έρευνες επί χρόνια από τους επιστήμονες του πεδίου της Γνωστικής Ψυχολογίας και τους φιλοσόφους, η έννοια της μάθησης δεν έχει γίνει πλήρως κατανοητή. Πώς, λοιπόν, θα μπορούσαν οι επιστήμονες του χώρου της ΤΝ να δημιουργήσουν υπολογιστικά συστήματα ικανά να μάθουν, να επιτύχουν, δηλαδή, τη λεγόμενη Μηχανική Μάθηση (Machine Learning).

Αυτή μπορεί να οριστεί ως:

«το φαινόμενο κατά το οποίο ένα σύστημα βελτιώνει την απόδοσή του κατά την εκτέλεση μιας συγκεκριμένης εργασίας, χωρίς να υπάρχει ανάγκη να προγραμματιστεί εκ νέου.»

Βάσει του ορισμού αυτού, η Μηχανική Μάθηση έχει ως σκοπό τη δημιουργία μηχανών ικανών να μαθαίνουν, να βελτιώνουν, δηλαδή, την απόδοσή τους σε κάποιους τομείς μέσω της αξιοποίησης προηγούμενης γνώσης και εμπειρίας. Ένας σχετικός γενικός ορισμός Μηχανικής Μάθησης δίνεται από τον Mitchell (1997):

«Ένα πρόγραμμα υπολογιστή λέμε ότι μαθαίνει από την εμπειρία Ε ως προς κάποια κλάση εργασιών Τ και μέτρο απόδοσης Ρ, αν η απόδοσή του σε εργασίες από το Τ, όπως μετριέται από το Ρ, βελτιώνεται μέσω της εμπειρίας Ε.»

Στην Επαγωγική Μάθηση (Inductive Learning), με τη διαδικασία της επαγωγής (induction) ο άνθρωπος μαθαίνει κατανοώντας το περιβάλλον του μέσω παρατηρήσεων και δημιουργεί μια απλοποιημένη (αφαιρετική) εκδοχή του που ονομάζεται νοητικό μοντέλο (mental model). Επιπλέον, ο άνθρωπος έχει τη δυνατότητα να οργανώνει και να συσχετίζει τις εμπειρίες και τις παρατηρήσεις του δημιουργώντας νέες δομές που ονομάζονται νοητικά πρότυπα (mental patterns), με αξιοποίηση και του επαγωγικού και του απαγωγικού συλλογισμού. Στη δημιουργία νέων προτύπων από παλαιά βασίζονται οι τρόποι μάθησης που εξαρτώνται σε μεγαλύτερο ή μικρότερο βαθμό από την προ υπάρχουσα γνώση για ένα πρόβλημα, όπως είναι η μάθηση από επεξηγήσεις και η μάθηση από περιπτώσεις.

Σε σχέση με την ανθρώπινη ικανότητα προς μάθηση, οι φιλόσοφοι θέτουν το ερώτημα: *«Πώς μπορεί ένας επαγωγικός συλλογισμός που οδηγεί στη μάθηση να αξιολογηθεί ως προς την ορθότητά του;».* Αντίστοιχα, οι ψυχολόγοι ρωτούν: *«Πώς αποθηκεύει ο εγκέφαλος τα αποτελέσματα της διαδικασίας της μάθησης, δηλαδή τα νοητικά μοντέλα και τα πρότυπα;».* Στο χώρο της ΤΝ απλώς ρωτούν: «*Πώς μπορεί μία μηχανή να δημιουργήσει νέα μοντέλα και πρότυπα μάθησης από συγκεκριμένα παραδείγματα και πόσο αξιόπιστα είναι αυτά τα μοντέλα και πρότυπα στην πράξη;».*

Με βάση τα παραπάνω, μπορεί να δοθεί ο ακόλουθος εναλλακτικός ορισμός για τη Μηχανική Μάθηση:

Μηχανική Μάθηση ονομάζεται η ικανότητα ενός υπολογιστικού συστήματος να δημιουργεί μοντέλα ή πρότυπα από ένα σύνολο δεδομένων.

Ως κλάδος της ΤΝ, η Μηχανική Μάθηση ασχολείται με τη μελέτη αλγορίθμων που βελτιώνουν τη συμπεριφορά τους σε κάποια εργασία που τους έχει ανατεθεί χρησιμοποιώντας την εμπειρία τους.

Όσον αφορά τη σχεδίαση των συστημάτων Μηχανικής Μάθησης, για τα συστήματα που ανήκουν στη συμβολική ΤΝ, η δυνατότητα μάθησης προσδιορίζεται ως η ικανότητα πρόσκτησης επιπλέον γνώσης, που επιφέρει μεταβολές στην υπάρχουσα καταχωρημένη γνώση είτε αλλάζοντας χαρακτηριστικά της είτε με αυξομείωσή της. Στην περίπτωση των συστημάτων ΤΝ που ανήκουν στη Μη Συμβολική ΤΝ (όπως η περίπτωση των Τεχνητών Νευρωνικών Δικτύων), ως μάθηση προσδιορίζεται η

δυνατότητα που διαθέτουν τα συστήματα στο να μετασχηματίζουν την εσωτερική τους δομή, παρά στο να μεταβάλλουν κατάλληλα τη γνώση που έχει καταχωρηθεί μέσα σε αυτά κατά το σχεδιασμό τους.

Αν και απέχουμε πάρα πολύ από τη δημιουργία μηχανών που μαθαίνουν τόσο καλά όσο ο άνθρωπος, για συγκεκριμένες περιοχές μάθησης έχουν αναπτυχθεί αλγόριθμοι οι οποίοι έχουν επιτρέψει την εμφάνιση σύγχρονων εμπορικών εφαρμογών με σημαντική επιτυχία. Επιπλέον, τα αποτελέσματα από τις εφαρμογές της ΤΝ αρχίζουν ήδη να είναι ορατά και να δίνουν απαντήσεις σε αναπάντητα, έως τώρα, ερωτήματα των άλλων κλάδων που διερευνούν την ικανότητα του ανθρώπου να μαθαίνει.

Ο τομέας της Μηχανικής Μάθησης αναπτύσσει, επίσης, επιτυχώς την Εξελικτική Μάθηση (Evolutionary Learning), η οποία μιμείται διαδικασίες φυσικής αναπαραγωγικής σε έμβια όντα. Χρησιμοποιείται κυρίως σε προβλήματα βελτιστοποίησης. Στην Εξελικτική Μάθηση κυριαρχούν οι γενετικοί αλγόριθμοι που θα παρουσιαστούν στο τέλος του κεφαλαίου.

Εκτός της ίδιας της ΤΝ, μεταξύ των επιστημονικών κλάδων που επωφελούνται από τα επιτεύγματα στον τομέα της Μηχανικής Μάθησης συγκαταλέγονται οι: Εξόρυξη Δεδομένων, Πιθανότητες και Στατιστική, Θεωρία της Πληροφορίας, Αριθμητική Βελτιστοποίηση, Θεωρία της Πολυπλοκότητας, Θεωρία Ελέγχου (προσαρμοστική), Ψυχολογία (εξελικτική, γνωστική), Νευροβιολογία και Γλωσσολογία.

1.1.2 Εργαλεία

Εν γένει, ο τομέας της Μηχανικής Μάθησης αναπτύσσει τρεις τρόπους μάθησης, ανάλογους με τους τρόπους με τους οποίους μαθαίνει ο άνθρωπος: επιβλεπόμενη μάθηση, μη επιβλεπόμενη μάθηση και ενισχυτική μάθηση. Πιο αναλυτικά:

- Επιβλεπόμενη Μάθηση (Supervised Learning) είναι η διαδικασία όπου ο αλγόριθμος κατασκευάζει μια συνάρτηση που απεικονίζει δεδομένες εισόδους (σύνολο εκπαίδευσης) σε γνωστές επιθυμητές εξόδους, με απώτερο στόχο τη γενίκευση της συνάρτησης αυτής και για εισόδους με άγνωστη έξοδο. Χρησιμοποιείται σε προβλήματα:
    - Ταξινόμησης (Classification)
    - Πρόγνωσης (Prediction)
    - Διερμηνείας (Interpretation)

- Μη Επιβλεπόμενη Μάθηση (Unsupervised Learning), όπου ο αλγόριθμος κατασκευάζει ένα μοντέλο για κάποιο σύνολο εισόδων υπό μορφή παρατηρήσεων χωρίς να γνωρίζει τις επιθυμητές εξόδους. Χρησιμοποιείται σε προβλήματα:
    - Ανάλυσης Συσχετισμών (Association Analysis)
    - Ομαδοποίησης (Clustering)
- Ενισχυτική Μάθηση (Reinforcement Learning), όπου ο αλγόριθμος μαθαίνει μια στρατηγική ενεργειών μέσα από άμεση αλληλεπίδραση με το περιβάλλον. Χρησιμοποιείται κυρίως σε προβλήματα Σχεδιασμού (Planning), όπως για παράδειγμα ο έλεγχος κίνησης ρομπότ και η βελτιστοποίηση εργασιών σε εργοστασιακούς χώρους.

Για κάθε πρόβλημα προς επίλυση στο χώρο της Μηχανικής Μάθησης υπάρχει ένας κατάλληλος τρόπος μάθησης και για κάθε τρόπο μάθησης υπάρχει τουλάχιστον ένας κατάλληλος αλγόριθμος που μπορεί να χρησιμοποιηθεί.

Όσον αφορά τη δική μας περίπτωση, θα χρησιμοποιήσουμε την Επιβλεπόμενη Μάθηση και πιο συγκεκριμένα θα χρησιμοποιήσουμε τεχνικές Regression και Classification.

1.1.3 Διαδικασία Επίλυσης

Για να περάσουμε στη διαδικασία της επίλυσης, δηλαδή ανάπτυξης του μοντέλου Μηχανικής Μάθησης, πρέπει να προϋπάρχουν τα εξής: ικανοποιητικός αριθμός δεδομένων προς επεξεργασία, ένας αλγόριθμος ο οποίος δέχεται αυτά τα δεδομένα ως είσοδο και βάσει αυτών παράγει ένα αποτέλεσμα, καταγραφή όλων αυτών των αποτελεσμάτων με σκοπό της την λήψη αποφάσεων αν είναι ορθά ή μη. Πιο συγκεκριμένα, τα δεδομένα μας προέρχονται από αισθητήρες και αφορούν θέρμανση και υγρασία περιβάλλοντος τόσο σε εσωτερικό χώρο όσο και στον εξωτερικό. Ο αλγόριθμος λειτουργεί συναντήσει κάποιας σταθεράς (**επιθυμητή** εσωτερική θερμοκρασία και υγρασία), των μεταβλητών τιμών της εσωτερικής θερμοκρασίας/υγρασίας και εξωτερικής αντίστοιχα και παράγει ένα αποτέλεσμα το οποίο και διαχειρίζεται την θέρμανση και ψύξη του συστήματος.

Για παράδειγμα, έστω πως η επιθυμητή θερμοκρασία και υγρασία θέλουμε να είναι στους 21 °C και έως 50% αντίστοιχα. Εάν υποθέσουμε πως η εσωτερική θερμοκρασία είναι στους 19,5 °C και 37% αντίστοιχα και η εξωτερική στους 10 °C και 70%, αφού το σύστημα επεξεργαστεί αυτά τα δεδομένα θα καταλήξει στο αν πρέπει ή όχι να ενεργοποιηθεί η ψύξη ή η θέρμανση αντίστοιχα. Επίσης, με στόχο την βελτιστοποίηση

του συστήματος, θα πρέπει να λάβουμε υπόψη και το πόσο χρόνο θα χρειαστεί το σύστημα για να φτάσει στην επιθυμητή κατάσταση, ανάλογα κάθε φορά με τα δεδομένα που έχουμε. Έστω ο παρακάτω υποθετικός πίνακας δεδομένων:

User Settings		**Indoor Data**		**Outdoor Data**		**System**
temperature	humidity	temperature	humidity	temperature	humidity	activate
21	50	19,5	37	10	70	1
21	50	21	45	15	40	0
21	50	20	40	13	55	1

Το ερώτημα που προκύπτει είναι, «*πώς μπορούμε να υπολογίσουμε τον χρόνο που θα χρειαστεί να λειτουργήσει το σύστημα για να φτάσει στα επιθυμητά όρια πριν αυτό λειτουργήσει;».* Σ' αυτό το σημείο θα εφαρμοστεί η Μηχανική Μάθηση, για να προβλέψουμε τόσο τον ακριβή χρόνο λειτουργίας του συστήματος όσο και την αξιολόγησή του, δηλαδή αν λειτούργησε ορθά ή όχι.

Δυαδική ταξινόμηση (Binary Classification)

Η δυαδική ταξινόμηση αναφέρεται σε εκείνες τις εργασίες ταξινόμησης που έχουν δύο ετικέτες τάξης, συνήθως περιγράφονται με 0 ή 1 (Boolean). Τα παραδείγματα περιλαμβάνουν:

- Ανίχνευση ανεπιθύμητων μηνυμάτων ηλεκτρονικού ταχυδρομείου (ανεπιθύμητο ή μη).
- Πρόβλεψη για την καύση (churn or not).
- Πρόβλεψη μετατροπής (αγορά ή όχι).

Συνήθως, οι εργασίες δυαδικής ταξινόμησης περιλαμβάνουν μια κλάση που είναι η κανονική κατάσταση και μια άλλη κατηγορία που είναι η μη φυσιολογική κατάσταση.

Για παράδειγμα, το "not spam" είναι η κανονική κατάσταση και το "spam" είναι η μη φυσιολογική κατάσταση. Ένα άλλο παράδειγμα είναι ότι «ο καρκίνος δεν ανιχνεύεται»

είναι η φυσιολογική κατάσταση μιας εργασίας που περιλαμβάνει ιατρικό τεστ και ο «καρκίνος που ανιχνεύεται» είναι η ανώμαλη κατάσταση.

Στην κλάση για την κανονική κατάσταση εκχωρείται η ετικέτα κλάσης 0 και στην τάξη με την ανώμαλη κατάσταση εκχωρείται η ετικέτα κλάσης 1.

Είναι σύνηθες να μοντελοποιούμε μια εργασία δυαδικής ταξινόμησης με ένα μοντέλο που προβλέπει μια κατανομή πιθανότητας Bernoulli για κάθε παράδειγμα.

Η κατανομή Bernoulli είναι μια διακριτή κατανομή πιθανότητας που καλύπτει μια περίπτωση όπου ένα συμβάν θα έχει δυαδικό αποτέλεσμα είτε ως 0 ή 1. Για την ταξινόμηση, αυτό σημαίνει ότι το μοντέλο προβλέπει την πιθανότητα ενός παραδείγματος που ανήκει στην κλάση 1 ή την ανώμαλη κατάσταση .

Οι δημοφιλείς αλγόριθμοι που μπορούν να χρησιμοποιηθούν για δυαδική ταξινόμηση περιλαμβάνουν:

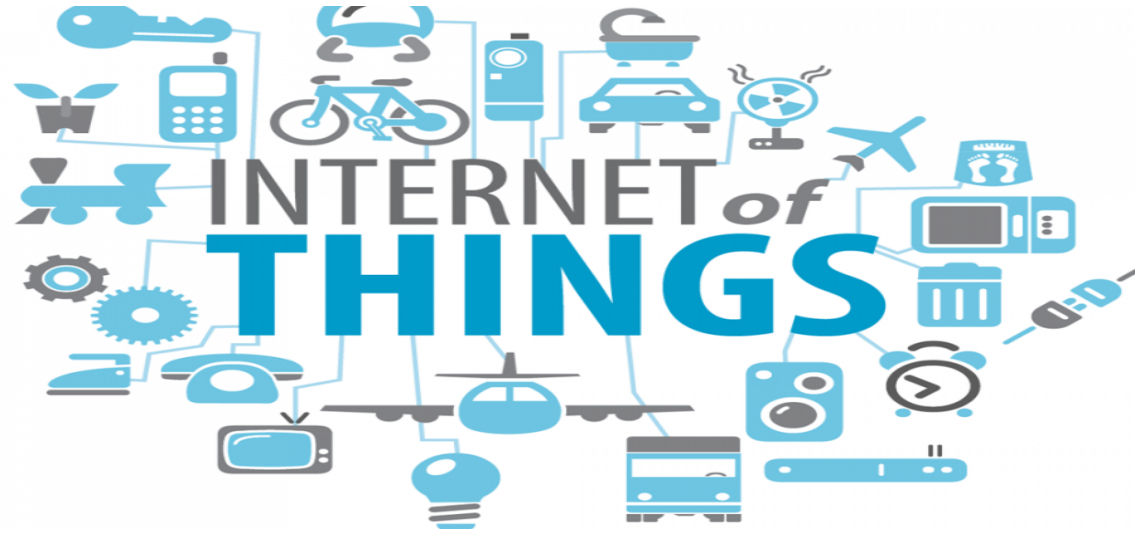
- Logistic Regression
- k-Nearest Neighbors
- Decision Trees
- Support Vector Machine
- Naive Bayes

Ορισμένοι αλγόριθμοι έχουν σχεδιαστεί ειδικά για δυαδική ταξινόμηση και δεν υποστηρίζουν εγγενώς περισσότερες από δύο κατηγορίες. Παραδείγματα περιλαμβάνουν Logistics Regression και Support Vector Machines.

1.1.4 Προβλήματα

Το κυρίως πρόβλημα που έχουμε να αντιμετωπίσουμε είναι, η αξιοπιστία των δεδομένων που έχουμε στην κατοχή μας, ο όγκος αυτών των δεδομένων καθώς και το είδος τους (celsius or fahrenheit). Επίσης θα πρέπει να τα φέρουμε σε τέτοια μορφή η οποία θα επιτρέπει την επεξεργασία τους από αλγορίθμους της Μηχανικής Μάθησης. Για την μετατροπή των δεδομένων ώστε να είναι σε μορφή κατανοητή από τους αλγορίθμους της Μηχανικής Μάθησης, θα χρησιμοποιήσουμε το Spark, εργαλείο διαχείρισης και επεξεργασίας Μεγάλων Δεδομένων (Big Data), Stream Analytics κλπ.

1.2 Ορισμός του IoT – Internet of Things



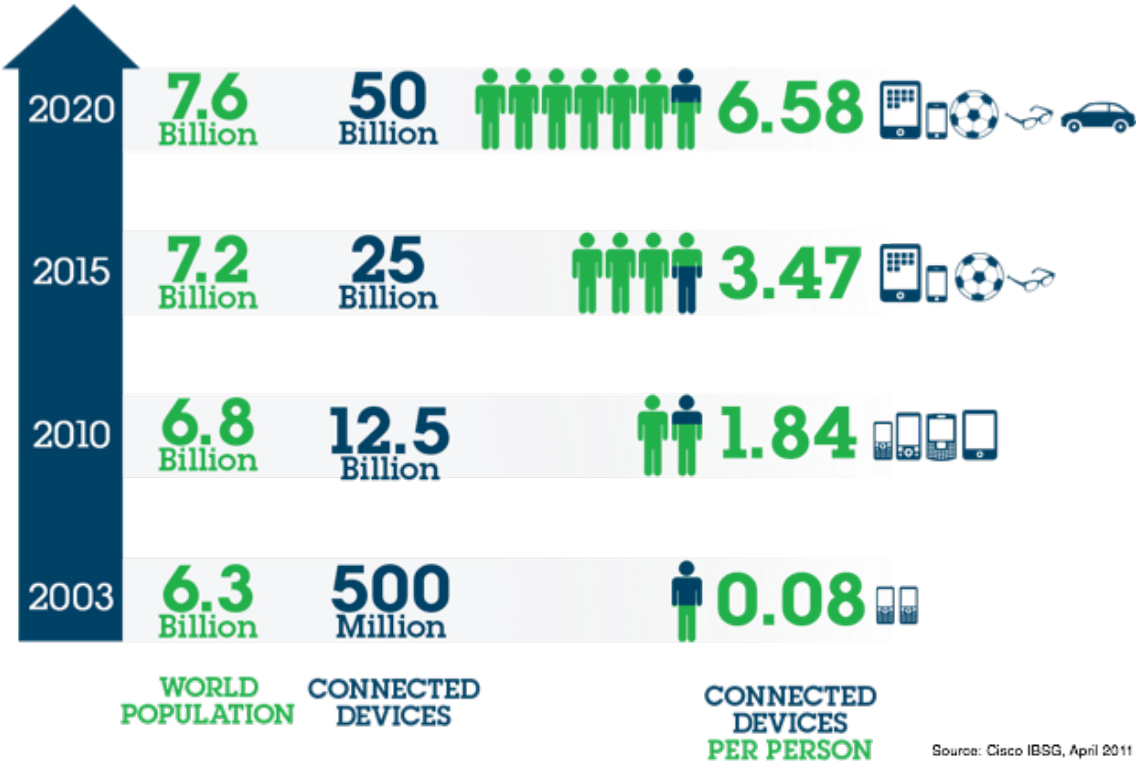
Εικόνα 1. Internet of Things Logo

Η ανακάλυψη του διαδικτύου (internet) έκανε την μεγαλύτερη αλλαγή στον κόσμο μας όπως τον ξέρουμε μέχρι σήμερα. Είναι αν όχι το μεγαλύτερο, τότε σίγουρα ένα από τα μεγαλύτερα επιτεύγματα του σύγχρονου πολιτισμού μας. Ωστόσο, το επόμενο 'μεγάλο πράγμα' ή αλλιώς 'the next big thing' όπως συνηθίζεται να λέμε, είναι μία τεχνολογία η οποία προήλθε από το 'internet' και έχει άμεση σχέση με αυτό. Αναφερόμαστε στην τεχνολογία 'διαδίκτυο των πραγμάτων' (IoT – Internet of Things). Την ιδέα την είχε ο 'Kevin Ashton', ένας Βρετανός πρωτοπόρος ηλεκτρονικός ο οποίος είναι υπεύθυνος και για την ανακάλυψη της 'ταυτοποίησης μέσω ραδιοσυχνοτήτων' (RFID), τεχνολογία η οποία είναι ευρέως διαδεδομένη σήμερα με τον όρο 'Barcode'.

Η βασική έννοια του 'IoT', είναι ότι όλες οι συσκευές πέρα από το κινητό μας, το tablet και τον υπολογιστή μας, θα μπορούν να έχουν πρόσβαση στο διαδίκτυο, θα έχουν αισθητήρες και θα συλλέγουν δεδομένα και θα μπορούμε να τις ελέγχουμε εξ' αποστάσεως. Ένα από τα πιο χαρακτηριστικά παραδείγματα χρήσης αυτής της τεχνολογίας, είναι η δυνατότητα δημιουργίας ενός 'έξυπνου σπιτιού' (Smart Home) κατά την οποία το σπίτι αποτελείται από συσκευές οι οποίες έχουν πρόσβαση στο διαδίκτυο και μπορούν να λειτουργούν αυτόνομα και αυτόματα. Δηλαδή, θα μπορούσε για παράδειγμα να υπάρχει 'έξυπνη συσκευή' αντί για τον κλασικό θερμοστάτη για τον έλεγχο της θέρμανσης στο σπίτι, η οποία θα μας έδινε την δυνατότητα πέρα από το να ελέγχουμε τη θερμοκρασία στο σπίτι, να μπορούμε να ενεργοποιούμε ή να

απενεργοποιούμε την θέρμανση εξ' αποστάσεως με μία απλή κίνηση.

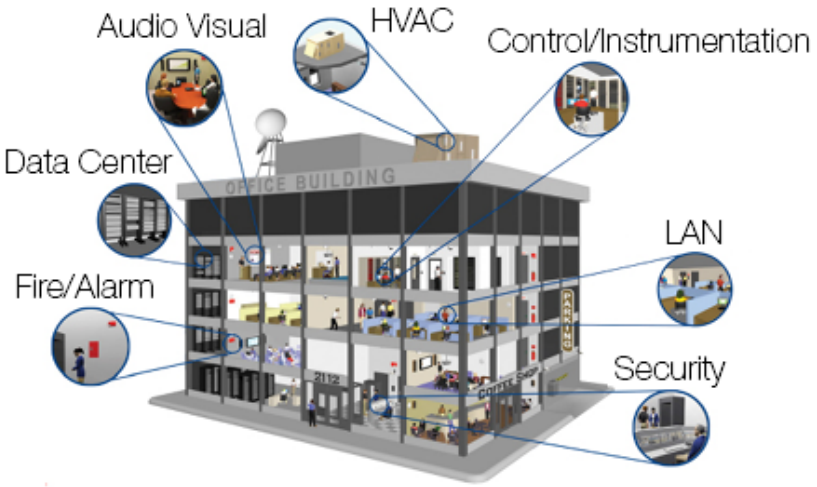
Σκοπός της τεχνολογίας αυτής, είναι να κάνει τη ζωή μας πιο εύκολη αλλά και πιο αυτοματοποιημένη. Πιο συγκεκριμένα, ο όρος ή έννοια 'διαδίκτυο των πραγμάτων' οφείλεται στο γεγονός ό,τι η κάθε συσκευή έχει ενσωματωμένους αισθητήρες πάνω της, λογισμικό για την λειτουργεία της και σύνδεση στο διαδίκτυο τόσο για την αλληλεπίδραση με τις άλλες συσκευές όσο και για διαχείριση της εξ' αποστάσεως. Πλέον όλες οι μεγάλες επιχειρήσεις τείνουν να χρησιμοποιούν όλο και περισσότερο αυτή την τεχνολογία γιατί τους παρέχει τόσο ευκολία στη διαχείριση δεδομένων και λειτουργειών, όσο και ασφάλεια στην εκτέλεση και λειτουργεία αυτών. Ουσιαστικά, η ζωή μας όσον αφορά τη διαχείριση και λειτουργεία των συσκευών μας οικιακών και μη, δεν θα 'ναι ποτέ η ίδια. Ενδεικτικά, υπολογίζεται πως μέχρι το τέλος του 2020, πάνω από δύο δισεκατομμύρια συσκευές θα αποτελούν μέρος του 'δικτύου έξυπνων συσκευών' που θα έχει αναπτυχθεί μέχρι τότε.



Εικόνα 2. Αριθμητική εκτίμηση 'έξυπνων συσκευών' μέχρι το τέλος του 2020

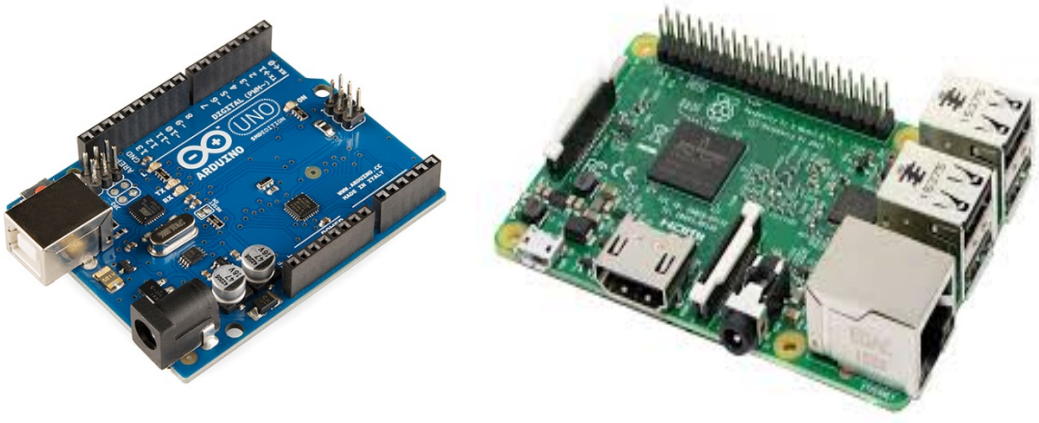
1.3 Τι είναι το ‘έξυπνο σπίτι’ (Smarthome)

Όταν μιλάμε για ‘έξυπνο σπίτι’, οι δυνατότητες της έννοιας πηγαίνουν αρκετά μακριά. Αντί για μεμονωμένες συσκευές που λειτουργούν ανεξάρτητα, ένα έξυπνο σπίτι ενσωματώνει πολλαπλά υποσυστήματα που ελέγχονται όλα από έναν κύριο ελεγκτή αυτοματισμού σπιτιού.



Εικόνα 3. Παράδειγμα πρότυπου 'έξυπνου' σπιτιού

Αυτός ο κύριος ελεγκτής αυτοματισμού είναι σαν τον κεντρικό επεξεργαστή ενός υπολογιστή (CPU), λαμβάνει εισροές από όλες τις συσκευές γύρω από το σπίτι, εκδίδει εντολές και ελέγχει τα πάντα. Αυτοί οι ελεγκτές γενικά δεν χρησιμοποιούν πολύπλοκο λογισμικό, επιτρέποντάς τους να εκτελέσουν απλές ή πολλαπλές ενέργειες με βάση την ποικιλία των συμβάντων. Οι δύο πιο ευρέως διαδεδομένοι τέτοιου τύπου ελεγκτές, είναι το ‘Arduino’ και το ‘Raspberry Pi’.



Εικόνα 4. Arduino Uno, Raspberry Pi

2 Hardware, Software και API's που θα χρησιμοποιήσουμε

Παρακάτω ακολουθεί εκτενής ανάλυση και παρουσίαση τόσο του Hardware και του Software όσο και των εφαρμογών (API's) που θα χρησιμοποιήσουμε στο σύστημα μας ώστε να αναπτυχθεί το 'Smart Home'.

2.1 Υλικό (Hardware)

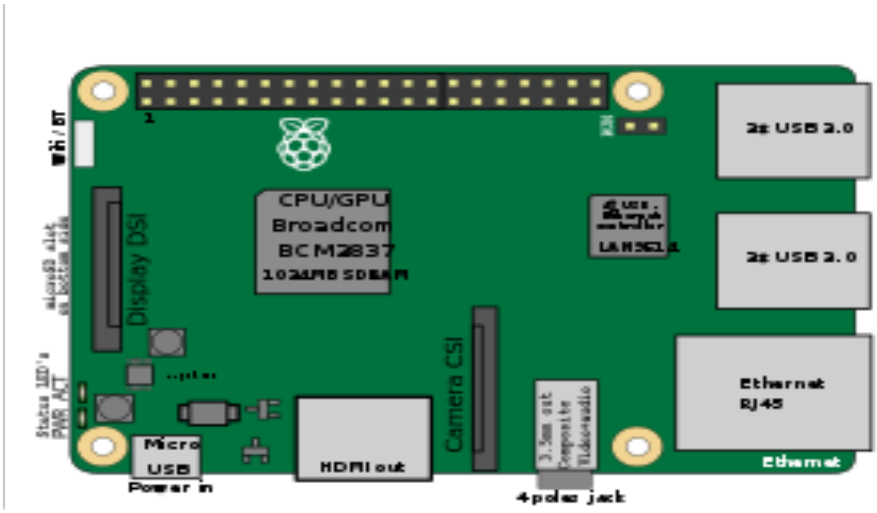
Παρακάτω περιγράφεται το υλικό ('Hardware') που θα χρησιμοποιήσουμε για την ανάπτυξη της εφαρμογής μας 'SmartHome'. Το συγκεκριμένο υλικό που θα χρησιμοποιήσουμε μπορεί να το αποκτήσει οποιοσδήποτε έναντι κάποιας αξίας και γενικά έχει επιλεχθεί προσεκτικά ώστε να είναι εύκολο τόσο στην διασύνδεση με τον χρήστη όσο και στην συναρμολόγηση από αυτόν.

2.1.1 Raspberry Pi 3 model B

Το 'Raspberry Pi' είναι ένας μικροϋπολογιστής όπου σχεδιάστηκε και κατασκευάστηκε στο Ηνωμένο Βασίλειο από την ομώνυμη εταιρία 'Raspberry Pi Foundation'. Ο σκοπός δημιουργίας του ήταν η δυνατότητα να παρέχονται χαμηλού κόστους υπολογιστές σε όλο τον κόσμο ώστε να μπορούν όλοι να αποκτήσουν βασικές γνώσεις στον προγραμματισμό. Ωστόσο, η απήχηση που είχε στο κοινό ήταν τεράστια, τόσο για την χαμηλή τιμή πώλησής του όσο για τις απεριόριστες δυνατότητες που παρέχει στον χρήστη. Χαρακτηριστικό της επιτυχίας του είναι πως μέχρι το Φεβρουάριο του 2015 είχαν πωληθεί πέντε εκατομμύρια (5.000.000) 'Raspberry Pi' ενώ μέχρι τον Νοέμβριο του επόμενου έτους, ο αριθμός αυτός εκτοξεύθηκε στα έντεκα εκατομμύρια (11.000.000) συσκευές.

Το 'Raspberry Pi' υποστηρίζει ανοιχτού κώδικα λογισμικά και αυτό δίνει την δυνατότητα στους προγραμματιστές να δοκιμάζουν πολλές νέες ιδέες πάνω σ' αυτό. Ο επεξεργαστής του είναι ο '64-bit quad-core ARM Cortex-A53' στα 1.2 GHz με 512 KB shared L2 cache και 1024Mb Sdram ενώ η κάρτα γραφικών που διαθέτει είναι η

‘ARM1176JZF-S’ στα 700 MHz μαζί με L1 cache στα 16 KB και L2 cache 128 KB.



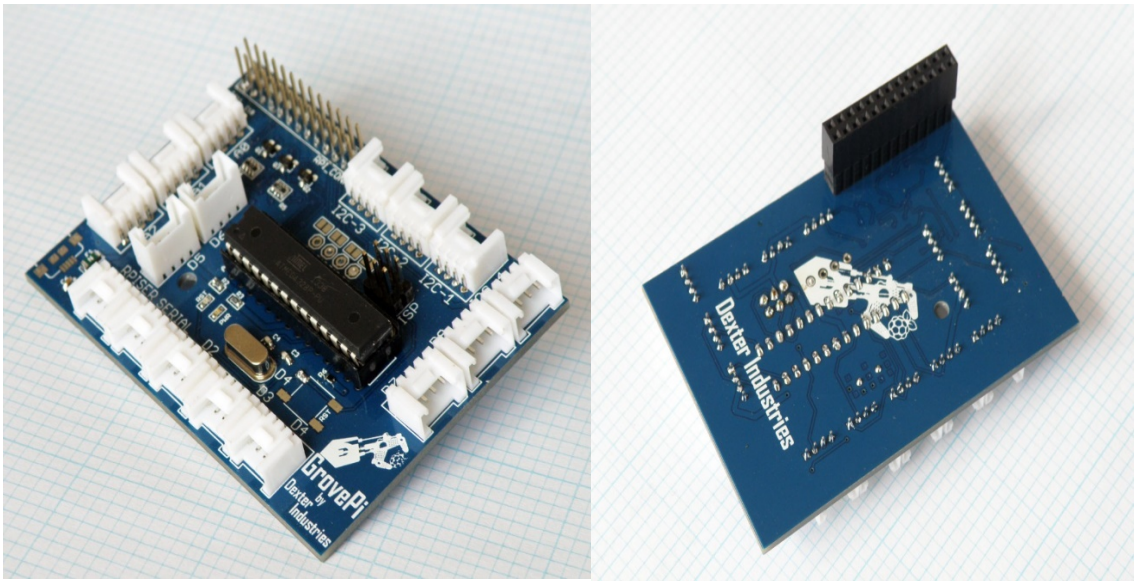
Εικόνα 5. Μακέτα με χαρακτηριστικά από το ‘Raspberry Pi 3’

Στο σύστημα μας το ‘Raspberry Pi 3’ θα αποτελεί τον κεντρικό υπολογιστή – διακομιστή, καθώς θα’ ναι αυτό το οποίο θα επεξεργάζεται τα δεδομένα των αισθητήρων, θα πραγματοποιεί όλες τις συνδέσεις (upload/download) από το διαδίκτυο και θα αποτελεί τον κεντρικό διακομιστή (Web Server) για την ιστοσελίδα που θα κατασκευάσουμε. Το λογισμικό του (Operating System) είναι μια τροποποιημένη έκδοση του ‘Raspbian OS’ το ‘Dexter Industries Raspbian OS’, ενδεδειγμένο για τη χρησιμοποίηση του ‘Raspberry Pi’ στη διαχείριση αισθητήρων και ευρύτερα στον τομέα της ρομποτικής. Άλλα λογισμικά τα οποία υποστηρίζονται επίσης είναι: το Ubuntu MATE, το Snappy Ubuntu Core, το Windows 10 IoT Core, το RISC OS και πολλές άλλες εκδόσεις και τροποποιήσεις των παραπάνω.

2.1.2 Grove Pi Shield

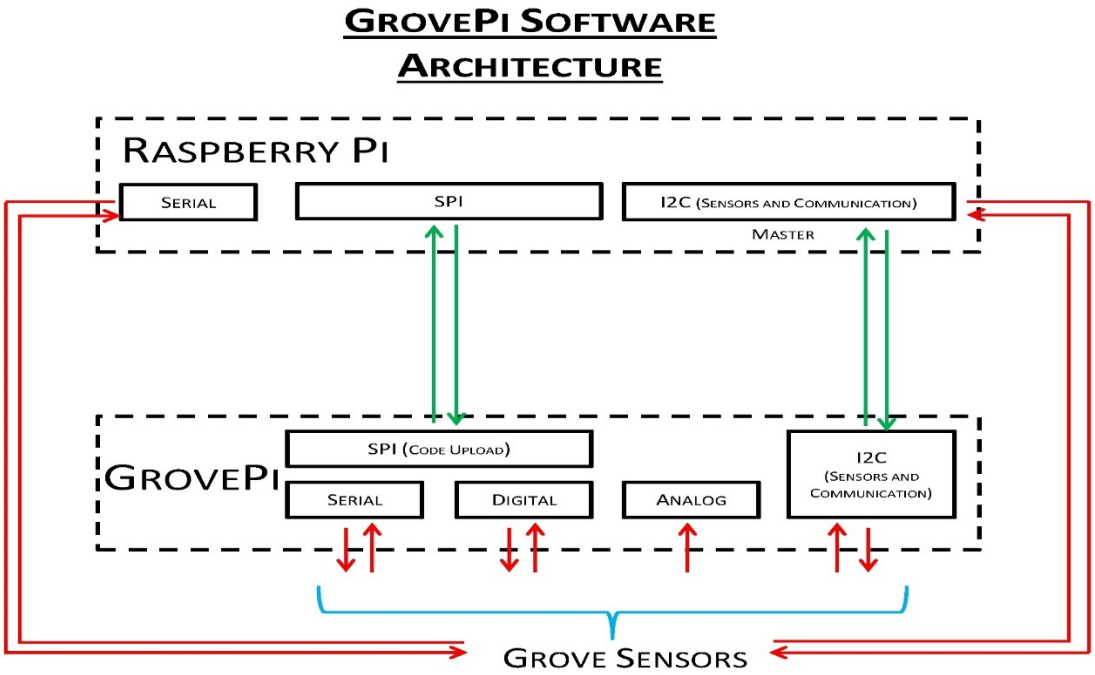
Το ‘GrovePi’ είναι ένας μικροϋπολογιστής όπως και το ‘Raspberry Pi’ αλλά με πολύ λιγότερη υπολογιστική ισχύ και περιορισμένες δυνατότητες. Αποτελεί τον κεντρικό επεξεργαστή για τον αριθμό των αισθητήρων που είναι συνδεδεμένοι σ’ αυτό. Ωστόσο, δεν μπορεί να λειτουργήσει αυτόνομα (Stand-Alone) και γι’ αυτό συνδέεται με το

‘Raspberry Pi’ ως επέκταση του ευρύτερου συστήματος.



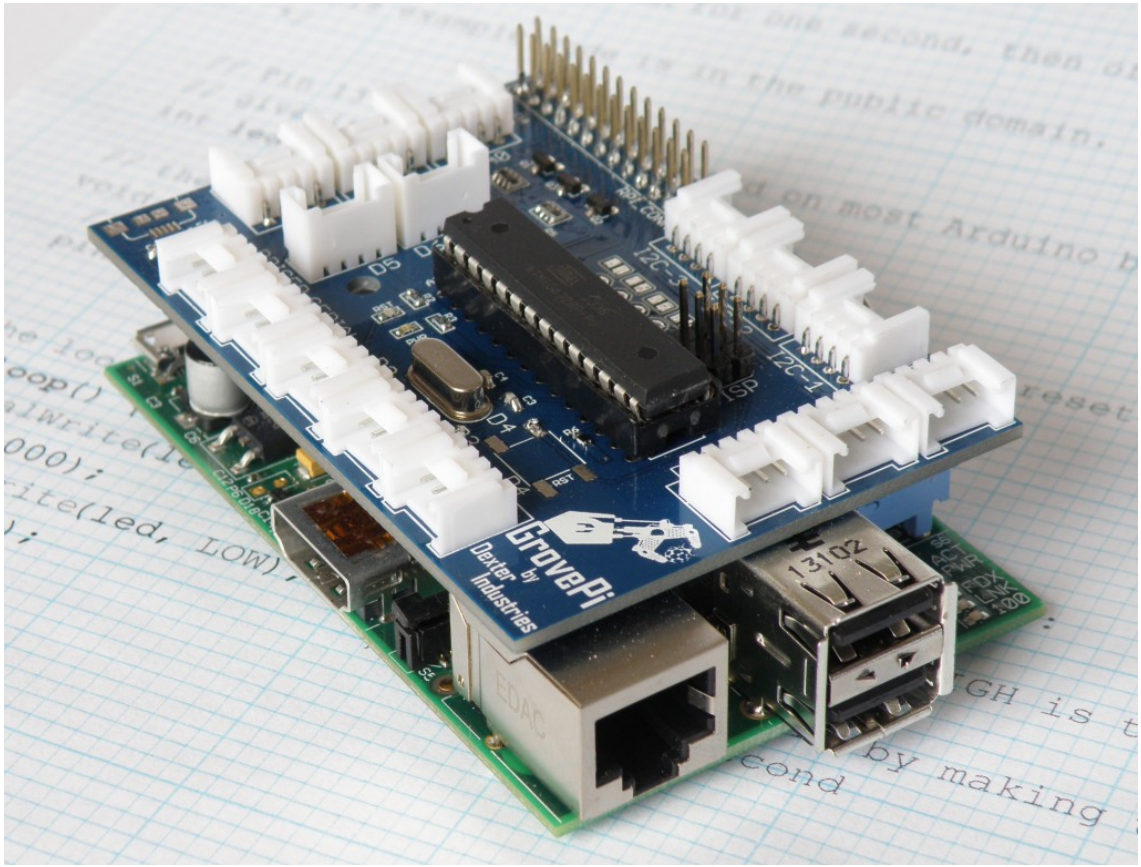
Εικόνα 6. Το ‘GrovePi’ μπρος και πίσω όψη

Η επικοινωνία μεταξύ των δύο γίνεται μέσω ‘I2C’ πρωτόκολλο. Το ‘Raspberry Pi’ στέλνει τις εντολές και το ‘GrovePi’ τις επεξεργάζεται και στέλνει πίσω τα αποτελέσματα (feedback) που έχουν προκύψει ώστε να γίνουν οι κατάλληλες ενέργειες.



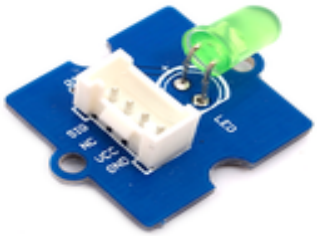
Εικόνα 7. ‘I2C’ πρωτόκολλο επικοινωνίας μεταξύ ‘Raspberry Pi’ και ‘GrovePi’

Οι αισθητήρες που συνδέονται με το ‘GrovePi’ είναι όλοι ‘plug & play’ δηλαδή δεν χρειάζεται να εγκαταστήσουμε οδηγούς (drivers) ή να τους προγραμματίσουμε για να λειτουργήσουν. Επίσης, υποστηρίζονται τόσο ψηφιακοί αισθητήρες όσο και αναλογικοί ή σειριακοί. Αυτό δίνει την δυνατότητα να έχει πιο ακριβή αποτελέσματα σε μετρήσεις όπως αυτά της θερμοκρασίας για παράδειγμα. Όπως προαναφέρθηκε παραπάνω, το ‘GrovePi’ δεν έχει την ίδια υπολογιστική ισχύ όπως το ‘Raspberry Pi’ όμως είναι ικανό να διαχειριστεί από μόνο του τα δεδομένα που καταγράφει καθώς και να τα επεξεργαστεί μέχρι ένα βαθμό, χωρίς να επιβαρύνει το ‘Raspberry Pi’ με περιττές ενέργειες.

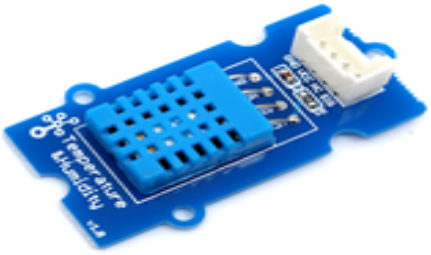


Εικόνα 8. Τα ‘Raspberry Pi’ και ‘GrovePi’ συνδεδεμένα

2.1.3 Αισθητήρες που θα χρησιμοποιηθούν στην εφαρμογή



Εικόνα 9. Led ενδειξης ενεργου συστηματος



Εικόνα 10. Αισθητήρας θερμοκρασίας/υγρασίας

Εικόνα 11. Αισθητήρας φωτός

Εικόνα 12. Αισθητήρας θορύβου

Εικόνα 13. Relay 220V

2.2 Λογισμικό (Software)

Το λογισμικό του κεντρικού υπολογιστή είναι το ‘Raspbian OS’ που αποτελεί μια πιο ‘ελαφριά’ έκδοση των ‘Linux’. Είναι με τέτοιο τρόπο ανεπτυγμένο ώστε να λειτουργεί ιδανικά σε μικροϋπολογιστές και να έχει όσο το δυνατόν καλύτερη διασύνδεση με διάφορους τύπους αισθητήρων.

Ενδεικτικά, παρακάτω περιγράφονται μέχρι ένα βαθμό οι εφαρμογές που θα χρησιμοποιήσουμε για το σύστημα μας, καθώς και οι δυνατότητες που παρέχει η κάθε μία από αυτές.

2.3 Διεπαφές προγραμματισμού εφαρμογών (API's)

Τι ακριβώς είναι το API; Διαρκώς ακούμε πόσο χρήσιμο είναι το API ενώ δεν είναι λίγοι εκείνοι που επισημαίνουν την σημαντικότητα του για τις επιχειρήσεις. Πώς θα μπορούσαμε ακριβώς να ορίσουμε το API;

Το API είναι η συντομογραφία που σχηματίζεται από το Application Programming Interface (Διασύνδεση προγραμματισμού εφαρμογών). Το API είναι ένα ενδιάμεσο λογισμικό που επιτρέπει την επικοινωνία μεταξύ δύο εφαρμογών. Με απλά λόγια, είναι ο φορέας που παραδίδει το αίτημά σας στον πάροχο που είστε και στη συνέχεια επιστέφει την απάντηση πίσω σε εσάς. Ένα API καθορίζει τις λειτουργίες που είναι ανεξάρτητες από τις αντίστοιχες υλοποιήσεις τους. Αυτό επιτρέπει στις εν λόγω υλοποιήσεις και ορισμούς να διαφέρουν χωρίς να εκθέτει ο ένας τον άλλον.

Ως εκ τούτου, μέσα από ένα καλό API είναι ευκολότερο να αναπτυχθεί ένα πρόγραμμα με την παροχή των δομικών στοιχείων, καθώς είναι σύνηθες όταν οι προγραμματιστές ξεκινάνε την δημιουργία ενός κώδικα να μην ξεκινάνε από το απόλυτο μηδέν. Για την παραγωγή του έργου είναι σημαντικό να υπάρχει η δυνατότητα έστω και ένα μικρό τμήμα απο το API να επαναλαμβάνεται ακόμα και σε πολύπλοκες διαδικασίες. Η ταχύτητα που το API επιτρέπει στους προγραμματιστές να χτίσουν τις εφαρμογές είναι σημαντική για τον σημερινό ρυθμό ανάπτυξης εφαρμογών.

Σήμερα οι προγραμματιστές είναι πιο παραγωγικοί από ποτέ και δεν χρειάζεται να επανεφεύρουν τον τροχό κάθε φορά που ξεκινούν να γράφουν ένα νέο πρόγραμμα. Αντ' αυτού μπορούν να επικεντρωθούν στις εφαρμογές τους και στις λειτουργίες που προσφέρονται από το API με ταχύτητα και ευκινησία.

Ένα από τα κύρια πλεονεκτήματα του API είναι ότι επιτρέπει την άντληση πληροφοριών από το ένα σύστημα στο άλλο. Για όσο ο πάροχος υπηρεσιών παραδίδει στο τελικό σημείο οι μεταβολές στην υποδομή πίσω από το τελικό σημείο δεν θα είναι

ορατές από τις εφαρμογές που βασίζονται στο συγκεκριμένο API. Ως εκ τούτου, ο πάροχος υπηρεσιών δίνει μεγάλη ευελιξία σε ό,τι αφορά τις προσφερόμενες υπηρεσίες. Για παράδειγμα, εάν η υποδομή πίσω από το API περιλαμβάνει κάποιο κέντρο δεδομένων με φυσικούς πόρους, τότε ο πάροχος υπηρεσιών μπορεί εύκολα να στραφεί στα virtual servers που υπάρχουν στο cloud.

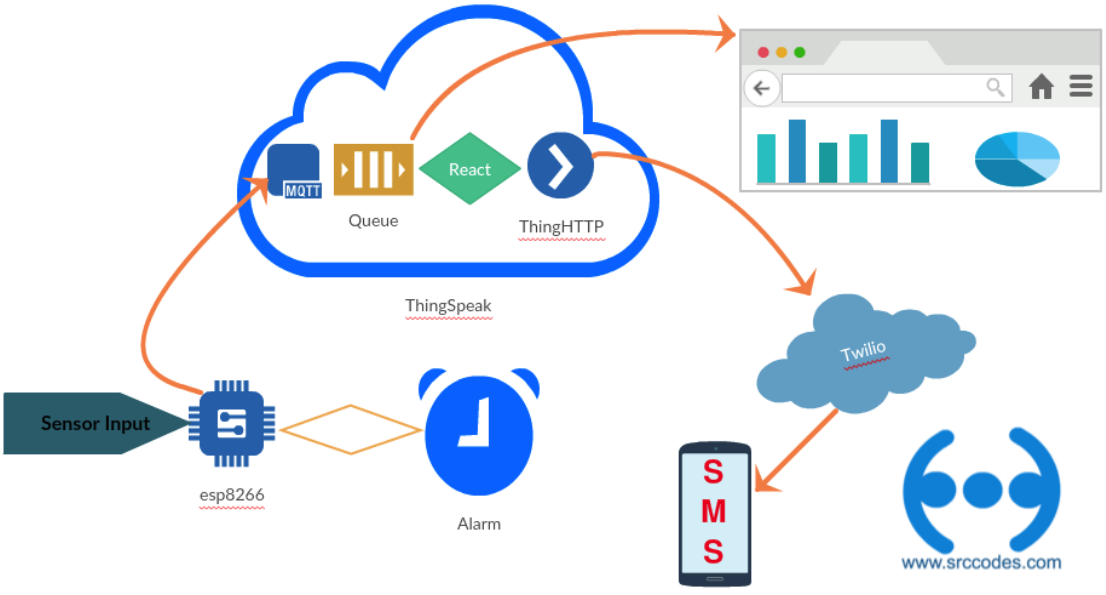
Εάν το λογισμικό που εκτελείται σε αυτούς τους διακομιστές (όπως π.χ. λογισμικό επεξεργασίας πιστωτικών καρτών) είναι γραμμένο σε γλώσσα προγραμματισμού Java και εκτελείται σε έναν, βασισμένο στην Oracle, διακομιστή Java εφαρμογών τότε ο πάροχος μπορεί να το μεταφέρει στο Note.js (Javascript στην πλευρά του διακομιστή) που εκτελείται σε Windows Azure.

Η δυνατότητα που δίνει το API να εναλλάσσονται τα συστήματα με τόση ευκολία είναι το όραμα κάθε προγραμματιστή.

Ας δούμε λοιπόν ποια API's θα χρησιμοποιήσουμε εμείς και γιατί.

2.3.1 ThingSpeak

Για να μπορούμε να αξιοποιήσουμε τα δεδομένα που συλλέγονται από τους αισθητήρες που έχουμε εγκαταστήσει, θα χρησιμοποιήσουμε την εφαρμογή 'ThingSpeak'.



Εικόνα 5. Διάγραμμα τρόπου λειτουργείας της εφαρμογής

Η συγκεκριμένη εφαρμογή μας δίνει τεράστιες δυνατότητες όσον αφορά την επεξεργασία των δεδομένων που συλλέγουμε και αυτό γιατί αρχικά, έχει άμεση σχέση με το πολύ εργαλείο 'MATLAB' , καθώς επίσης η εφαρμογή αναπτύχθηκε αποκλειστικά για χρήση σε 'IoT' συσκευές.

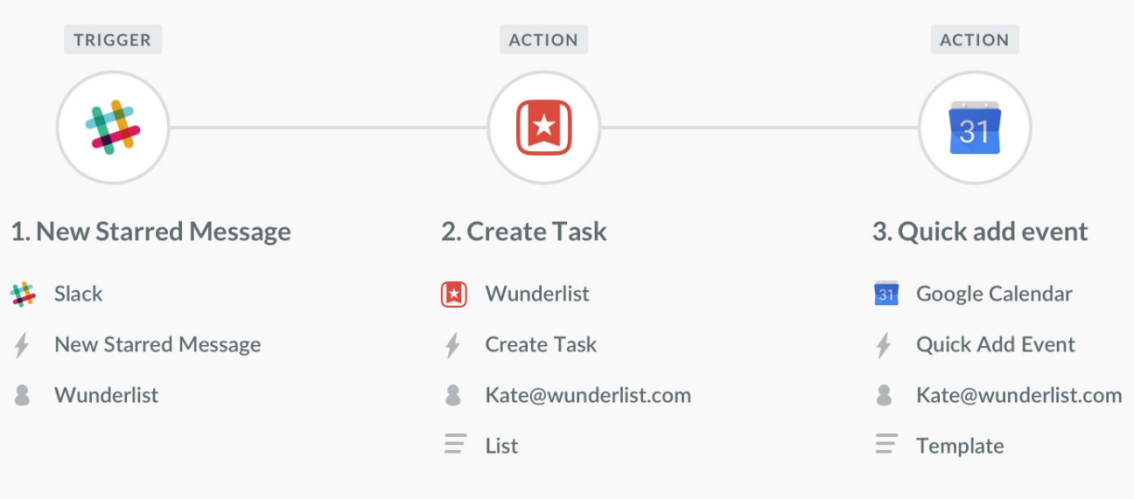
Μερικές από τις δυνατότητες της εφαρμογής είναι η συλλογή δεδομένων από σταθμό(στην περίπτωσή μας το 'SmartHome' μας, η προβολή των δεδομένων αυτών(Private/Public), η εκτενής επεξεργασία και ανάλυση των δεδομένων μέσα από τις διάφορες πλατφόρμες της εφαρμογής(παρέχει μέχρι και δυνατότητα 'εκπαίδευσης' συστήματος (AI) , μέσω του 'MATLAB'.

Επίσης, μας δίνει την δυνατότητα να παρακολουθούμε τα δεδομένα που ανεβάζουμε ανά πάσα στιγμή και επίσης με συνεχή ροή ανανέωσης αυτών.

Τέλος, όπως όλες οι υπηρεσίες που θα χρησιμοποιήσουμε, έτσι και αυτήν παρέχεται δωρεάν για κάποιες συγκεκριμένες λειτουργείες της.

2.3.2 Zapier

Για να μπορούμε να έχουμε συνεχή ενημέρωση σχετικά με τα δεδομένα του συστήματος, και λέγοντας ενημέρωση εννοούμε είτε ειδοποιήσεις είτε mail στον προσωπικό λογαριασμό μας, θα χρησιμοποιήσουμε την εφαρμογή 'Zappier'.



Εικόνα 6. Προσομοίωση λειτουργίας της εφαρμογής

Το 'Zappier' είναι μια 'online'-εφαρμογή αυτοματοποίησης δεδομένων, δηλαδή, μπορούμε να παρέχουμε δεδομένα και ανάλογα με τις τιμές ή ανάλογα με τις

κατάλληλες συνθήκες που έχουμε θέσει εμείς, να εκτελούνται αυτόματα μέσω της εφαρμογής κάποιες άλλες λειτουργείες.
Για παράδειγμα, έστω πως θέλουμε να ειδοποιούμαστε με email όποτε τίθεται σε λειτουργεία το καλοριφέρ. Η εφαρμογή συλλέγει τα δεδομένα που αφορούν την λειτουργεία ή μη του καλοριφέρ, τα ελέγχει και αν είναι αληθή(αναμμένο καλοριφέρ) μας ειδοποιεί με email στον προσωπικό μας λογαριασμό.

Η υπηρεσία μπορεί να λειτουργεί με έναν τεράστιο αριθμό άλλων εφαρμογών(περισσότερες από 1000), αρκεί να παίρνει τα δεδομένα που απαιτούνται για την λειτουργεία της.

Εικόνα 7. Λογότυπο εφαρμογής και διασύνδεσης αυτής με άλλες

Ενδεικτικά θα χρησιμοποιήσουμε την σύνδεση με το Gmail για την ειδοποίηση με email που αναφέραμε πιο πάνω.

Στην ελεύθερη έκδοση της η εφαρμογή μας δίνει την δυνατότητα χρησιμοποίησης μέχρι και 8 ‘zaps’, δηλαδή 8 αυτοματοποιημένες λειτουργείες με άλλες εφαρμογές.

2.3.3 DarkSky

Για να έχουμε πάντα αξιόπιστα και ακριβή δεδομένα σχετικά με τις εξωτερικές καιρικές συνθήκες που επικρατούν, θα χρησιμοποιήσουμε τις δυνατότητες που μας παρέχει, δωρεάν, η εταιρία 'DarkSky' με την ομώνυμη εφαρμογή της.

Η βασική ιδέα της εφαρμογής βασίζεται στο γεγονός ότι χιλιάδες 'ρομπότ-ελεγκτές' ανά τον κόσμο πραγματοποιούν μετρήσεις ακριβείας όσον αφορά τις καιρικές συνθήκες, αυτές αποθηκεύονται στην κεντρική βάση δεδομένων του συστήματος και έπειτα γίνονται η οποία είναι προσβάσιμη στους χρήστες.

Εικόνα 8. Λογότυπο εφαρμογής

Η εταιρία στο πρώιμο στάδιο της ήταν 'self-funded', δηλαδή όλα τα έξοδα καλύπτονταν από τους δημιουργούς της.

Πλέον συνεργάζεται και έχει εξαγοραστεί από μία άλλη εταιρία, την 'Applied Invention', με σκοπό την ανάπτυξη και εξέλιξη της εφαρμογής για πιο γενικευμένη χρήση.

2.3.4 NoiP

Για να μπορούμε να έχουμε πάντα πρόσβαση στο 'πρότζεκτ' μας, κυρίως απομακρυσμένη, θα πρέπει να έχουμε αναθέσει στο διακομιστή μας ένα κεντρικό όνομα υπολογιστή(Hostame).

Εικόνα 14. Λογότυπο εφαρμογής

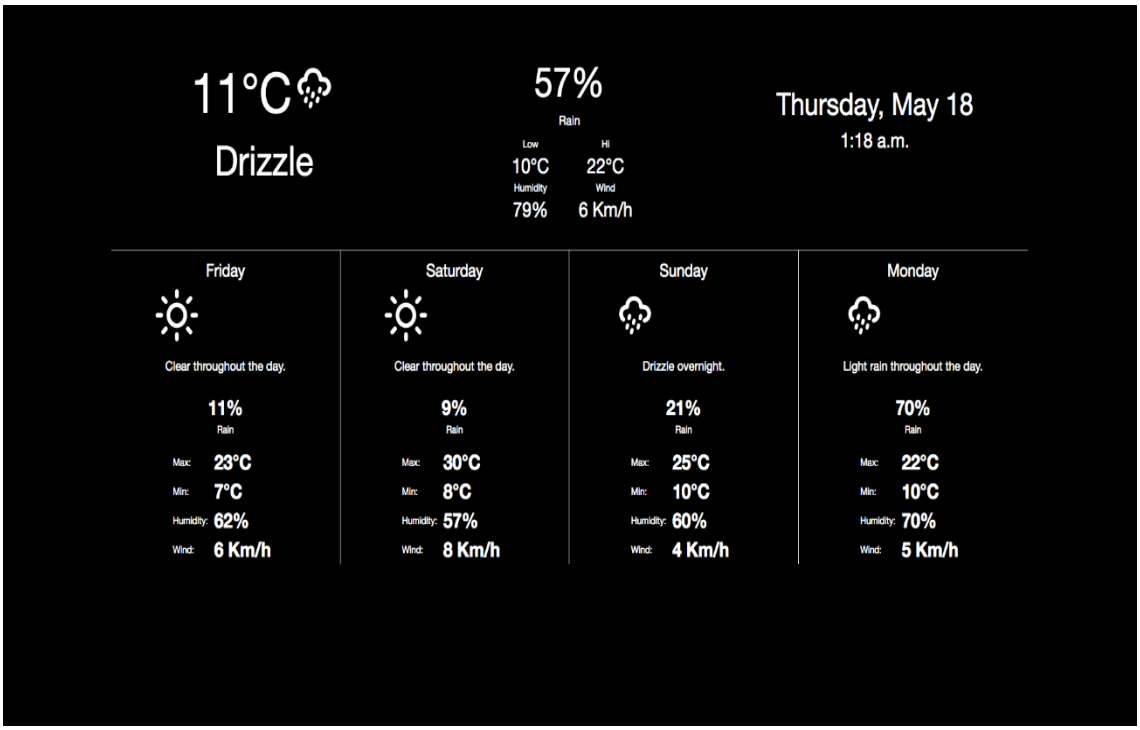
Η υπηρεσία ‘No-IP’ δίνει την δυνατότητα σε μία δυναμική IP να λειτουργεί σαν στατική , δηλαδή να έχουμε το ‘Hostname’ που επιθυμούμε χωρίς να χρειάζεται να γνωρίζουμε την εκάστοτε διεύθυνση IP που έχει ανατεθεί στον κεντρικό υπολογιστή μας.

Η συγκεκριμένη υπηρεσία είναι αρκετά αξιόπιστη ακόμα και στην ‘ελευθέρη’ έκδοση της(30 ημέρες, έπειτα χρειάζεται μια τυπική ανανέωση της άδειας χρήσης και λειτουργεί εκ νέου).

2.4 Δημιουργία ιστοσελίδας πρόβλεψης καιρού

Η ιστοσελίδα περιέχει τα δεδομένα που παίρνουμε από το ‘Dark Sky-API’ και είναι σύμφωνα με τις υπηρεσίες του κατά 99% ακριβείς σε πρόβλεψη 36 ωρών. Προβάλλεται η πρόβλεψη πέντε ημερών με λεπτομερή στοιχεία για διάφορες καιρικές συνθήκες.

Η ιστοσελίδα έχει δημιουργηθεί με σκοπό ο χρήστης να έχει μία πλήρη και ταυτόχρονα περιεκτική εικόνα πρόβλεψης του καιρού, καθώς τα ίδια δεδομένα που επεξεργάζεται και ο κεντρικός υπολογιστής του συστήματός μας και ανάλογα εκτελούνται διάφορες εντολές που αφορούν κυρίως την θέρμανση του σπιτιού. Αποτελείται από δύο αρχεία, το ‘dark-sky.php’ και το ‘index.html’ τα οποία είναι γραμμένα σε γλώσσες προγραμματισμού ‘PHP’ και ‘HTML’.



Εικόνα 15. Η ιστοσελίδα πρόβλεψης καιρού

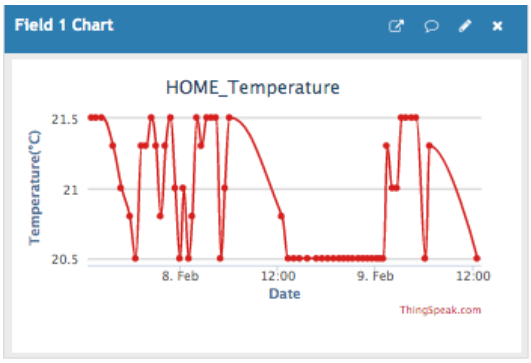
Ακολουθεί ο κώδικας μόνο του πρώτου, καθώς το ‘index.html’ περιέχει 500+ γραμμές κώδικα:

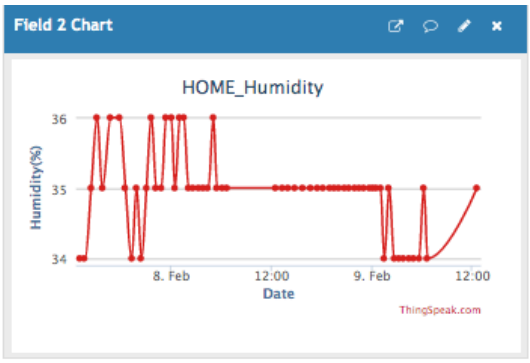
```
<?php

class Forecast
{
    const API_ENDPOINT = 'https://api.forecast.io/forecast/';
    private $api_key;
    public function __construct($api_key)
    {
        $this->api_key = $api_key;
    }
    private function request($latitude, $longitude, $time = null, $options = array())
    {
        $request_url = self::API_ENDPOINT
            . $this->api_key
            . '/'
            . $latitude
            . ','
            . $longitude
            . ((is_null($time)) ? '' : ','. $time);
        if (!empty($options)) {
            $request_url .= '?'. http_build_query($options);
        }

        $response = json_decode(file_get_contents($request_url));
        $response->headers = $http_response_header;
        return $response;
    }
    public function get($latitude, $longitude, $time = null, $options = array())
    {
        return $this->request($latitude, $longitude, $time, $options);
    }
}

// Add your API key here
$forecast = new Forecast('put your api key here!!!');

// Set timezone
date_default_timezone_set('Europe/Athens');
$date = new DateTime();

// Get the forecast at a given time
echo json_encode($forecast->get('39.640882','22.421276'));


?>
```

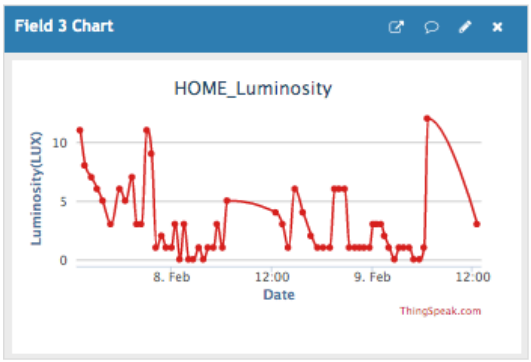
2.5 Αποτελέσματα, προβολή και διαχείριση δεδομένων (Results)

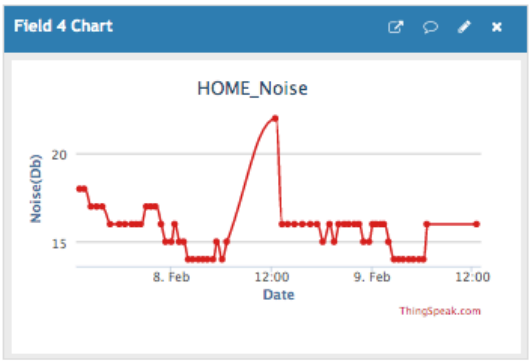
Εφόσον έχουν γίνει όλα σωστά, τα πρώτα αποτελέσματα που θα βλέπουμε θα είναι κάπως έτσι.

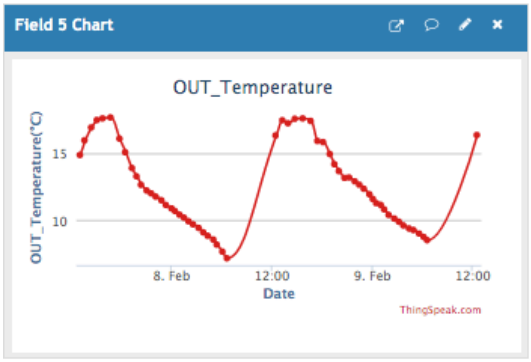
- Δεδομένα εσωτερικών αισθητήρων και δεδομένα εξωτερικού χώρου, προβολή ιστόγραμμάτων μέσω του ThingSpeak API

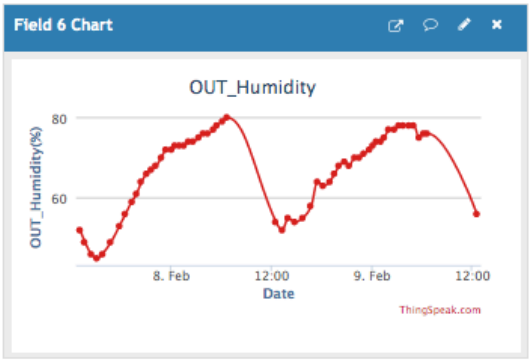


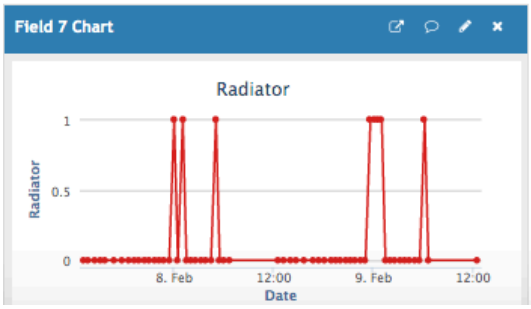


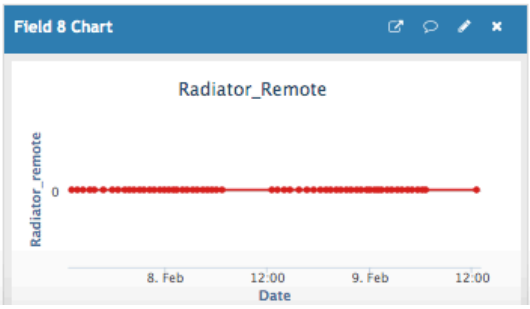




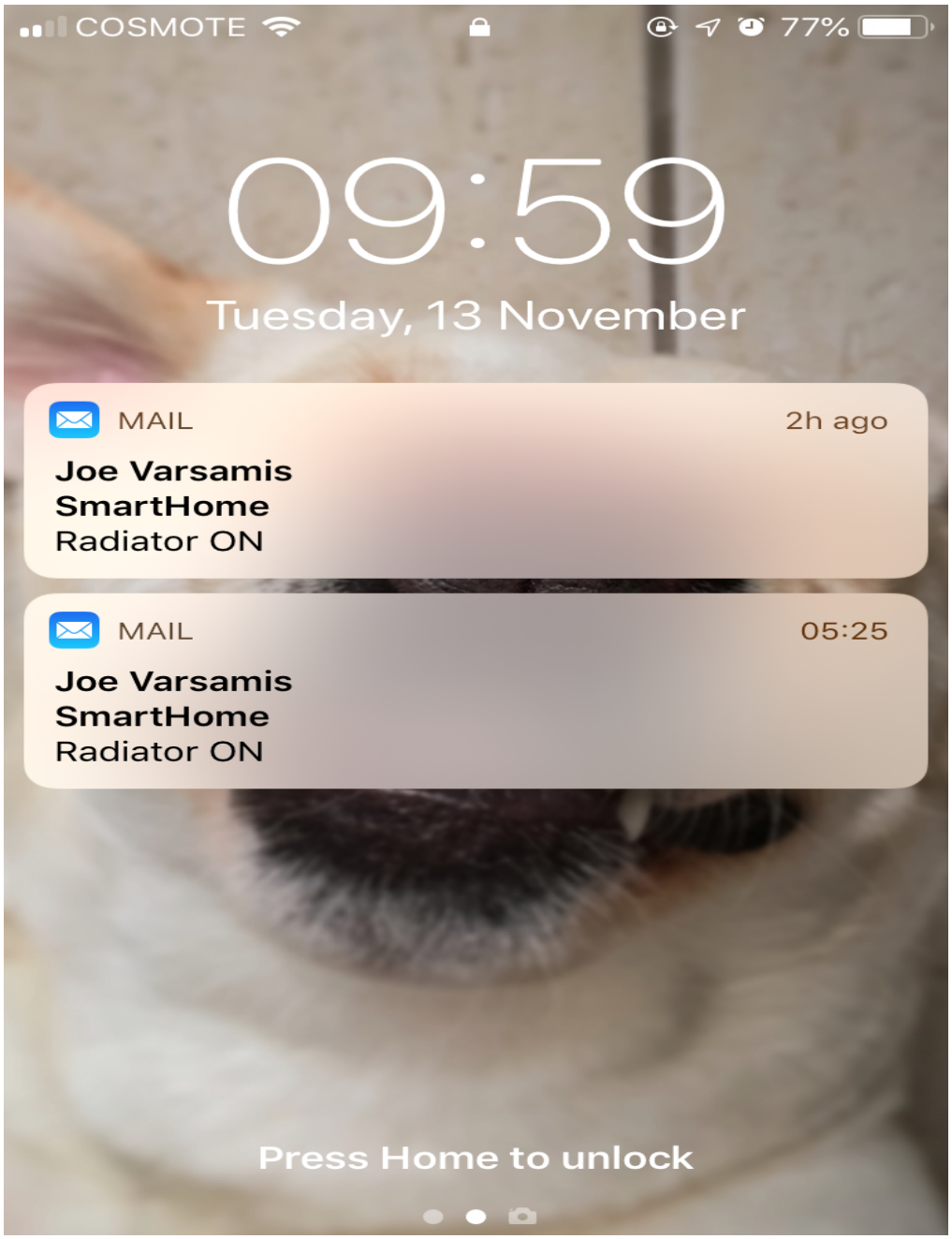








- Ειδοποίηση(email) ενεργοποίησης θέρμανσης τόσο αυτόματα όσο και χειροκίνειτα.



Τα δεδομένα μπορούμε να τα χρησιμοποιήσουμε με όποιον τρόπο θέλουμε καθώς μπορούμε να τα εξάγουμε ως xml, json και csv αρχείο.

3 Ανάπτυξη συστήματος Μηχανικής Μάθησης (Machine Learning)

Ακολουθεί εκτενής παρουσίαση και ανάλυση του συστήματος που αναπτύχθηκε με Μηχανική Μάθηση, οι αλγόριθμοι-μεθοδολογίες που χρησιμοποιήθηκαν καθώς και η εξόρυξη, προβολή και επεξεργασία των νέων δεδομένων που προέκυψαν μετά την υλοποίηση της Μηχανικής Μάθησης.

3.1 Υφιστάμενη κατάσταση (State of the Art)

Η υφιστάμενη κατάσταση του συστήματος έχει ως εξής:

- Οι εσωτερικοί αισθητήρες τροφοδοτούν με δεδομένα θερμοκρασίας, υγρασίας, φωτεινότητας και θορύβου το σύστημα
- Το DarkSky ‘τροφοδοτεί’ το σύστημα με τα εξωτερικά δεδομένα όπως θερμοκρασία, υγρασία, ταχύτητα και κατεύθυνση ανέμου, ακτινοβολία UV κλπ
- Κατά την πρώτη εκκίνηση του συστήματος (first boot), το σύστημα ‘ρωτάει’ τον χρήστη ποια θέλει να είναι η επιθυμητή εσωτερική θερμοκρασία/υγρασία
- Ανάλογα με τα δεδομένα εισροής, ο αλγόριθμος εκρέει ένα αποτέλεσμα το οποίο είναι η ενεργοποίηση ή όχι της Θέρμανσης/Ψύξης (Radiator: boolean)
- Η ενεργοποίηση διαρκεί για μία ώρα (60 λεπτά) και έπειτα ξαναγίνεται έλεγχος των όλων των παραπάνω δεδομένων
- Επίσης ο χρήστης μπορεί αν επέμβει (override) και να ενεργοποιήσει ή απενεργοποιήσει το σύστημα χειροκίνητα (manually)

Πρώτη Έκδοση Συστήματος:

- Το σύστημα επεξεργάζεται τα δεδομένα και απλώς αποφασίζει για την ενεργοποίηση του ή όχι
- Δεν λαμβάνει υπόψη του άλλα δεδομένα όπως η πρόβλεψη του καιρού, δηλαδή μέσα στις επόμενες ώρες πως θα κυμανθούν οι τιμές των εξωτερικών δεδομένων
- Αυτό ουσιαστικά σημαίνει πως το σύστημα είναι απλώς ‘Αυτόματο’ και σε καμία περίπτωση ‘Έξυπνο’

Δεύτερη Έκδοση Συστήματος:

- Αξιοποίηση της πρόβλεψης καιρού με σκοπό την συνεχόμενη εκτέλεση του συστήματος χωρίς διακοπές, δηλαδή αν η εξωτερική θερμοκρασίας πρόκειται να πέσει κατά 5 βαθμούς κελσίου για παράδειγμα, να συνεχίσει η θέρμανση να είναι ενεργοποιημένη και να γίνει επανέλεγχος των δεδομένων όταν οι τιμές φτάσουν στα επιθυμητά επίπεδα
- Υπολογισμός του χρόνου που απαιτείται για να φτάσει ο λέβητας θέρμανσης στην πλήρη ισχύ του, δηλαδή ποιο είναι το χρονικό διάστημα (ambient_time) που μεσολαβεί από τη στιγμή που θα ενεργοποιηθεί το καλοριφέρ μέχρι τη στιγμή που το σώμα θα έχει πλήρη ισχύ(να 'καίει' δηλαδή)
- Αποθήκευση όλων των χρονικών διαστημάτων αυτών σε εξωτερικό αρχείο τύπου (overall_time.txt), προβολή του συνολικού χρόνου εκτέλεσης
- Επεξεργασία του συνολικού χρόνου εκτέλεσης με σκοπό τον έλεγχο των λογαριασμών θέρμανσης και εξακρίβωση αυτών ή τον υπολογισμό του κόστους θέρμανσης

Τρίτη Έκδοση Συστήματος:

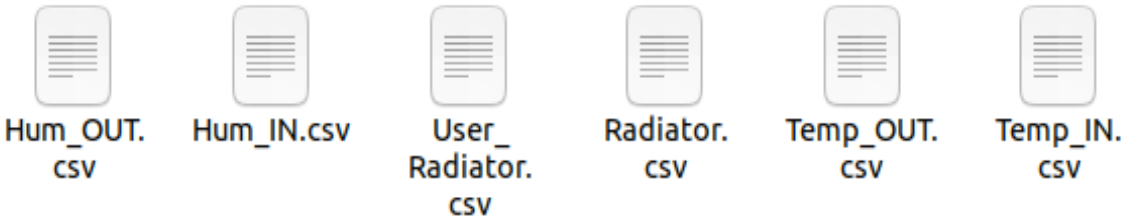
- Συνεχής εκτέλεση του συστήματος χωρίς διακοπές για έλεγχο ανά μία ώρα
- Υπό ιδανικές συνθήκες, δηλαδή μεταξύ -5 έως +10 βαθμούς κελσίου, ο χρόνος για να φτάσει ο λέβητας σε πλήρη ισχύ είναι 15 λεπτά
- Πλήρη αξιοποίηση των εξωτερικών περιβαλλοντικών δεδομένων καθώς και του 'ambient_time' με σκοπό τη λειτουργία του συστήματος κυρίως βάσει αυτών
- Ανάπτυξη αλγορίθμου υπολογισμού χρόνου και κόστους κατανάλωσης, προβολή του αποτελέσματος στον πίνακα ελέγχου του συστήματος ή αποθήκευση σε αρχείο(xml, txt) για μεταγενέστερη αξιοποίηση του

Τέταρτη Έκδοση Συστήματος:

- Ανάπτυξη, υλοποίηση και χρήση Μηχανικής Μάθησης στο υπάρχον σύστημα
- Αξιοποίηση όλων των παραμέτρων που επηρεάζουν την εκτέλεση του συστήματος
- Ικανότητα του συστήματος να 'μαθαίνει' από τις διορθώσεις που έχουμε κάνει
- Πλέον το σύστημα που έχει αναπτυχθεί μπορεί να θεωρηθεί 'Έξυπνο'

3.2 Ορισμός Προβλήματος (Problem Statement)

Το μείζων πρόβλημα στην υλοποίηση (implementation) της Μηχανικής Μάθησης, είναι να καταλάβουμε ποια μέθοδο θα χρησιμοποιήσουμε (Regression, Decision Trees, Classification κλπ.). Επίσης τα δεδομένα πρέπει να έχουν κάποια συγκεκριμένη μορφή για να μπορεί να γίνει η επεξεργασία αυτών. Για τον λόγο ότι το ThingSpeak API, το οποίο χρησιμοποιούμε για την καταγραφή όλων των δεδομένων, αποθηκεύει τα δεδομένα σε μορφή που δεν είναι προβλεπόμενη για την περίπτωσή μας, θα επεξεργαστούμε (με SQL querries) τα δεδομένα αρχικά με το Spark. Επίσης, για κάθε αισθητήρα ή γενικότερα για κάθε μεταβλητή, το ThingSpeak API αποθηκεύει όλα τα δεδομένα της κάθε μεταβλητής σε ξεχωριστά αρχεία. Άρα εν τέλει θα έχουμε τα 8 αρχεία: Temp_IN.csv, Hum_IN.csv, Temp_OUT.csv, Hum_OUT.csv, Luminosity.csv, Noise.csv, Radiator.csv, User_Radiator.csv



3.3 Επίλυση Προβλήματος

Η επίλυση του υφιστάμενου προβλήματος μας, μπορεί κάλλιστα να χωριστεί σε τρία στάδια: Ακατέργαστα Δεδομένα, Αλγόριθμοι Ταξινόμησης και Παλινδρόμησης, Ανάπτυξη Κώδικα

3.3.1 Raw Data Input (Ακατέργαστα Δεδομένα Εισροής)

Αρχίζοντας, τα εξωτερικά δεδομένα πρόβλεψης καιρικών συνθηκών που λαμβάνουμε από το DarkSky API, είναι στην επιθυμητή μορφή προς επεξεργασία, δηλαδή αποθηκεύονται είτε σε αρχεία τύπου xml είτε σε αρχεία τύπου json.

Εν συνεχεία, τα δεδομένα από τους εσωτερικούς αισθητήρες που καταγράφονται ανά πάσα στιγμή, πρώτα υφίστανται επεξεργασία από το ThingSpeak API και έπειτα αποθηκεύονται τοπικά στο σύστημα. Ωστόσο, τα δεδομένα δεν βρίσκονται στην επιθυμητή μορφή προς επεξεργασία. Στο σημείο αυτό, θα γίνει χρήση του Spark Framework που μας δίνει τόσο την δυνατότητα να επεξεργαστούμε κάθε είδους δεδομένα εισροών που έχουμε, όσο και στην μαζική εισαγωγή και επεξεργασία αυτών

καθώς το Spark αποτελεί ισχυρό εργαλείο στον τομέα των Μεγάλων Δεδομένων (Big Data) και των Stream Analytics.

3.3.2 Regression and Classification (Παλινδρόμηση & Ταξινόμηση)

Ένας από τους συνηθέστερους τρόπους Μηχανικής Μάθησης είναι μέσω επαγωγών οι οποίες παρέχουν δια του προτασιακού λογισμού ένα μηχανισμό εξαγωγής (όχι απαραίτητα σωστών) λογικών συμπερασμάτων από παραδείγματα.

Οι αλγόριθμοι Επαγωγικής Μάθησης (Inductive Learning algorithms) είναι ένα είδος αλγορίθμων που αναπτύχθηκε στο πλαίσιο της ΤΝ και ιδιαίτερα στο χώρο της Μηχανικής Μάθησης. Σκοπός των αλγορίθμων αυτών είναι η κατάληξη σε αποφάσεις σχετικές με τις σχέσεις που κυριαρχούν μέσα σε ένα σύνολο παραδειγμάτων που έχουν συγκεντρωθεί από παρατηρήσεις. Οι αλγόριθμοι επιβλεπόμενης Επαγωγικής Μάθησης εφαρμόζονται κυρίως σε προβλήματα ταξινόμησης (classification problems) και σε προβλήματα παρεμβολής (regression problems). Δημιουργούνται μοντέλα πρόβλεψης διακριτών τάξεων κατά την κατηγοριοποίηση και αριθμητικών τιμών κατά την παρεμβολή. Για προβλήματα πρόβλεψης, χρησιμοποιείται κυρίως μια παραλλαγή τους, οι αλγόριθμοι ημι-επιβλεπόμενης Μηχανικής Μάθησης, οι οποίοι λειτουργούν με σύνολο εκπαίδευσης μέσα στο οποίο υπάρχουν παραδείγματα μη γνωστές εξόδους.

Στην επιβλεπόμενη Επαγωγική Μάθηση (supervised Inductive Learning) το σύστημα πρέπει να "μάθει", δηλαδή να κατασκευάσει ένα νέο μοντέλο υπό μορφή μιας συνάρτησης πρόγνωσης (predictor function), η οποία θα απεικονίζει δεδομένες εισόδους σε γνωστές, επιθυμητές εξόδους, με απώτερο στόχο τη γενίκευση της συνάρτησης αυτής και για εισόδους με άγνωστη έξοδο. Για τη συνάρτηση πρόγνωσης ισχύουν τα ακόλουθα:

- Κάθε είσοδος, δεδομένη ή μη, που μπορεί να δεχθεί η συνάρτηση χαρακτηρίζεται ως στιγμιότυπο (instance), δημιουργώντας έτσι ένα σύνολο στιγμιότυπων.
- Οι είσοδοι περιγράφονται με βάση τα γνωρίσματα (attributes) που διαθέτουν και έχουν χαρακτηριστεί ως σημαντικά από την αρχή της μελέτης του προβλήματος που καλείται να επιλύσει το σύστημα.
- Οι δεδομένες είσοδοι συγκεντρώνονται από παρατηρήσεις και αποτελούν το λεγόμενο σύνολο εκπαίδευσης (training set) που αποτελεί υποσύνολο του συνόλου στιγμιότυπων.

- Το υπόλοιπο μέρος του συνόλου στιγμιότυπων αποτελεί το σύνολο ελέγχου (test set) που θα χρησιμοποιηθεί κατά τη φάση πιστοποίησης.
- Η συνάρτηση που απεικονίζει μια είσοδο από το σύνολο εκπαίδευσης στη γνωστή της έξοδο καλείται συνάρτηση στόχου (goal function).
- Η τιμή που επιστρέφει η συνάρτηση στόχου για ένα στιγμιότυπο από το σύνολο στιγμιότυπων, δίνεται σε μια μεταβλητή που καλείται μεταβλητή στόχου (goal variable).

Στην επιβλεπόμενη μάθηση, η συμπεριφορά της συνάρτησης στόχου βελτιώνεται μέσω διαδικασιών εκπαίδευσης με τη βοήθεια της συνάρτησης λάθους (error function) που εντοπίζει τη διαφορά της μεταβλητής στόχου από την επιθυμητή έξοδο.

3.3.3 Ανάπτυξη Κώδικα σε Python2.7

Παρακάτω παρουσιάζεται αναλυτικά ο κώδικας που χρησιμοποιήθηκε τόσο για την υλοποίηση και ανάπτυξη της Μηχανικής Μάθησης, όσο και για την μετατροπή των δεδομένων ώστε να φτάσουν στην επιθυμητή προς επεξεργασία μορφή τους.

Κώδικας Μετατροπής Δεδομένων (Spark)

Το παρακάτω python script(myMain3.py) είναι αυτό το οποίο μετατρέπει τα δεδομένα στην προβλεπόμενη μορφή προς επεξεργασία. Πιο συγκεκριμένα, δέχεται εισροές δεδομένων από ξεχωριστά αρχεία τα οποία έχουν όμως ένα κοινό στοιχείο (entry_id). Με την εκτέλεση διάφορων SQL ερωτημάτων μπορούμε να κάνουμε JOIN διάφορα αχρεία μεταξύ τους και εν τέλει να δημιουργήσουμε ένα αρχείο το οποίο θα περιέχει όλα τα γνωρίσματα (attributes) που είναι απαραίτητα για το επόμενο στάδιο, της ανάπτυξης Μηχανικής Μάθησης. Επιπλέον, πρέπει αν δημιουργήσουμε και ένα Training Set, το οποίο θα περιέχει όλες εκείνες τις φορές όπου ο χρήστης ενεργοποίησε χειροκίνητα τη Θέρμανση/Ψύξη.

```
import pyspark
import pyspark.sql.functions as f
import pandas as pd
import os
import matplotlib.pyplot as plt
from pyspark.sql import SparkSession
from pyspark.sql.types import *
from pyspark.sql.functions import col,struct,when


spark = SparkSession.builder.master("local[1]") \
                    .appName('SparkByExamples.com') \
                    .getOrCreate()
spark.sparkContext.setLogLevel('ERROR')
```

```
#Temp_IN#
Temp_INSchema = StructType([
    StructField("created_at",StringType(),True),
    StructField("entry_id",StringType(),True),
    StructField("Temp_IN",StringType(),True)
  ])

Temp_INDataFrame =
spark.read.schema(Temp_INSchema).format("csv").option("sep",",").option("header","true")
.load("/home/ubuntu/Desktop/Thesis/Temp_IN.csv")
Temp_INDataFrame.createOrReplaceTempView("Temp_INTable")
#Temp_INDataFrame.show()

#Hum_IN#
Hum_INSchema = StructType([
    StructField("created_at",StringType(),True),
    StructField("entry_id",StringType(),True),
    StructField("Hum_IN",StringType(),True)
  ])

Hum_INDataFrame =
spark.read.schema(Hum_INSchema).format("csv").option("sep",",").option("header","true").
load("/home/ubuntu/Desktop/Thesis/Hum_IN.csv")
Hum_INDataFrame.createOrReplaceTempView("Hum_INTable")
#Hum_INDataFrame.show()

#Temp_OUT#
Temp_OUTSchema = StructType([
    StructField("created_at",StringType(),True),
    StructField("entry_id",StringType(),True),
    StructField("Temp_OUT",StringType(),True)
  ])

Temp_INDataFrame =
spark.read.schema(Temp_OUTSchema).format("csv").option("sep",",").option("header","true"
).load("/home/ubuntu/Desktop/Thesis/Temp_OUT.csv")
Temp_INDataFrame.createOrReplaceTempView("Temp_OUTTable")
#Temp_INDataFrame.show()

#Hum_OUT#
Hum_OUTSchema = StructType([
    StructField("created_at",StringType(),True),
    StructField("entry_id",StringType(),True),
    StructField("Hum_OUT",StringType(),True)
  ])

Hum_OUTDataFrame =
spark.read.schema(Hum_OUTSchema).format("csv").option("sep",",").option("header","true")
.load("/home/ubuntu/Desktop/Thesis/Hum_OUT.csv")
Hum_OUTDataFrame.createOrReplaceTempView("Hum_OUTTable")
#Hum_OUTDataFrame.show()

#Radiator#
RadiatorSchema = StructType([
    StructField("created_at",StringType(),True),
    StructField("entry_id",StringType(),True),
    StructField("Radiator",StringType(),True)
  ])
```

```
RadiatorDataFrame =
spark.read.schema(RadiatorSchema).format("csv").option("sep",",").option("header","true"
).load("/home/ubuntu/Desktop/Thesis/Radiator.csv")
RadiatorDataFrame.createOrReplaceTempView("RadiatorTable")
#RadiatorDataFrame.show()

#USer_Radiator#
User_RadiatorSchema = StructType([
    StructField("created_at",StringType(),True),
    StructField("entry_id",StringType(),True),
    StructField("Temp_IN",StringType(),True),
    StructField("Hum_IN",StringType(),True),
    StructField("Luminosity",StringType(),True),
    StructField("Noise",StringType(),True),
    StructField("Temp_OUT",StringType(),True),
    StructField("Hum_OUT",StringType(),True),
    StructField("Radiator",StringType(),True),
    StructField("User_Radiator",StringType(),True),
    StructField("latitude",StringType(),True),
    StructField("longitude",StringType(),True),
    StructField("elevation",StringType(),True),
    StructField("status",StringType(),True)
  ])

User_RadiatorDataFrame =
spark.read.schema(User_RadiatorSchema).format("csv").option("sep",",").option("header","
true").load("/home/ubuntu/Desktop/Thesis/feeds.csv")
User_RadiatorDataFrame.createOrReplaceTempView("User_RadiatorTable")
#User_RadiatorDataFrame.show()


#Example#
ExampleSchema = StructType([
    StructField("created_at",StringType(),True),
    StructField("entry_id",StringType(),True),
    StructField("Temp_IN",StringType(),True),
    StructField("Hum_IN",StringType(),True),
    StructField("Luminosity",StringType(),True),
    StructField("Noise",StringType(),True),
    StructField("Temp_OUT",StringType(),True),
    StructField("Hum_OUT",StringType(),True),
    StructField("Radiator",StringType(),True),
    StructField("User_Radiator",StringType(),True),
    StructField("latitude",StringType(),True),
    StructField("longitude",StringType(),True),
    StructField("elevation",StringType(),True),
    StructField("status",StringType(),True)
  ])

ExampleDataFrame =
spark.read.schema(ExampleSchema).format("csv").option("sep",",").option("header","true")
.load("/home/ubuntu/Desktop/Thesis/example/feed.csv")
ExampleDataFrame.createOrReplaceTempView("ExampleTable")
ExampleDataFrame.show()


##Query For Example DataSet--All Tables##
groupByExampleDataSet = spark.sql("SELECT Ti.entry_id as ID, Ti.Temp_IN, To.Temp_OUT,
Hi.Hum_IN, Ho.Hum_OUT, R.Radiator FROM Temp_INTable as Ti, Temp_OUTTable as To,
Hum_INTable as Hi, Hum_OUTTable as Ho, RadiatorTable as R WHERE Ti.entry_id =
To.entry_id AND Ti.entry_id = R.entry_id AND Ti.entry_id = R.entry_id AND Ti.entry_id =
```

```
Ho.entry_id AND Ti.entry_id = Hi.entry_id  GROUP BY Ti.entry_id, Ti.Temp_IN,
To.Temp_OUT, Hi.Hum_IN, Ho.Hum_OUT, R.Radiator ORDER BY ID")
groupByExampleDataSet.show()
groupByExam-
ple-
Da-
ta-
Set.coalesce(1).write.option("header","true").csv('/home/ubuntu/Desktop/Thesis/Results/E
xampleDataSet')
os.system('mv /home/ubuntu/Desktop/Thesis/Results/ExampleDataSet/*.csv
/home/ubuntu/Desktop/Thesis/Results/ExampleDataSet/ExampleDataSet.csv')

##Query For USER OVERRIDE##
groupByUSERSet = spark.sql("SELECT entry_id as ID, Temp_IN, Temp_OUT, Hum_IN, Hum_OUT,
User_Radiator FROM User_RadiatorTable WHERE User_Radiator = 1 GROUP BY entry_id,
Temp_IN, Temp_OUT, Hum_IN, Hum_OUT, User_Radiator ORDER BY ID")
groupByUSERSet.show()
groupByUSER-
Set.coalesce(1).write.option("header","true").csv('/home/ubuntu/Desktop/Thesis/Results/U
SER')
os.system('mv /home/ubuntu/Desktop/Thesis/Results/USER/*.csv
/home/ubuntu/Desktop/Thesis/Results/USER/USER.csv')

##Query For TrainingSet##
groupByTrainingSet = spark.sql("SELECT Ti.entry_id as ID, Ti.Temp_IN, To.Temp_OUT,
Hi.Hum_IN, Ho.Hum_OUT, R.Radiator FROM Temp_INTable as Ti, Temp_OUTTable as To,
Hum_INTable as Hi, Hum_OUTTable as Ho, RadiatorTable as R WHERE Ti.entry_id =
To.entry_id AND Ti.entry_id = R.entry_id AND Ti.entry_id = R.entry_id AND Ti.entry_id =
Ho.entry_id AND Ti.entry_id = Hi.entry_id AND Radiator = 1 GROUP BY Ti.entry_id,
Ti.Temp_IN, To.Temp_OUT, Hi.Hum_IN, Ho.Hum_OUT, R.Radiator ORDER BY ID")
groupByTrainingSet.show()
groupByTrainingS-
et.coalesce(1).write.option("header","true").csv('/home/ubuntu/Desktop/Thesis/Results/Tr
ainingSet')
os.system('mv /home/ubuntu/Desktop/Thesis/Results/TrainingSet/*.csv
/home/ubuntu/Desktop/Thesis/Results/TrainingSet/TrainingSet.csv')

##Query For Linear_Regression##
groupByClassification = spark.sql("SELECT Temp_IN, Radiator, Ti.entry_id as ID FROM
Temp_INTable as Ti, RadiatorTable as R WHERE Ti.entry_id = R.entry_id GROUP BY Temp_IN,
Radiator, Ti.entry_id ORDER BY ID")
groupByClassification.show()
groupByClassifica-
tion.coalesce(1).write.option("header","true").csv('/home/ubuntu/Desktop/Thesis/Results/
REGRESSION')
os.system('mv /home/ubuntu/Desktop/Thesis/Results/REGRESSION/*.csv
/home/ubuntu/Desktop/Thesis/Results/REGRESSION/REGRESSION.csv')


##Query For ExampleSet--One Table##
groupByExampleSet = spark.sql("SELECT entry_id as ID, Temp_IN, Temp_OUT, Hum_IN, Hum_OUT
FROM ExampleTable WHERE GROUP BY entry_id, Temp_IN, Temp_OUT, Hum_IN, Hum_OUT ORDER BY
ID")
groupByExampleSet.show()
groupByExample-
Set.coalesce(1).write.option("header","true").csv('/home/ubuntu/Desktop/Thesis/Results/E
xampleSet')
os.system('mv /home/ubuntu/Desktop/Thesis/Results/ExampleSet/*.csv
/home/ubuntu/Desktop/Thesis/Results/ExampleSet/ExampleSet.csv')
spark.stop()
```

Κώδικας Αλγορίθμου Ταξινόμησης

```
import pandas as pd
import matplotlib.pyplot as plt
import pylab as pl
from pandas.plotting import scatter_matrix
from matplotlib import cm
stats =
pd.read_csv('/home/ubuntu/Desktop/Thesis/Results/ExampleDataSet/ExampleDataSet.csv')
stats.head()
print(stats.shape)
stats.drop('ID', axis=1).plot(kind='box', subplots=True, layout=(6,6), sharex=False,
sharey=False, figsize=(19,19), title='Box Plot for each input variable')
plt.savefig('/home/ubuntu/Desktop/Thesis/Figures/stats_box')
#plt.show()

stats.drop('ID' ,axis=1).hist(bins=30, figsize=(9,9))
pl.suptitle("Histogram for each numeric input variable")
plt.savefig('/home/ubuntu/Desktop/Thesis/Figures/Classification_stats_histogram')
plt.show()
```

Κώδικας Αλγορίθμου Παλινδρόμησης

```
import numpy as np
import matplotlib.pyplot as plt  # To visualize
import pandas as pd  # To read data
from sklearn.linear_model import LinearRegression


data = pd.read_csv('/home/ubuntu/Desktop/Thesis/Results/REGRESSION/REGRESSION.csv')  #
load data set
X = data.iloc[:, 0].values.reshape(-1, 1)  # values converts it into a numpy array
Y = data.iloc[:, 1].values.reshape(-1, 1)  # -1 means that calculate the dimension of
rows, but have 1 column
linear_regressor = LinearRegression()  # create object for the class
linear_regressor.fit(X, Y)  # perform linear regression
Y_pred = linear_regressor.predict(X)  # make predictions

plt.scatter(X, Y)
plt.plot(X, Y_pred, color='red')
plt.show()
plt.savefig('/home/ubuntu/Desktop/Thesis/Figures/Linear_Regression')
```

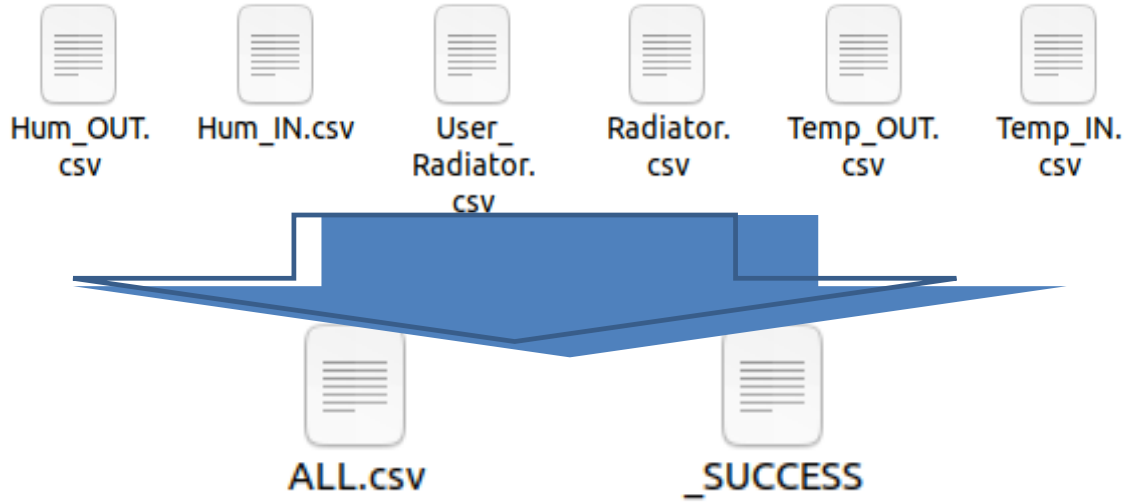
3.4 Αποτελέσματα

Τελική μορφή δεδομένων αφού προηγήθηκε η επεξεργασία τους από το Spark.

```
+-----+-------+--------+------+-------+--------+-------------+
|   ID|Temp_IN|Temp_OUT|Hum_IN|Hum_OUT|Radiator|User_Radiator|
+-----+-------+--------+------+-------+--------+-------------+
|95073|   20.5|   10.69|  35.0|   77.0|       1|            0|
|95074|   20.5|   10.04|  35.0|   80.0|       1|            0|
|95075|   21.3|    9.55|  34.0|   81.0|       0|            0|
|95076|    nan|       9|  34.0|   83.0|       0|            0|
|95077|   21.3|    8.67|  34.0|   84.0|       0|            0|
|95078|   21.3|    8.21|  34.0|   86.0|       0|            0|
|95079|   20.5|     7.8|  35.0|   87.0|       1|            0|
|95080|   21.3|    7.47|  34.0|   88.0|       0|            0|
|95081|   21.5|    7.16|  34.0|   89.0|       0|            0|
|95082|   21.5|    6.85|  34.0|   89.0|       0|            0|
|95083|   21.0|    6.54|  35.0|   89.0|       0|            0|
|95084|   20.8|    6.21|  35.0|   90.0|       0|            0|
|95085|   20.5|    5.92|  35.0|   90.0|       1|            0|
|95086|   21.3|    5.76|  34.0|   90.0|       0|            0|
|95087|   21.5|    5.53|  34.0|   91.0|       0|            0|
|95088|   21.5|    5.29|  34.0|   91.0|       0|            0|
|95089|   21.0|    4.99|  34.0|   92.0|       0|            0|
|95090|   20.8|    4.63|  35.0|   93.0|       0|            0|
|95091|   20.5|    4.18|  35.0|   94.0|       1|            0|
|95092|   21.5|     5.1|  34.0|   92.0|       0|            0|
+-----+-------+--------+------+-------+--------+-------------+
only showing top 20 rows
```

Εικόνα 16. Spark Dataframe (20 top rows only)

Συγχώνευση όλων των αρχείων σε ένα, δηλαδή στο ALL.csv (το αρχείο _SUCCESS υποδηλώνει πως η διαδικασία ολοκληρώθηκε επιτυχώς)



Εικόνα 17. Τελικό αρχείο Δεδομένων ALL.csv

Εκτέλεση του κώδικα Ταξινόμησης (Classification), Decision Tree.

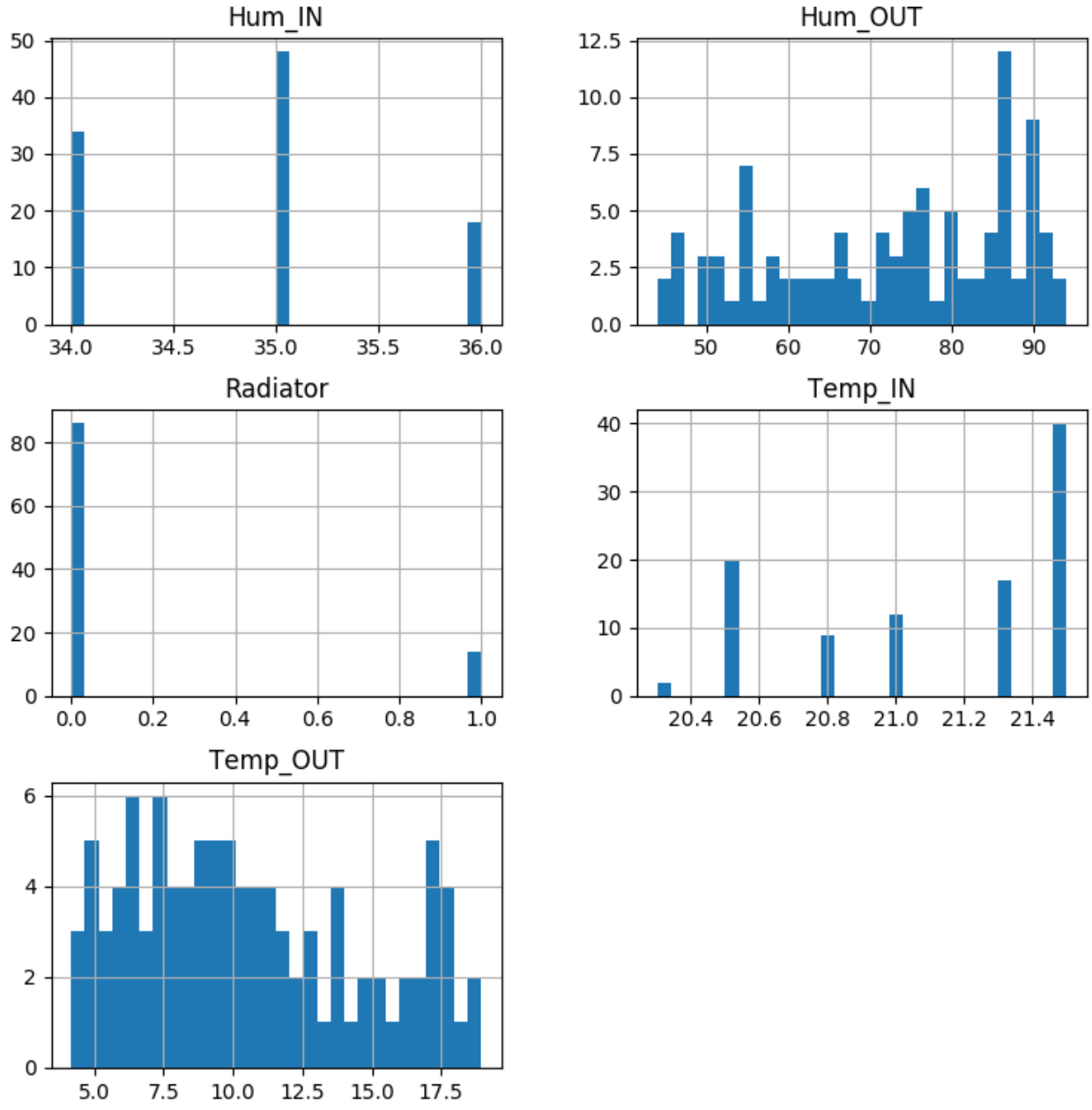


Figure 1. Histogram

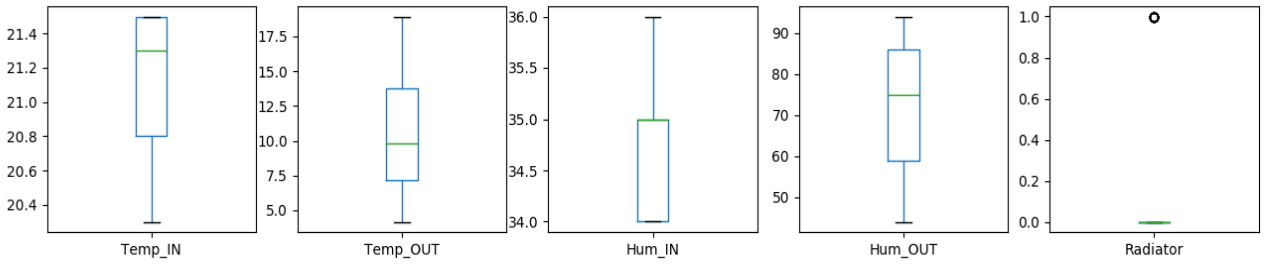


Figure 2. Box Plot

Εκτέλεση του κώδικα Παλινδρόμησης (Linear Regression)

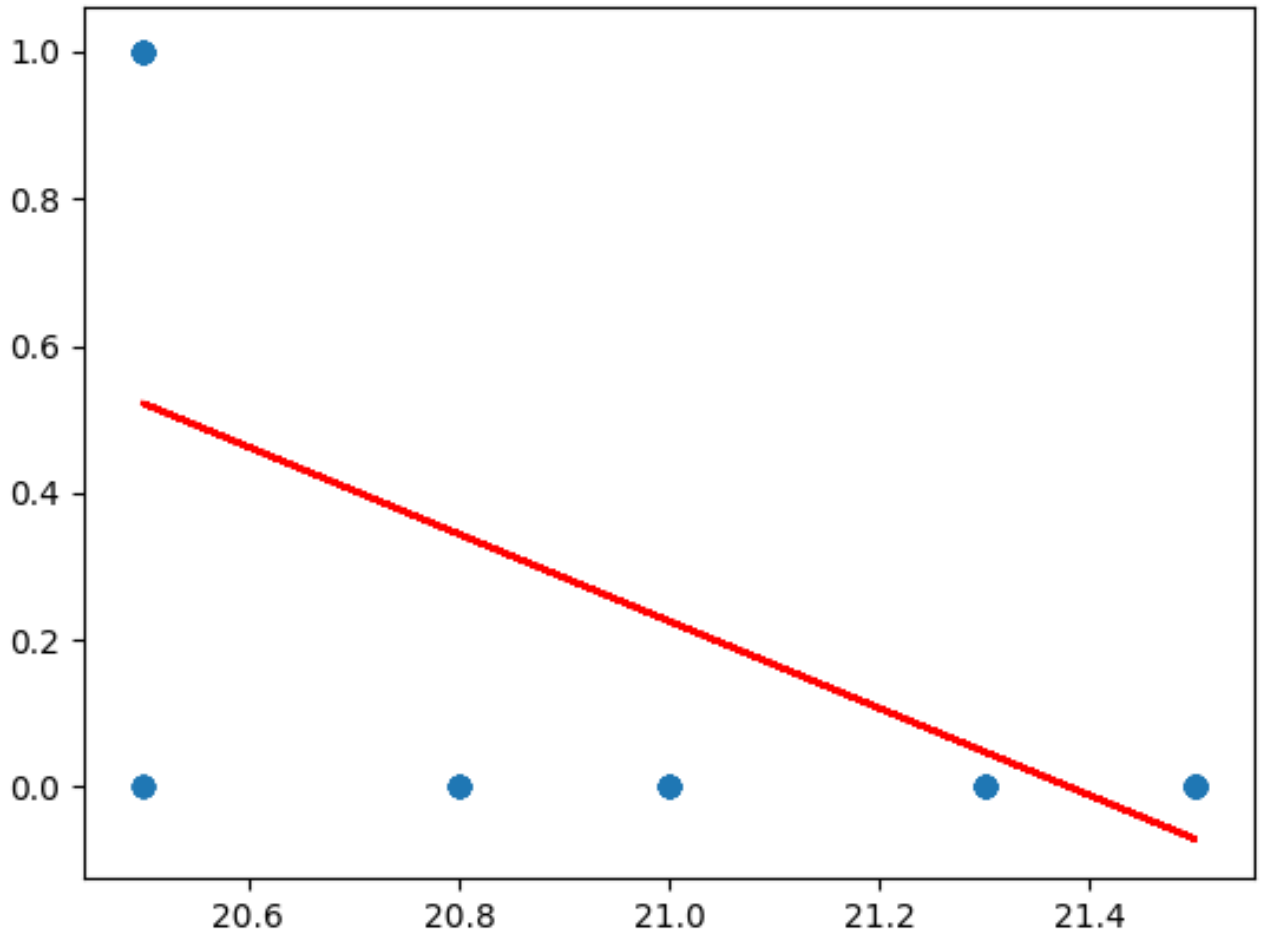


Figure 3. X=Temp_IN, Y=Radiator(0-1)

Ανάπτυξη Μοντέλων στο RapidMiner Studio

Ταξινόμηση (Classification)

Ανάπτυξη μοντέλου Ταξινόμησης

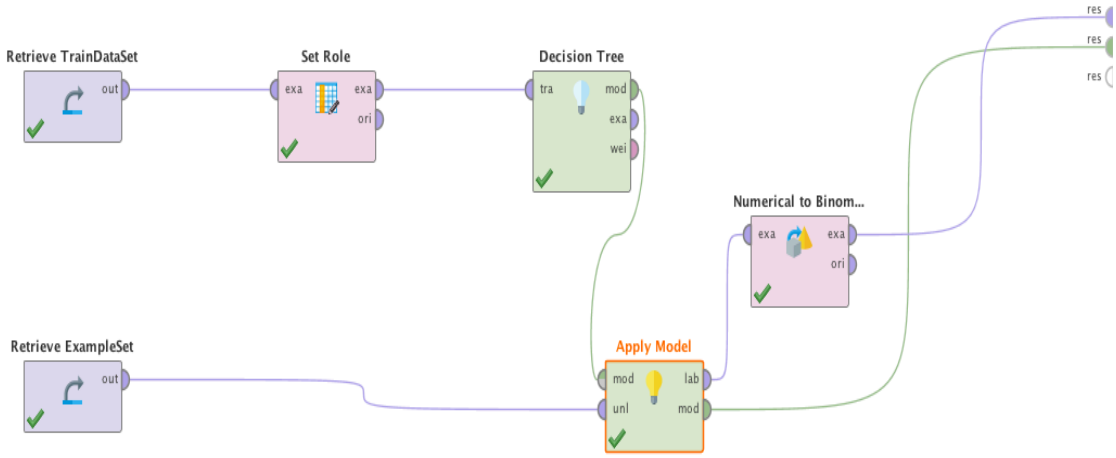


Figure 4. RapidMiner Studio model for Decision Tree

Row No.	ID	Temp_IN	Temp_OUT	Hum_IN	Hum_OUT
1	95155	21	10.710	35	73
2	95156	20.500	10.440	36	73
3	95157	20.800	10.220	36	73
4	95158	21.500	9.950	35	74
5	95159	21.300	9.720	35	74
6	95160	21.500	9.460	35	75
7	95161	21.500	9.140	35	76
8	95162	21.500	8.880	35	76
9	95163	20.500	8.600	36	77
10	95164	21	8.230	35	78
11	95165	21.500	7.710	35	79
12	95166	21	7.220	35	80
13	95167	20.800	16.280	35	54
14	95168	20.500	17.400	35	52
15	95169	20.500	17.180	35	55
16	95170	20.500	17.510	35	54
17	95171	20.500	17.540	35	55
18	95172	20.500	17.350	35	58
19	95173	20.500	15.880	35	64
20	95174	20.500	15.790	35	63
21	95175	20.500	14.910	35	64
22	95176	20.500	14.160	35	66
23	95177	20.500	13.650	35	68
24	95178	20.500	13.150	35	69
25	95179	20.500	13.180	35	68
26	95180	20.500	12.900	35	70
27	95181	20.500	12.650	35	70
28	95182	20.500	12.350	35	71

...	prediction(Radiator)	ID	Temp_IN	Temp_OUT	Hum_IN	Hu...
1	false	95155	21	10.710	35	73
2	true	95156	20.500	10.440	36	73
3	false	95157	20.800	10.220	36	73
4	false	95158	21.500	9.950	35	74
5	false	95159	21.300	9.720	35	74
6	false	95160	21.500	9.460	35	75
7	false	95161	21.500	9.140	35	76
8	false	95162	21.500	8.880	35	76
9	true	95163	20.500	8.600	36	77
10	false	95164	21	8.230	35	78
11	false	95165	21.500	7.710	35	79
12	false	95166	21	7.220	35	80
13	false	95167	20.800	16.280	35	54
14	false	95168	20.500	17.400	35	52
15	false	95169	20.500	17.180	35	55
16	false	95170	20.500	17.510	35	54
17	false	95171	20.500	17.540	35	55
18	false	95172	20.500	17.350	35	58
19	true	95173	20.500	15.880	35	64
20	true	95174	20.500	15.790	35	63
21	true	95175	20.500	14.910	35	64
22	true	95176	20.500	14.160	35	66
23	true	95177	20.500	13.650	35	68
24	true	95178	20.500	13.150	35	69
25	true	95179	20.500	13.180	35	68
26	true	95180	20.500	12.900	35	70
27	true	95181	20.500	12.650	35	70
28	true	95182	20.500	12.350	35	71

Figure 5. Left Unlabeled Data, Right Prediction Data for ‘Radiator’

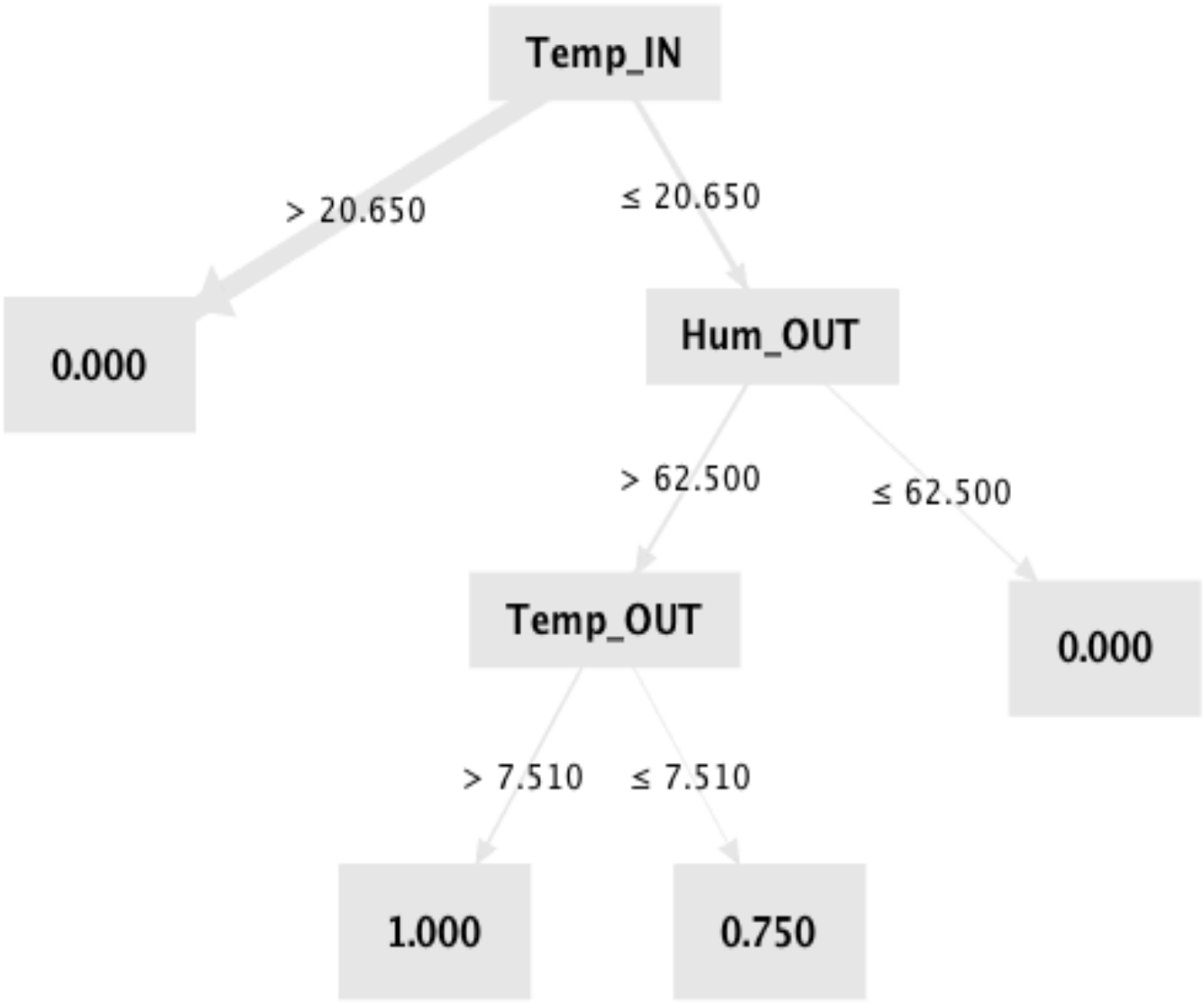


Figure 6. Decision Tree Model

```
If Temp_IN <= 20,.650 and Hum_OUT > 62.50 and Temp_OUT < 7,5
   grovepi.digitalWrite(relay,1) #Radiator ON
else
   grovepi.digitalWrite(relay,0) #Radiator OFF
```

Παλινδρόμηση (Regression)

Ανάπτυξη μοντέλου Παλινδρόμησης

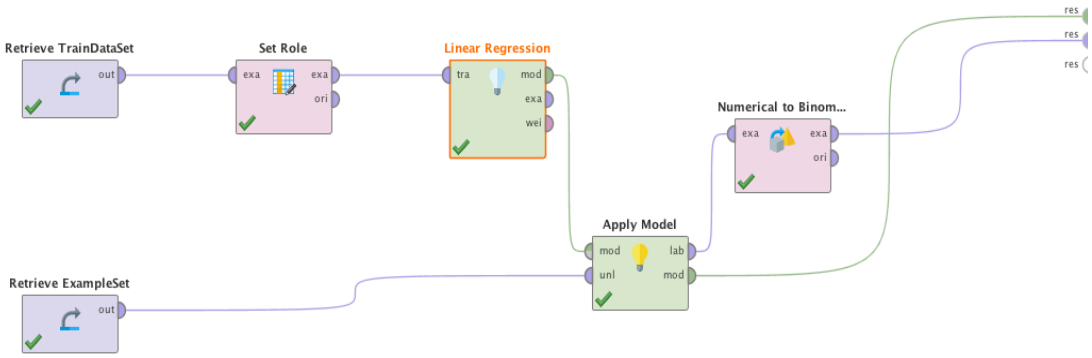


Figure 7. RapidMiner Studio model for Regression

Attribute	Coefficient	Std. Error	Std. Coefficient	Tolerance	t-Stat	p-Value	Code
ID	-0.003	0.001	-0.271	0.989	-3.042	0.003	***
Temp_IN	-0.595	0.067	-0.696	0.998	-8.822	0.000	****
Temp_OUT	-0.063	0.029	-0.748	0.953	-2.198	0.030	**
Hum_OUT	-0.016	0.008	-0.671	0.896	-1.902	0.060	*
(Intercept)	324.143	102.527	?	?	3.162	0.002	***

Figure 8. Linear Regression

Figure 9. Linear Regression - Scatter Model

3.5 Αξιολόγηση και Ανασκόπηση Αποτελεσμάτων

Τα αποτελέσματα που έχουμε συλλέξει έπειτα από την υλοποίηση της Μηχανικής Μάθησης στο σύστημά μας, αποδεικνύουν δύο βασικά στοιχεία: Αρχικά, ο πρώτος αλγόριθμος λειτουργίας του συστήματος καθίσταται εκ των υστέρων ‘άχρηστος’ και αυτό για τον λόγο ότι πλέον το σύστημα δεν λαμβάνει υπόψιν μόνο την εσωτερική θερμοκρασία για την λειτουργία του αλλά όλες τις παραμέτρους που έχει ως εισροή. Επίσης, παρατηρούμε πως αν και το σύνολο των δεδομένων με τις ενέργειες του χρήστη (ExampleDataSet) είναι μικρό σε όγκο, οι αλγόριθμοι Μηχανικής Μάθησης προέβλεψαν σωστά τα αποτελέσματα με τα αντίστοιχα δεδομένα εισροής.

Ενδεικτικά παρουσιάζουμε κομμάτι του κώδικα πριν και μετά την υλοποίηση της Μηχανικής Μάθησης στο Σύστημά μας.

Ο Κώδικας του συστήματος ΠΡΙΝ την υλοποίηση της Μηχανικής Μάθησης

```
#---------------Let's Begin------------
now = datetime.datetime.utcnow()
sensor = 8
blue = 0
#radiator relay
relay = 7
grovepi.pinMode(relay,"OUTPUT")
#red led
led1 = 2
grovepi.pinMode(led1,"OUTPUT")
grovepi.pinMode(led2,"OUTPUT")
setRGB(0,128,64)
setRGB(0,128,0)
setText("----Welcome----" "\n" "Loading......")
time.sleep(3)
while True :
      clientID = ''
      # Create a random clientID.
        for x in range(1,16):
            clientID+=random.choice(string.alphanum)
        try:

      ts = time.time()
                st = datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-%d %H:%M:%S')
                now = datetime.datetime.utcnow()
               print ('TIMESTAMP:', st)

                #---------------FORECASTIOPY API------------------
                setText("--FORECASTIOPY--"  "\n"  "-Weather Report-" )
                time.sleep(1)
                apikey = "90d3e695cf5f55650c4ffa88d012ca70"
                Larisa = [39.640984, 22.421268]
                fio = ForecastIO.ForecastIO(apikey,
                        units=ForecastIO.ForecastIO.UNITS_SI,
                        lang=ForecastIO.ForecastIO.LANG_ENGLISH,
                        latitude=Larisa[0], longitude=Larisa[1])
```

```
                print('Latitude', fio.latitude, 'Longitude', fio.longitude)
                print('Timezone', fio.timezone, 'Offset', fio.offset)
                print(fio.get_url()) # You might want to see the request url
                  #----------Get current weather data from darksky.net
                if fio.has_currently() is True:
                        currently = FIOCurrently.FIOCurrently(fio)
                        print('Currently')
                        for item in currently.get().keys():
                                print(item + ' : ' + unicode(currently.get()[item]))
                                # Or access attributes directly
                        print("Current Temperature:", "%.1f" %(currently.temperature))
                        print("Current Humidity:", "%.1f"  %(  currently.humidity * 100)
)
           print("Current WindSpeed:", "%.2f" % (currently.windSpeed))
           print("Current CloudCover:", "%.3f" % (currently.cloudCover))
           print("Current precipIntensity:", "%.1f" % (currently.precipIntensity))
           print("Current precipProbality:", "%.1f" % (currently.precipProbability))
           print("Current Summary:" , currently.summary)
                else:
                        print('No Currently data')
        #----some tests to see output for eather condition
        if currently.cloudCover == 1:
           print("It's gonna Rain!!")
        elif currently.cloudCover !=0 and currently.summary == "Clear":
           print("Good Weather!!")
        else:
           print("OK weather")
        setText('Weather data' '\n'  'loaded')
        time.sleep(4)
                setText("Temperature:" + str("%.1f" %(currently.temperature)) + '\n'
"Humidity:" + str("%.1f" %(float(currently.humidity * 100 )))  + "%")
                time.sleep(21)
        time.sleep(4)
                grovepi.digitalWrite(led2,1)
                # unit initialization
                sound=int(grovepi.analogRead(1)/10.24)
                s = str(int(grovepi.analogRead(1)/10.24))
                print ("sound", s)
        time.sleep(5)
                light=int(grovepi.analogRead(2)/10.24)
                l = str(int(grovepi.analogRead(2)/10.24))
                print ("light", l)
                setRGB(0,128,64)
                setRGB(0,128,0)
                setText("Loading.." +"\n" + "IP:" + get_ip())
                [tempTest,humidity] = grovepi.dht(sensor,blue)
                print ("tempTest", tempTest )
                time.sleep(15)
                [temp1,humidity] = grovepi.dht(sensor,blue)
                print ("temp1", temp1 - 1.5)
                time.sleep(18)
                [temp2,humidity] = grovepi.dht(sensor,blue)
                print ("temp2", temp2 - 1.5)
                time.sleep(15)
                [temp3,humidity] = grovepi.dht(sensor,blue)
                print ("temp3", temp3 - 1.5)
                [temp4,humidity] = grovepi.dht(sensor,blue)
                time.sleep(15)
                print ("temp4", temp4 - 1.5)
                time.sleep(6)
                temp = round(((temp1-1.5)+(temp2-1.5)+(temp3-1.5)+(temp4-1.5))/4,1)
```

```
                print ("temp", temp , "humidity" , humidity)
                if humidity > 29 and humidity < 45 :
                        if temp < 19.5  and currently.temperature > 12.5 :
                                grovepi.digitalWrite(relay,1)
                                grovepi.digitalWrite(led1,1)
                                radiator = "ON"
                                time.sleep(20)
                                grovepi.digitalWrite(relay,1)
                                value  = 1    #radiator
                                value2 = 0    #radiatorRemote
                                allDa-
ta="field1="+str(temp)+"&field2="+str(humidity)+"&field3="+str(l)+"&field4="+str(s)+"&fi
eld5="+str(currently.temperature)+"&field6="+str("%.1f"%(float(currently.humidity*100)))
+"&field7=" + str(value)+"&field8=" + str(value2)
                                publish.single(topic, payload=allData, host-
name=mqttHost, transport=tTransport,
port=tPort,auth={'username':mqttUsername,'password':mqttAPIKey})
                                print (radiator,st)
                                print("Current Temperature:",  currently.temperature)
                                print("Current Humidity:", currently.humidity * 100  )
                                print ("temp", temp)
                                print ("Humidity", humidity )
                                setRGB(0,128,64)
                                setRGB(0,128,0)
                                setText("Temperature:" + str(temp) + '\n' "Humidity:" +
str(int(humidity)) + "%")
                                print("case 1")
                                time.sleep(1800)
            timeCounter = 1800
            contents = main()
            Write(timeCounter,contents)
                else:
                        print("Humidity in Dangerous Levels!!!")
                        setRGB(0,128,64)
                        setRGB(0,128,0)
                        setText("---ATTENTION---" '\n' + str(temp) + str((inthumidity))
)
                        time.sleep(3600)
        except KeyboardInterrupt:
                grovepi.digitalWrite(relay,0)
                grovepi.digitalWrite(led1,0)
                radiator = "OFF"
                time.sleep(2)
                grovepi.digitalWrite(relay,0)
                print ("  except keyboard_interrupt "  + radiator)
                setRGB(0,128,64)
                setRGB(0,128,0)
                setText("except keyboard_interrupt")
                break
        except:
                grovepi.digitalWrite(led1,0)
                grovepi.digitalWrite(relay,0)
                radiator = "OFF"
                time.sleep(20)
                grovepi.digitalWrite(relay,0)
                print (" except false" + radiator)
                setRGB(0,128,64)
                setRGB(0,128,0)
                setText("except error")
                os.system('systemctl restart relay.service')
                break
```

Ο Κώδικας του συστήματος ΜΕΤΑ την υλοποίηση της Μηχανικής Μάθησης

```
#---------------Let's Begin------------
now = datetime.datetime.utcnow()
sensor = 8
blue = 0
#radiator relay
relay = 7
grovepi.pinMode(relay,"OUTPUT")
#red led
led1 = 2
grovepi.pinMode(led1,"OUTPUT")
grovepi.pinMode(led2,"OUTPUT")
setRGB(0,128,64)
setRGB(0,128,0)
setText("----Welcome----" "\n" "Loading......")
time.sleep(3)
while True :
      clientID = ''
      # Create a random clientID.
        for x in range(1,16):
            clientID+=random.choice(string.alphanum)
        try:

      ts = time.time()
                st = datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-%d %H:%M:%S')
                now = datetime.datetime.utcnow()
               print ('TIMESTAMP:', st)

                #---------------FORECASTIOPY API-----------------
                setText("--FORECASTIOPY--"  "\n"  "-Weather Report-" )
                time.sleep(1)
                apikey = "90d3e695cf5f55650c4ffa88d012ca70"
                Larisa = [39.640984, 22.421268]
                fio = ForecastIO.ForecastIO(apikey,
                        units=ForecastIO.ForecastIO.UNITS_SI,
                        lang=ForecastIO.ForecastIO.LANG_ENGLISH,
                        latitude=Larisa[0], longitude=Larisa[1])
                print('Latitude', fio.latitude, 'Longitude', fio.longitude)
                print('Timezone', fio.timezone, 'Offset', fio.offset)
                print(fio.get_url()) # You might want to see the request url
                  #----------Get current weather data from darksky.net
                if fio.has_currently() is True:
                        currently = FIOCurrently.FIOCurrently(fio)
                        print('Currently')
                        for item in currently.get().keys():
                                print(item + ' : ' + unicode(currently.get()[item]))
                                # Or access attributes directly
                        print("Current Temperature:", "%.1f" %(currently.temperature))
                        print("Current Humidity:", "%.1f"  %(  currently.humidity * 100)
)
         print("Current WindSpeed:", "%.2f" % (currently.windSpeed))
         print("Current CloudCover:", "%.3f" % (currently.cloudCover))
         print("Current precipIntensity:", "%.1f" % (currently.precipIntensity))
         print("Current precipProbality:", "%.1f" % (currently.precipProbability))
         print("Current Summary:" , currently.summary)
                else:
                        print('No Currently data')
      setText('Weather data' '\n'  'loaded')
```

```
      time.sleep(4)
                setText("Temperature:" + str("%.1f" %(currently.temperature)) + '\n'
"Humidity:" + str("%.1f" %(float(currently.humidity * 100 )))  + "%")
                time.sleep(21)
      time.sleep(4)
                grovepi.digitalWrite(led2,1)
                # unit initialization
                sound=int(grovepi.analogRead(1)/10.24)
                s = str(int(grovepi.analogRead(1)/10.24))
                print ("sound", s)
      time.sleep(5)
                light=int(grovepi.analogRead(2)/10.24)
                l = str(int(grovepi.analogRead(2)/10.24))
                print ("light", l)
                setRGB(0,128,64)
                setRGB(0,128,0)
                setText("Loading.." +"\n" + "IP:" + get_ip())
                [tempTest,humidity] = grovepi.dht(sensor,blue)
                print ("tempTest", tempTest )
                time.sleep(15)
                [temp1,humidity] = grovepi.dht(sensor,blue)
                print ("temp1", temp1 - 1.5)
                time.sleep(18)
                [temp2,humidity] = grovepi.dht(sensor,blue)
                print ("temp2", temp2 - 1.5)
                time.sleep(15)
                [temp3,humidity] = grovepi.dht(sensor,blue)
                print ("temp3", temp3 - 1.5)
                [temp4,humidity] = grovepi.dht(sensor,blue)
                time.sleep(15)
                print ("temp4", temp4 - 1.5)
                time.sleep(6)
                temp = round(((temp1-1.5)+(temp2-1.5)+(temp3-1.5)+(temp4-1.5))/4,1)
                print ("temp", temp , "humidity" , humidity)

If Temp_IN <= 20,.650 and Hum_OUT > 62.50 and Temp_OUT < 7,5
   grovepi.digitalWrite(relay,1) #Radiator ON
else
   grovepi.digitalWrite(relay,0) #Radiator OFF
```

3.6 Βελτιστοποίηση και Καινοτομία

Επόμενος στόχος είναι η ενσωμάτωση του συστήματος πρόβλεψης εξωτερικών καιρικών συνθηκών στο ήδη υπάρχον σύστημά μας. Πιο συγκεκριμένα, ανάπτυξη συστήματος το οποίο θα λαμβάνει ως δεδομένα εισροής τις εσωτερικές και εξωτερικές συνθήκες αλλά και την μεταβολή των εξωτερικών συνθηκών έως ένα μεταγενέστερο χρονικό διάστημα. Ακολούθως, και αφού πρώτα έχει γίνει ικανοποιητική συλλογή δεδομένων ώστε να είναι εφικτή η υλοποίηση της Μηχανικής Μάθησης, θα επανεξετάσουμε και θα διορθώσουμε το σύστημα βάση των αποτελεσμάτων της Μηχανικής Μάθησης. Τέλος, θα αναπτυχθεί κώδικας ο οποίος θα καταγράφει τις φορές όπου ο χρήστης παράκαμψε το σύστημα και θα επαναλειτουργεί εκ νέου με βάση τις νέες προτιμήσεις του χρήστη.

3.7 Συμπέρασμα

Είναι ασφαλές να πούμε πως ανάλογα με την περίπτωση που εξετάζουμε κάθε φορά, η Μηχανική Μάθηση μπορεί να αναπτυχθεί και να υλοποιηθεί εφόσον υπάρχει μεγάλος όγκος και ποικιλία δεδομένων προς επεξεργασία. Όσο αυξάνεται το 'DataSet' μας σε όγκο τόσο καλύτερα λειτουργεί και η Μηχανική Μάθηση, καθώς μην ξεχνάμε πως οι προβλέψεις της ΜΜ προκύπτουν από παλαιότερες μετρήσεις-καταγραφές δεδομένων. Στην περίπτωσή μας, η Μηχανική Μάθηση άλλαξε άρδην τον τρόπο λειτουργίας του συστήματος καθώς πλέον λαμβάνονται υπόψιν 8 (οχτώ) μεταβλητές ενώ πρώτα μόνο 2 (δύο).

Για παράδειγμα:

- Έστω επιθυμητές εσωτερικές συνθήκες 21 βαθμοί κελσίου και έως 45% υγρασία
- Έστω εξωτερικές συνθήκες 11 βαθμοί κελσίου και έως 75% υγρασία
- Χρονικό διάστημα μέγιστης ισχύος, ambient_time=15 λεπτά
- Πρόγνωση μεταβολής εξωτερικών συνθηκών για τις επόμενες 3 (τρεις) ώρες, Θερμοκρασία 9.6 - 11.4 βαθμοί κελσίου και 75% - 90% υγρασία
- Το σύστημα θα λειτουργήσει με βάσει όλες αυτές τι παραμέτρους με σκοπό ναι μεν οι εσωτερικές συνθήκες να 'πλησιάζουν' στις προτιμήσεις του χρήστη αλλά και να γίνει η επίτευξη αυτών με την ελάχιστη δυνατή κατανάλωση

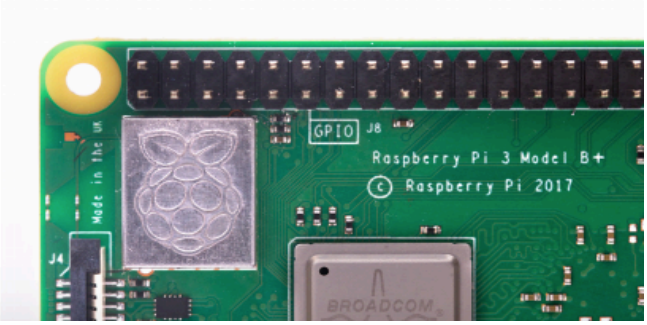
Άρα τελικά, εάν το σύστημα στην πρώτη του έκδοση θα λειτουργούσε για μια ώρα (60 λεπτά) 'fixed time' και έπειτα θα πραγματοποιούσε επανέλεγχο και αξιολόγηση των δεδομένων, πλέον ο υπολογισμός του χρόνου λειτουργίας γίνεται δυναμικά χωρίς να χρειάζονται διακοπές και παύσεις στο σύστημα

4 Λεπτομέρειες Υλικού (Hardware Specifications and Datasheets)

4.1.1 Raspberry Pi 3 model b

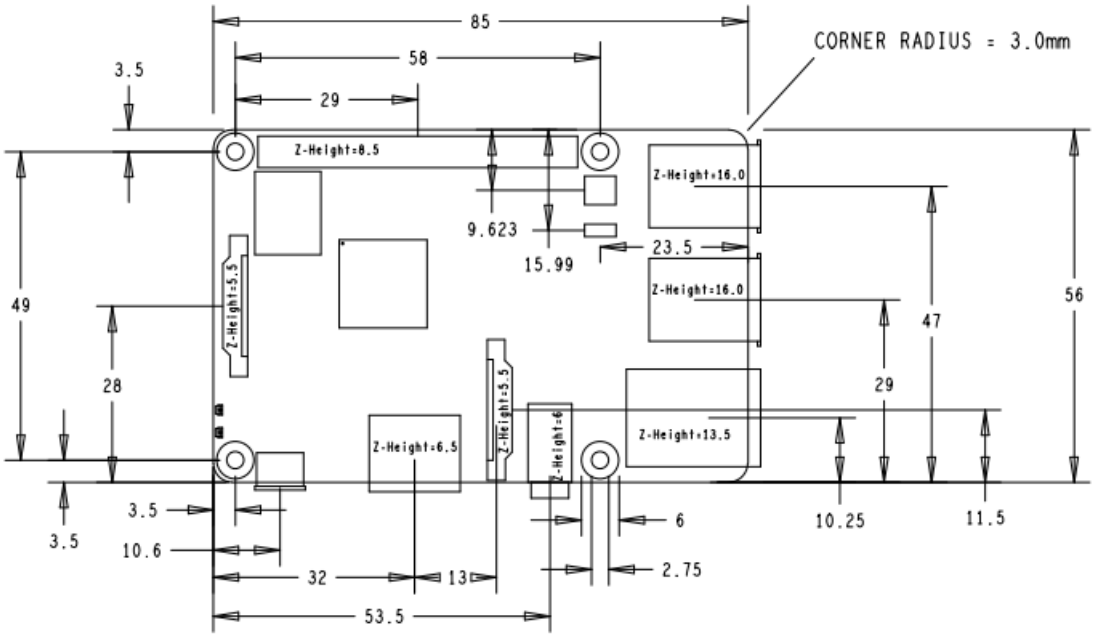
Processor:	Broadcom BCM2837B0, Cortex-A53 64-bit SoC @ 1.4 GHz
Memory:	1GB LPDDR2 SDRAM
Connectivity:	■ 2.4 GHz and 5 GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2, BLE ■ Gigabit Ethernet over USB 2.0 (maximum throughput 300 Mbps) ■ 4 × USB 2.0 ports
Access:	Extended 40-pin GPIO header
Video & sound:	■ 1 × full size HDMI ■ MIPI DSI display port ■ MIPI CSI camera port ■ 4 pole stereo output and composite video port
Multimedia:	H.264, MPEG-4 decode (1080p30); H.264 encode (1080p30); OpenGL ES 1.1, 2.0 graphics
SD card support:	Micro SD format for loading operating system and data storage
Input power:	■ 5 V/2.5 A DC via micro USB connector ■ 5 V DC via GPIO header ■ Power over Ethernet (PoE)–enabled (requires separate PoE HAT)
Environment:	Operating temperature, 0–50 °C
Compliance:	For a full list of local and regional product approvals, please visit www.raspberrypi.org/products/raspberry-pi-3-model-b+
Production lifetime:	The Raspberry Pi 3 Model B+ will remain in production until at least January 2023.

raspberrypi.org



Εικόνα 18. Raspberry Pi 3 model b datasheet

Physical specifications



Warnings

- This product should only be connected to an external power supply rated at 5 V/2.5 A DC. Any external power supply used with the Raspberry Pi 3 Model B+ shall comply with relevant regulations and standards applicable in the country of intended use.
- This product should be operated in a well-ventilated environment and, if used inside a case, the case should not be covered.
- Whilst in use, this product should be placed on a stable, flat, non-conductive surface and should not be contacted by conductive items.
- The connection of incompatible devices to the GPIO connection may affect compliance, result in damage to the unit, and invalidate the warranty.
- All peripherals used with this product should comply with relevant standards for the country of use and be marked accordingly to ensure that safety and performance requirements are met. These articles include but are not limited to keyboards, monitors, and mice when used in conjunction with the Raspberry Pi.
- The cables and connectors of all peripherals used with this product must have adequate insulation so that relevant safety requirements are met.

Safety instructions

To avoid malfunction of or damage to this product, please observe the following:

- Do not expose to water or moisture, or place on a conductive surface whilst in operation.
- Do not expose to heat from any source; the Raspberry Pi 3 Model B+ is designed for reliable operation at normal ambient temperatures.
- Take care whilst handling to avoid mechanical or electrical damage to the printed circuit board and connectors.
- Whilst it is powered, avoid handling the printed circuit board, or only handle it by the edges to minimise the risk of electrostatic discharge damage.

raspberrypi.org

Εικόνα 19. Raspberry Pi 3 model b specifications

4.1.2 Grove Pi

Features

- 7 digital Ports
- 3 analoge Ports
- 3 I2C ports
- 1 Serial port connect to GrovePi
- 1 Serial port connect to Raspberry Pi
- Grove header Vcc output Voltage: 5Vdc

Εικόνα 20.GrovePi features

FUNCTIONAL BLOCK DIAGRAM

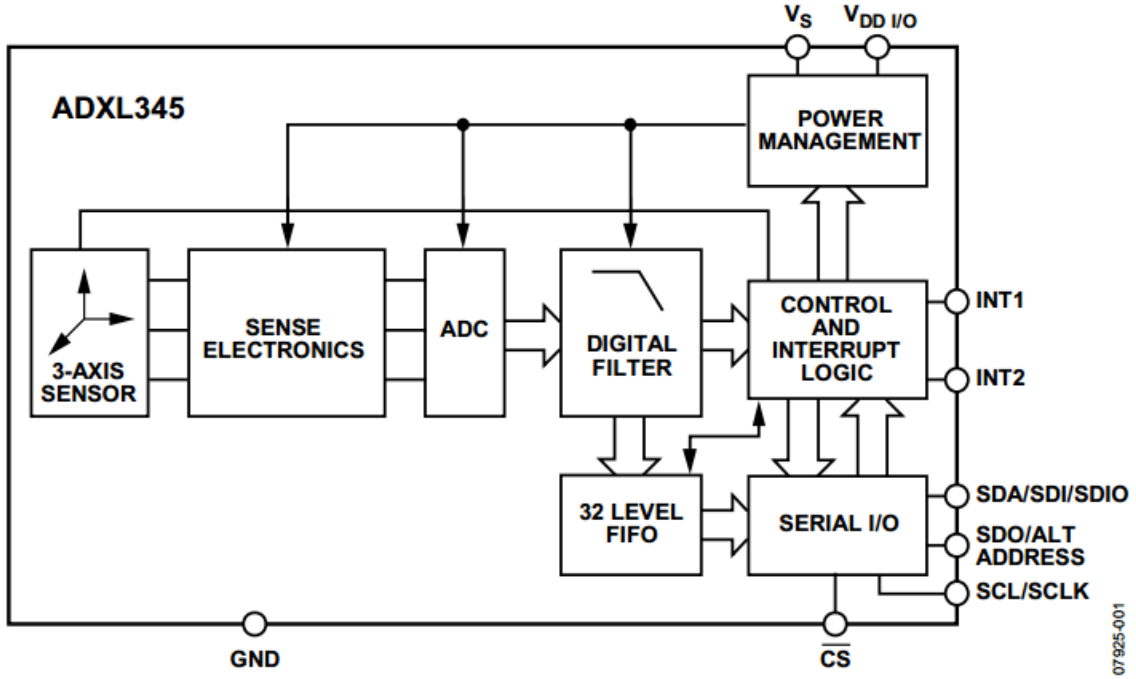


Figure 1.

Εικόνα 21.GrovePi Datasheet

5 Συμπεράσματα

Η αρχική ιδέα της πτυχιακής στην οποία και έχει βασιστεί η διπλωματική, είναι να μπορεί ο καθένας να μετατρέψει το σπίτι του σε ‘έξυπνο’. Τόσο το κόστος δημιουργίας, συγκριτικά με τα ολοκληρωμένα συστήματα που κυκλοφορούν στην αγορά, όσο και το επίπεδο δυσκολίας για την δημιουργία του, είναι χαμηλά με αποκλειστικό σκοπό να είναι δυνατόν να υλοποιηθεί και να αναπτυχθεί χωρίς να απαιτείται να έχει κάποιος εξιδεικευμένες γνώσεις επί του θέματος ή μεγάλο ‘μπάτζετ’. Στόχος της διπλωματικής εργασίας, είναι να αναβαθμιστεί το σύστημα κάνοντας χρήση Μηχανικής Μάθησης. Η αναβάθμιση αυτή μπορεί να έχει αντίκτυπο σε δύο τομείς: Την εξοικονόμηση ενέργειας και ταυτόχρονα μείωση του κόστους κατανάλωσης σε ευρώ, την πλήρη αυτοματοποίηση του συστήματος ώστε να μην απαιτείται καθόλου πλέον ο ανθρώπινος παράγοντας για την ορθή λειτουργία του. Έπειτα από την συνεχή βελτίωση του συστήματος, όπως αυτό φαίνεται και από τις συνεχόμενες εκδόσεις, μπορούμε να πούμε πως έχει αναπτυχθεί ένα ‘αυτόνομο’ σε λειτουργία και ‘έξυπνο’ αλγοριθμικά σύστημα.

6 Αντιμετώπιση Προβλημάτων (Troubleshooting)

Κατά την διάρκεια των τριών χρόνων που λειτουργεί το συγκεκριμένο σύστημα, προέκυψαν πολλά προβλήματα τόσο στη διαδικασία ομαλής λειτουργείας του συστήματος χωρίς ‘bugs’, όσο και στη διαδικασία ρύθμισης όλων των δωρεάν εφαρμογών(API’s) που χρησιμοποιούμε ώστε να συντονιστούν και να λειτουργούν κανονικά. Δεν υπάρχει κάτι συγκεκριμένο να προστεθεί ως λύση σε πρόβλημα καθώς το σύστημα δεν βρίσκεται πλέον σε δοκιμαστική(beta) χρήση και ως εκ τούτου είναι πλήρως λειτουργικό, ‘bug-free’. Ενδεικτικά και μόνο, το σημαντικότερο πρόβλημα που παρουσιάστηκε στο σύστημα είναι το εξής: αρχικά το σύστημα παίρνει όλα τα εξωτερικά δεδομένα(θερμοκρασία-υγρασία-ταχύτητα αέρα κτλ.) και έπειτα παίρνει τα εσωτερικά δεδομένα από τους αισθητήρες. Αφού γίνει αυτό και γίνουν και οι απαραίτητοι έλεγχοι και ενέργειες από τον κώδικά μας, πρέπει τα δεδομένα να ανέβουν στο ‘ThingSpeak’ για να μπορέσουμε να τα χρησιμοποιήσουμε. Εφόσον όμως χρησιμοποιούμε δωρεάν έκδοση, έχουμε όριο στον όγκο που κάνουμε upload. Οπότε αφού ανέβαινε η πρώτη τιμή των δεδομένων, θερμοκρασία, όλες οι υπόλοιπες περνούσαν κενές(null) στο σύστημα και αναμενόμενα δεν λειτουργούσε τίποτα. Η λύση μετά από πολύ πειραματισμό και δοκιμές βρέθηκε με έναν απλό μεν αλλά μπακάλικο τρόπο δε. Το upload των δεδομένων γίνεται με χρονικό όριο(time-interval) 28 δευτερολέπτων και το ‘ThingSpeak’ νομίζει πως περνάμε τα ίδια δεδομένα. Η λύση αυτή βρίσκεται στον κώδικα του κυρίως προγράμματος και περιγράφεται αναλυτικά με σχόλια πως λειτουργεί.

Βιβλιογραφία

[1]. API's, https://hellenictechnologies.com/ti-einai-to-apis-kai-pos-chrisimopoieitai/

[2]. Dark Sky documentation, https://darksky.net/help/website

[3]. ThingSpeak documentation, https://www.mathworks.com/help/thingspeak/

[4]. GrovePi homepage and documentation, https://www.dexterindustries.com/grovepi/

[5]. RaspberryPi homepage and documentation, https://www.raspberrypi.org/

[6]. NoiP documentation, https://www.noip.com/

[7]. Zapier documentation, https://zapier.com/apps

[8]. How to disable the web cam led, https://www.raspberrypi-spy.co.uk/2013/05/how-to-disable-the-red-led-on-the-pi-camera-module/

[9]. How to integrate Python with the Linux system services, https://www.raspberrypi-spy.co.uk/2015/10/how-to-autorun-a-python-script-on-boot-using-systemd/

[10]. How to grant access to php to execute shell services, https://unix.stackexchange.com/questions/115054/php-shell-exec-permission-on-linux-ubuntu/

[11]. Web cam setup using Motion, https://www.bouvet.no/bouvet-deler/utbrudd/building-a-motion-activated-security-camera-with-the-raspberry-pi-zero

[12]. Grovepi firmware update https://www.dexterindustries.com/GrovePi/get-started-with-the-grovepi/updating-firmware/)

[13]. ForecatioPy installation https://github.com/dvdme/forecastiopy

[14]. https://repository.kallipos.gr/bitstream/11419/3382/1/02_chapter_04.pdf

[15]. https://machinelearningmastery.com/types-of-classification-in-machine-learn-ing/#:~:text=In%20machine%20learning%2C%20classification%20refers,one%20of%20the%20known%20characters.

[16]. https://realpython.com/linear-regression-in-python/