ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ
ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

Εξόρυξη δεδομένων από Έξυπνες Συσκευές για πρόβλεψη χρήσης και συστάσεις

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Άγγελος Δ. Δανιήλ

Επιβλέπουσα : Ιωάννα Ρουσσάκη
Επίκ. Καθηγήτρια

Αθήνα, Μάρτιος 2020

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ
ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

Εξόρυξη δεδομένων από Έξυπνες Συσκευές για πρόβλεψη χρήσης και συστάσεις

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Άγγελος Δ. Δανιήλ

Επιβλέπουσα : Ιωάννα Ρουσσάκη
Επίκ. Καθηγήτρια

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την --η Μαρτίου 2020.

.....	.....	.....
Ιωάννα Ρουσσάκη Επίκ. Καθηγήτρια Ε.Μ.Π	Μιλτιάδης Αναγνώστου Καθηγητής Ε.Μ.Π	Συμεών Παπαβασιλείου Καθηγητής Ε.Μ.Π

Αθήνα, Μάρτιος 2020

.....

Άγγελος Δ. Δανιήλ

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Τα τελευταία χρόνια, με τη συνεχή εξέλιξη της τεχνολογίας, οι έξυπνες συσκευές και πιο συγκεκριμένα τα έξυπνα κινητά τηλέφωνα αποτελούν αναπόσπαστο κομμάτι της καθημερινότητας του κάθε ανθρώπου. Η συνεχόμενη όμως τάση εξέλιξης των έξυπνων κινητών σε συνδυασμό με τις όλο και περισσότερες δυνατότητες που προσφέρουν, οδηγούν όπως είναι εύκολα κατανοητό, στην ανάγκη για μεγαλύτερη αυτονομία της συσκευής.

Στην παρούσα διπλωματική, πραγματοποιείται η ανάπτυξη μιας εφαρμογής για συσκευές με λειτουργικό σύστημα Android. Σκοπός της συγκεκριμένης εφαρμογής είναι η καταγραφή και αποθήκευση δεδομένων χρήσης της συσκευής μέσα σε ένα προεπιλεγμένο χρονικό διάστημα. Στη συνέχεια τα δεδομένα αυτά επεξεργάζονται και έπειτα με τη βοήθεια αλγορίθμων μηχανικής μάθησης εξάγονται προβλέψεις που αφορούν τις απαιτήσεις του χρήστη για υπηρεσίες δεδομένων κινητής τηλεφωνίας καθώς και υπηρεσιών τοποθεσίας. Μέσα από αυτές τις προβλέψεις επιτυγχάνεται και ο απώτερος σκοπός που είναι η μεγαλύτερη διάρκεια ζωής της μπαταρίας της συσκευής, η οποία προκύπτει απο την αποφυγή της άσκοπης χρήσης των συγκεκριμένων υπηρεσιών.

Για τις ανάγκες της παρούσας διπλωματικής εργασίας, συλλέγονται δεδομένα από έναν χρήστη Android συσκευής μέσα σε διάστημα επτά ημερών, με ρυθμό καταγραφής δεδομένων ένα λεπτό. Τα δεδομένα αυτά επεξεργάζονται στη συνέχεια και προκύπτουν με αυτόν τον τρόπο οι ανάγκες του χρήστη για δεδομένα κινητής τηλεφωνίας και υπηρεσίες τοποθεσίας.

Λέξεις κλειδιά

Έξυπνες Συσκευές, Ανάπτυξη Εφαρμογών, Android, Μηχανική Μάθηση, Εξόρυξη Δεδομένων, Ταξινόμηση, Εξοικονόμηση Ενέργειας, Επιστήμη Δεδομένων

Abstract

In recent years, as technology evolves, smart devices and more specifically smartphones have become an integral part of each person's everyday life. However, the continuously evolving trend of smartphones, combined with the increasing capabilities they offer, is leading to the need for greater device autonomy in terms of battery life.

In this particular diploma thesis, an application for devices running Android operating system is being developed. The purpose of this application is to record and store device's usage data within a default time period. The collected data are then processed and with the help of machine learning algorithms we produce predictions related to the user requirements for mobile data services as well as location services. These predictions also achieve the ultimate goal of maximizing the battery life of the device, which occurs from avoiding unnecessary use of these services.

For the purposes of this diploma thesis, large amount of data are collected from an Android device user within seven days at a one-minute data capture rate. These data are then processed, resulting in the prediction of the user's needs for mobile data and location services.

Keywords

Smartphones, Software Development, Android, Machine Learning, Data Mining, Classification, Reduce of Energy Consumption, Data Science

Ευχαριστίες

Για την εκπόνηση της παρούσας διπλωματικής εργασίας θα ήθελα να ευχαριστήσω ιδιαιτέρως την επιβλέπουσα επίκουρη καθηγήτρια Ιωάννα Ρουσσάκη καθώς και τον υποψήφιο διδάκτορα Παύλο Κοσμίδη για την ευκαιρία που μου έδωσαν να ασχοληθώ με ένα τόσο ενδιαφέρον θέμα καθώς και για την πολύτιμη βοήθεια και καθοδήγηση που μου προσφέρθηκε καθόλη τη διάρκεια της διαδικασίας.

Ακόμα θα ήθελα να ευχαριστήσω θερμά την οικογένεια μου για όλη τη στήριξη που μου δείχνουν όλα αυτα τα χρόνια, καθώς και για την υπομονή που έχουν κάνει.

Τέλος ευχαριστώ τους φίλους μου που είναι πάντα εκεί για μένα για τις αξέχαστες στιγμές που έχουμε περάσει όλα αυτά τα ωραία φοιτητικά χρόνια.

Πίνακας περιεχομένων

Κεφάλαιο 1

Εισαγωγή

1.1 Έξυπνες συσκευές

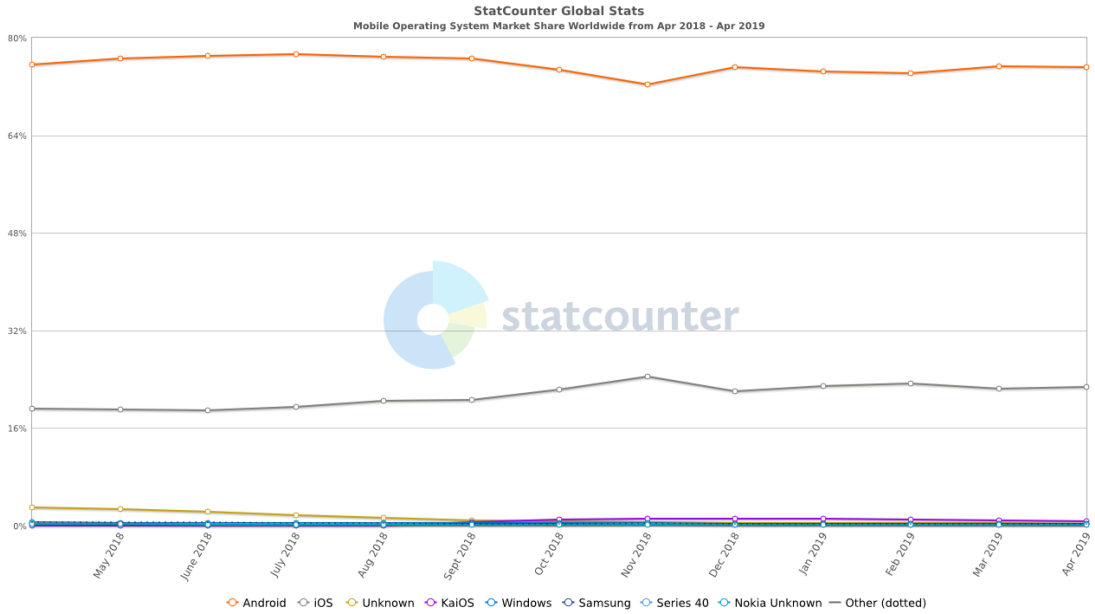
Η εξέλιξη της κινητής τηλεφωνίας τα τελευταία χρόνια είναι εμφανής σε κάθε πτυχή της καθημερινότητας των ανθρώπων. Μάλιστα είναι τόσο ραγδαία που οι τηλεφωνικές συνδέσεις παγκοσμίως ξεπερνούν ακόμα και τον πληθυσμό ολόκληρης της ανθρωπότητας. Πιο συγκεκριμένα τη δεδομένη χρονική στιγμή υπάρχουν περίπου 9 δισεκατομμύρια συνδέσεις ενώ ο πληθυσμός της Γης είναι μόλις πάνω από τα 7.5 δισεκατομμύρια [1].

Με την ανάπτυξη της τεχνολογίας καθώς και του διαδικτύου, το οποίο αποτελεί πλέον αναπόστατο κομμάτι της καθημερινότητας των ανθρώπων, οι ανάγκες τους για επικοινωνία τροποποιούνται σημαντικά και μπορούν να πάρουν διαφορετικές μορφές. Η κυριαρχία των ιστοσελίδων κοινωνικής δικτύωσης, όπως είναι το Facebook, το Twitter, το Google+, καθώς και ιστοσελίδων αναπαραγωγής πολυμέσων, όπως είναι το YouTube, στην καθημερινότητα των ανθρώπων οι οποίες έχουν δημιουργηθεί από εταιρίες κολοσσούς (Facebook, Google, Microsoft, Yahoo) οδήγησαν σε σημαντικές καινοτομίες. Μια τέτοια καινοτομία είναι και η κατασκευή έξυπνων συσκευών (Smartphones) με σκοπό την κάλυψη όσο το δυνατόν περισσότερων αναγκών των καταναλωτών.
Κάπως έτσι οδηγήθηκε η Apple στην κατασκευή και παρουσίαση της πρώτης της έξυπνης συσκευής (iPhone), δίνοντας έτσι το παράδειγμα και στις υπόλοιπες μεγάλες κατασκευάστριες εταιρείες, οι οποίες σε συνεργασία με το λογισμικό της Google οδηγήθηκαν επίσης στην κατασκευή τέτοιων συσκευών. Χάρη στη νέα αυτή τεχνολογία οι άνθρωποι απέκτησαν τη δυνατότητα να κρατούν στα χέρια τους συσκευές που σε πολλές περιπτώσεις έχουν υπολογιστική ισχύ μεγαλύτερη ακόμα και από πολλούς προσωπικούς υπολογιστές (Personal Computers - PC), προσφέροντας παράλληλα το μεγάλο πλεονέκτημα της φορητότητας. Η συνεχής εξέλιξη των έξυπνων συνδέεται και με την εξέλιξη των δικτύων τα οποία κάνουν την πρόσβαση στο διαδίκτυο εύκολη και γρήγορη, προσφέροντας έτσι τη δυνατότητα για τη χρήση όλο και περισσότερων εφαρμογών από οποιοδήποτε σημείο βρίσκεται ο χρήστης. Αρχικά ήταν τα δίκτυα τρίτης γενιάς (3G) τα οποία εισήγαγαν την περιήγηση στο διαδίκτυο, ενώ στη συνέχεια με τα δίκτυα τέταρτης γενιάς (4G) αυξήθηκαν κατα πολύ οι ταχύτητες καθιστώντας εφικτή την αναπαραγωγή περιέχομένου χωρίς να υπάρχουν καθόλου καθυστερήσεις. Τέλος τα δίκτυα πέμπτης γενιάς (5G), αναμένεται να αποτελέσουν βαρόμετρο στην παγκόσμια οικονομία καθώς θα καταστήσουν εφικτή την επικοινωνία μεταξύ των συνδεδεμένων συσκευών στο Διαδίκτυο των Πραγμάτων (Internet of Things) [2].

Εικόνα 1: Παράδειγμα έξυπνης συσκευής (Samsung Galaxy S10 [3]).

1.2 Λειτουργικά Συστήματα και Έξυπνες Συσκευές

Η συντριπτική πλειοψηφία των έξυπνων συσκευών σήμερα χρησιμοποιούν τα δυο δημοφιλέστερα λειτουργικά συστήματα, το Android της Google και το iOS της Apple. Σε αντίθεση με την Apple που διαθέτει το λειτουργικό της σύστημα μόνο στις συσκευές που κατασκευάζει η ίδια (iPhone), η Google το διανέμει εκτενώς σε μεγάλες εταιρίες κατασκευής κινητών συσκευών του χώρου, όπως για παράδειγμα η Samsung, η Sony, η Xiaomi, η Huawei, η OnePlus. Παράλληλα το χρησιμοποιεί πλέον και στις συσκευές που κατασκευάζει η ίδια (Google Pixel), οι οποίες διαθέτουν το λεγόμενο καθαρό Android (stock Android). Πέρα όμως απο τα δυο λειτουργικά συστήματα που προαναφέρθηκαν, πριν από μερικά χρόνια υπήρχαν και αλλα στο προσκήνιο τα οποία πλέον εχουν σχεδον εξαλειφθεί. Χαρακτηριστικό είναι το παράδειγμα του Symbian της Nokia, το οποίο μια δεκαετία πριν κατείχε το 44% της αγοράς, ενώ το Blackberry OS της RIM την ίδια περίοδο ήταν δεύτερο με το 20%. Πλέον το Android και το iOS, εμφανίζονται να έχουν σχεδόν όλο το μερίδιο της αγοράς των λειτουργικών συστημάτων, με ποσοστό που αθροιστικά αγγίζει το 98%, με το υπόλοιπο περίπου 2% να μοιράζεται όπως φαίνεται στις παρακάτω εικόνες [4].



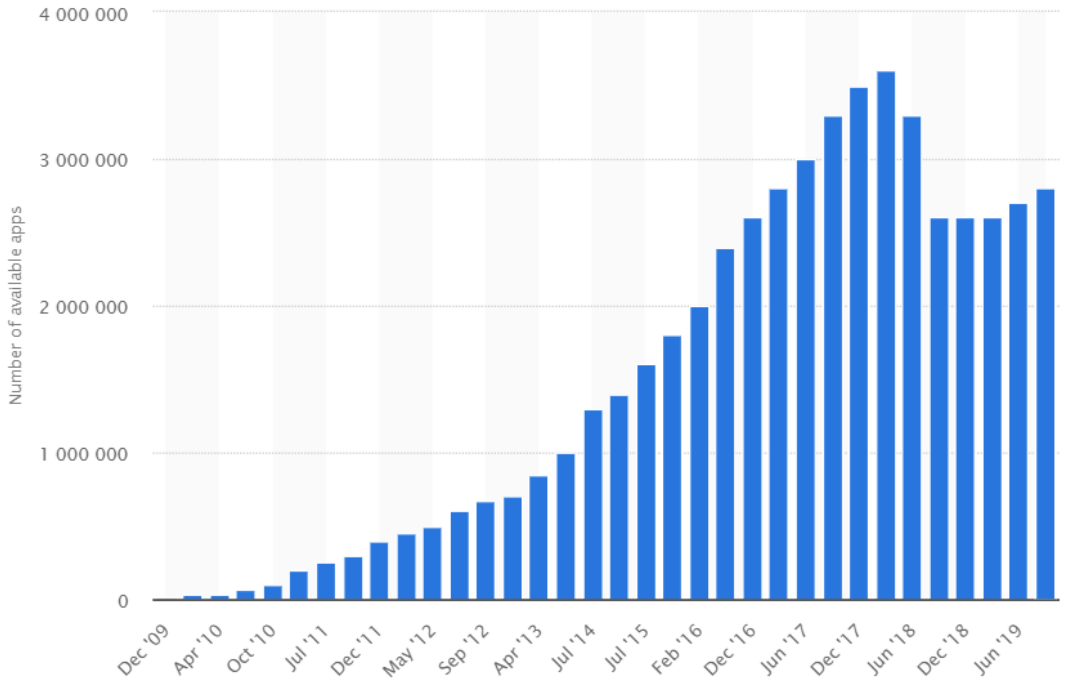
Εικόνα 2: Γράφημα χρήσης των λειτουργικών συστημάτων τους τελευταίους μήνες [4].

Εικόνα 3: Ποσοστό χρήσης του κάθε λειτουργικού συστήματος αυτή τη χρονική στιγμή [4].

Η πληθώρα των εφαρμογών που προσφέρουν τα δυο αυτά λειτουργικά συστήματα είναι τόσο μεγάλη που μπορεί να καλύψει τις ανάγκες ακόμα και του πιο απαιτητικού χρήση. Οι εφαρμογές αυτές μπορεί να είναι ευρείας χρήσης, όπως για παράδειγμα είναι οι εφαρμογές τηλεφώνου, μηνυμάτων κειμένου (SMS), περιήγησης στο διαδίκτυο, μηνυμάτων ηλεκτρονικού ταχυδρομείου (email) κλπ, ενώ παράλληλα μπορούν να χαρακτηριστούν και πιο ειδικού σκοπού όπως είναι για παράδειγμα η παρακολούθηση της καθημερινής αθλητικής δραστηριότητας του χρήση.

Πέρα όμως από τον τεράστιο όγκο εφαρμογών που προσφέρουν τα συγκεκριμενά λειτουργικά συστήματα, δίνουν και τη δυνατότητα στον καθένα να δημιουργήσει τη δική του εφαρμογή. Τα ηλεκτρονικά καταστήματα εφαρμογών (app stores) διαθέτουν εκατομμύρια εφαρμογές, πολλές από τις οποίες έχουν δημιουργηθεί από απλούς χρήστες που διέθεταν γνώσεις προγραμματισμού. Χάρη στο Android και το iOS αυτή τη στιγμή υπάρχουν εφαρμογές για όλα τα διαθέσιμα κοινωνικά δίκτυα, κάνοντας έτσι την επικοινωνία μεταξύ των ανθρώπων ακόμα πιο εύκολη και άμεση, ενώ παράλληλα για την κάλυψη της συνεχόμενης ανάγκης για όλο και περισσότερο αποθηκευτικό χώρο προσφέρεται η δυνατότητα χρήσης των υπηρεσιών υπολογιστικού νέφους (Cloud) από τους χρήστες των δύο αυτών λογισμικών [5]. Με αυτόν τον τρόπο ο χρήστης μπορεί να έχει ανά πάσα στιγμή πρόσβαση σε ένα τεράστιο όγκο δεδομένων με μοναδικό προαπαιτούμενο να είναι συνδεδεμένη η συσκευή στο δίκτυο.

Όπως αναφέρθηκε και παραπάνω, στα ηλεκτρονικά καταστήματα εφαρμογών είναι διαθέσιμη μια τεράστια γκάμα από εφαρμογές. Ο χρήστης έχει τη δυνατότητα να περιηγηθεί από την έξυπνη συσκευή του στα συγκεκριμένα καταστήματα και εφόσον του κινήσει κάτι το ενδιαφέρον να εγκαταστήσει κάποιες από αυτές τις εφαρμογές. Οι περισσότερες διατίθενται δωρεάν ενώ οι υπόλοιπες απαιτούν για την εγκατάσταση τους ενα μικρό αντίτιμο το οποίο είναι συνήθως μικρότερο από 5 ευρώ. Αξίζει να σημειωθεί ότι στο ηλεκτρονικό κατάστημα της Google (Google Play Store), μέχρι τον Ιούνιο του 2019 είχαν καταμετρηθεί τουλάχιστον 2.600.000 εφαρμογές (βλέπε Εικόνα 4), ενώ ο συνολικός αριθμός των λήψεων υπολογίζεται στα 70 δισεκατομμύρια (μόνο για τις συσκευές με λειτουργικό σύστημα Android) [6]. Όπως είναι εύκολα κατανοητό ο συγκεκριμένος τομέας της ανάπτυξης εφαρμογών για λειτουργικό σύστημα Android και γενικότερα για έξυπνες συσκευές, απασχολεί όλους όσους ασχολούνται με τον προγραμματισμό παγκοσμίως, αφού πλέον οι χρήστες με έξυπνες συσκευές είναι τόσοι πολλοί που δημιουργούνται συνεχώς νέες απαιτήσεις.



Εικόνα 4: Ο συνολικός αριθμός διαθέσιμων εφαρμογών στο Google Play Store μέχρι το Ιούνιο του 2019 [6].

1.3 Το λειτουργικό σύστημα Android της Google

Το Android είναι ένα λειτουργικό σύστημα το οποίο χρησιμοποιείται στην πλειοψηφία των έξυπνων συσκευών σήμερα και έχει αναπτυχθεί από την Google, βασισμένο σε μια τροποποιημένη έκδοση του πυρήνα Linux (Linux kernel) [7]. Αρχικά σχεδιάστηκε για φορητές συσκευές αφής, όπως είναι δηλαδή οι έξυπνες συσκευές και οι έξυπνες ταμπλέτες (tablets), ενώ στη συνέχεια επεκτάθηκε και σε συσκευές που αφορούν άλλους τομείς της καθημερινότητας του ανθρώπου.

Χαρακτηριστικά παραδείγματα είναι οι έξυπνες τηλεοράσεις (smartTVs) καθώς και το λειτουργικό σύστημα που χρησιμοποιείται πλέον σε ένα μεγάλο εύρος σύγχρονων αυτοκινήτων το οποίο είναι το Android Auto. Το λειτουργικό σύστημα Android διαχειρίζεται το υλικό (hardware) της συσκευής και παρέχει τις τεχνολογίες που είναι απαραίτητες για τη λειτουργία των εγκατεστημένων εφαρμογών. Μια έξυπνη συσκευή σήμερα με το συγκεκριμένο λειτουργικό σύστημα, παρέχεται με προεγκατεστημένες κάποιες βασικές εφαρμογές. Τέτοιες είναι το Τηλέφωνο, τα Μηνύματα, το Gmail (εφαρμογή ηλεκτρονικού ταχυδρομείου που έχει αναπτυχθεί απο τη Google), το Google Chrome (περιηγητής ιστού) και αρκετές άλλες, που μπορούν να καλύψουν τις βασικές ανάγκες που έχει πλέον ένας χρήστης απο ένα σύγχρονο κινητό τηλέφωνο.

Το Android Software Development Kit (SDK) περιέχει όλα τα απαραίτητα εργαλεία και τις διεπαφές που απαιτούνται για την ανάπτυξη, εγκατάσταση, λειτουργία καθώς και το δοκιμή των εφαρμογών που εμφανίζονται στην αρχική οθόνη μιας Android συσκευής [7]. 'Ολες οι εφαρμογές αναπτύσσονται χρησιμοποιώντας πλαίσια (frameworks) ανάπτυξης εφαρμογών Android. Οι επίσημες γλώσσες ανάπτυξης εφαρμογών είναι η αντικειμενοστρεφής γλωσσα προγραμματισμού Java, ενώ τα τελευταία χρόνια χρησιμοποιείται όλο και περισσότερο η επίσης αντικειμενοστρεφής γλώσσα Kotlin, η οποία έχει αναπτυχθεί απο τη Google [7]. Οι εφαρμογές εγκαθίστανται και τοποθετούνται παράλληλα με τις ήδη εγκατεστημένες εφαρμογές στην οθόνη παρουσίασης εφαρμογών της συσκευής και είναι πάντα διαθέσιμες στο χρήστη. Η κατανόηση των διαθέσιμων τεχνολογιών και των εργαλείων από τα οποία αποτελείται το Android SDK είναι απαραίτητη προυπόθεση για τον σχεδιασμό και την ανάπτυξη μιας τέτοιας εφαρμογής.

Τη στιγμή που γράφεται η παρούσα διπλωματική εργασία, το λειτουργικό σύστημα Android βρίσκεται στην έκδοση 9.0 (Pie).

Έκδοση	**Κωδική Ονομασία**	**Ημερομηνία**	**API level**	**Διανομή**
1.6	*Donut*	15 Σεπτεμβρίου 2009	4	0 %
2.0	*Eclair*	26 Οκτωβρίου 2009	7	0 %
2.2	*Froyo*	20 Μαΐου 2010	8	0%
2.3	*Gingerbread*	9 Φεβρουαρίου 2011	10	0.2%
3.2	*Honeycomb*	15 Ιουλίου 2011	13	0%
4.0	*Ice Cream Sandwich*	16 Δεκεμβρίου 2011	15	0.3%
4.1	*Jelly Bean*	9 Ιουλίου 2012	16	1.2%
4.2	*Jelly Bean*	13 Νοεμβρίου 2012	17	1.9%
4.3	*Jelly Bean*	24 Ιουλίου 2013	18	0.5%
4.4	*KitKat*	31 Οκτωβρίου 2013	19	9.1%

Έκδοση	Κωδική Ονομασία	Ημερομηνία	API level	Διανομή
5.0	*Lollipop*	3 Νοεμβρίου 2014	21	4.2%
5.1	*Lollipop*	9 Μαρτίου 2015	22	16.2%
6.0	*Marshmallow*	5 Οκτωβρίου 2015	23	23.5%
7.0	*Nougat*	22 Αυγούστου 2016	24	21.2%
7.1	*Nougat*	4 Οκτωβρίου 2016	25	9.6%
8.0	*Oreo*	21 Αυγούστου 2017	26	10.1%
8.1	*Oreo*	5 Δεκεμβρίου 2017	27	2.0%
9.0	*Pie*	6 Αυγούστου 2018	28	

Πίνακας 1: Όλες οι εκδόσεις του λειτουργικού συστήματος Android [7].

1.4 Ανάλυση δεδομένων από έξυπνη συσκευή

Η ανάλυση των δεδομένων που προκύπτουν από μια έξυπνη συσκευή που χρησιμοποιεί το λειτουργικό σύστημα Android, μπορεί να αποτελέσει μια σημαντική πηγή εξαγωγής συμπερεσμάτων για τις προτιμήσεις των χρηστών καθώς και για την πραγματοποίηση προβλέψεων. Σε καθημερινή βάση, αμέτρητοι είναι αυτοί που χρησιμοποιούν μια έξυπνη συσκευή Android (Android smartphone) για τις βασικές και όχι μόνο ανάγκες τους, αφού όπως είδαμε παραπάνω το μεγαλύτερο μερίδιο της αγοράς κινητών τηλεφώνων έχει εγκατεστημένο το λειτουργικό σύστημα Android. Μέσα από την πληθώρα των εφαρμογών που μπορεί να φιλοξενήσει μια τέτοια συσκευή καθώς και από τη χρήση της ίδιας της συσκευής στη διάρκεια της ημέρας προκύπτει σημαντική πληροφόρηση για τον εκάστοτε χρήστη η οποία στη συνέχεια μπορεί να χρησιμοποιηθεί για να βελτιώσει ακόμα περισσότερο τη λειτουργικότητα της συσκευής αλλά και την εμπειρία χρήσης της.

1.5 Αντικείμενο της Διπλωματικής

Με τη συνεχώς αυξανόμενη δημοτικότητα των έξυπνων συσκευών, οι άνθρωποι στις μέρες μας δαπανούν πολύ περισσότερο χρόνο με αυτές από ότι παλαιότερα. Η χρήση των συγκεριμένων συσκευών δεν καλύπτει πλέον μόνο τις βασικές ανάγκες του κάθε χρήστη, όπως είναι για παράδειγμα η αποστολή ενός μηνύματος ή η πραγματοποίηση κλήσεων, αλλά καλύπτει πολλές περισσότερες υψηλού επιπέδου ανάγκες που συνδέονται άμεσα σχεδόν με κάθε πτυχή της καθημερινής μας ζωής, π.χ. αναπαραγωγή μουσικής. Τέτοιες ανάγκες υποστηρίζονται κυρίως από διάφορες εφαρμογές, οι οποίες είναι ειδικά σχεδιασμένα προγράμματα λογισμικού κατάλληλα για εκτέλεση σε έξυπνες συσκευές. Καθώς οι νεώτερες τεχνολογίες (π.χ. δικτύωση

4G) και οι πολύπλοκες και απαιτητικές εφαρμογές (π.χ. τρισδιάστατο gaming, Apple's FaceTime) εξελίσσονται και γίνονται όλο και πιο δημοφιλείς, το χάσμα μεταξύ της δυνατότητας χρήσης της συσκευής και της διάρκεια ζωής της μπαταρίας συνεχίζει να αυξάνεται. Με αυτόν τον τρόπο προκαλείται ενόχληση στους εκάστοτε χρήστες, οι οποίοι γίνονται όλο και περισσότερο εξαρτώμενοι από τις έξυπνες συσκευές τους. Η ανισότητα αυτή αποτέλεσε βασικό καταλύτη τα τελευταία χρόνια για την ανάπτυξη αλγορίθμων και στρατηγικών βελτιστοποίησης ενέργειας που στηρίζονται κυρίως σε λογισμικό.

Η παρούσα διπλωματική επικεντρώνεται στην ανάπτυξη μιας Android εφαρμογής, η οποία εκτελείται στο παρασκήνιο για συγκεκριμένο χρονικό διάστημα. Τα δεδομένα που συλλέγονται μέσω της εφαρμογής από τη συσκευή στο ορισμένο χρονικό διάστημα, χρησιμοποιούνται για την εξαγωγή συμπερασμάτων και προβλέψεων με γνώμονα την εξοικονόμηση ενέργειας. Με βάση την εξαγόμενη γνώση από τα δεδομένα αυτά, εφαρμόζονται και συγκρίνονται μεταξύ τους ποικίλοι και διαφορετικοί αλγόριθμοι μηχανικής μάθησης για την κατηγοριοποίηση του χρήστη ανάλογα με τη χρήση που κάνει στη συσκευή. Επιπλέον μέσα από την πληθώρα των χαρακτηριστικών που μας παρέχουν τα δεδομένα αυτά, όπως η ώρα της ημέρας, η στάθμη της μπαταρίας, η κατάσταση του δικτύου, επιλέγονται εκείνα τα χαρακτηριστικά που βελτιστοποιούν το μοντέλο μηχανικής μάθησης και φαίνεται να έχουν προγνωστικές δεξιότητες. Η συγκεκριμένη διπλωματική εμπνεύστηκε από τα την έρευνα που πραγματοποιήθηκε κυρίως στα άρθρα [8] – [10]. Και στις τρείς περιπτώσεις έγινε εξόρυξη δεδομένων απο συσκευή με λειτουργικό σύστημα Android. Στο άρθρο [8] ο βασικός σκοπός ήταν η επίτευξη εξοικονόμησης ενέργειας μέσα απο την καλύτερη διαχείρηση του υλικού της συσκευής. Από την άλλη μερία στα άρθρα [9]-[10] ο στόχος παρέμεινε ο ίδιος, όμως έγινε προσπάθεια να επιτευχθεί μέσα απο την καλύτερη διαχειρήση και χρήση των εγκατεστημένων εφαρμογών της εκάστοτε συσκευής.

1.6 Οργάνωση Κειμένου

Το υπόλοιπο της παρούσας διπλωματικής είναι οργανωμένο ως εξής:

Στο κεφάλαιο 2 παρουσιάζεται η σχετική βιβλιογραφία πάνω στην οποία βασίστηκε η παρούσα διπλωματική για την εκπόνηση της.

Στο κεφάλαιο 3 παρουσιάζεται αναλυτικά το τμήμα της διπλωματικής που αφορά την ανάπτυξη εφαρμοργής Android. Αρχικά γίνεται μια εισαγωγή στην ανάπτυξη εφαρμογών και πιο συγκεκριμένα Android εφαρμογών, ενώ στη συνέχεια παρουσιάζεται και το τεχνικό υπόβαθρο, στο οποίο περιέχεται η γλώσσα προγραμματισμού Java καθώς και τα εργαλεία ανάπτυξης που χρησιμοποιήθηκαν.

Στο κεφάλαιο 4 αναλύεται το θεωρητικό υπόβαθρο γύρω από τη μηχανική μάθηση, καθώς και ορισμένοι αλγόριθμοι που χρησιμοποιήθηκαν. Παράλληλα παρουσιάζεται

η Python ως επιλεγόμενη γλώσσα για την ανάλυση των δεδομένων που προκύπτουν καθώς και κάποιες επιπλέον πληροφορίες για τα εργαλεία και τα περιβάλλοντα ανάπτυξης που χρησιμοποιήθηκαν.

Στο κεφάλαιο 5 περιγράφονται αναλυτικά η σχεδίαση και η υλοποίηση της κύριας εφαρμογής. Περιγράφεται το απαραίτητο λογισμικό που χρησιμοποιήθηκε για να πραγματοποιηθεί η ανάπτυξή της, καθώς και οι δομές δεδομένων και οι αλγόριθμοι που χρησιμοποιήθηκαν. Επίσης αναλύεται η αλληλεπίδραση μεταξύ των διάφορων στοιχείων της εφαρμογής καθώς και η διαδικασία ανάκτησης, συλλογής και αποθήκευσης των δεδομένων απο τη συσκευή για την εξαγωγή συμπερασμάτων.

Στο κεφάλαιο 6 παρουσιάζονται τα εξαγόμενα χαρακτηριστικά από τα συλλεχθέντα δεδομένα με βάση τα οποία δημιουργείται το σύνολο δεδομένων, η διαδικασία προεπεξεργασίας, μετεπεξεργασίας και επιλογής τους για την τροφοδότηση των αλγορίθμων μηχανικής μάθησης. Ακόμα παρουσιάζονται τα αποτελέσματα των αλγορίθμων μηχανικής μάθησης στο πρόβλημα της εξοικονόμησης ενέργειας της συσκευής.

Στο κεφάλαιο 7, παρουσιάζονται τα τελικά συμπεράσματα που προκύπτουν από την διπλωματική εργασία και προτείνονται μελλοντικές επεκτάσεις.

Κεφάλαιο 2

Συναφής Βιβλιογραφία

Αρκετές προσπάθειες έχουν γίνει τα τελευταία χρόνια από διάφορους ερευνητές, προκειμένου να βρεθεί ένα μοντέλο που αξιοποιεί τα δεδομένα που συλλέγει από μια συσκευή, με σκοπό να αυξηθεί η διάρκεια ζωής της μπαταρίας της συσκευής, χωρίς ταυτόχρονα να μειώνεται η εμπειρία χρήσης της και η λειτουργικότητα τους.

Οι συγγραφείς του άρθρου [8] επιχείρησαν να βρουν ένα μοντέλο το οποίο προβλέπει κάθε χρονική στιγμή τις απαιτήσεις του χρήστη για δεδομένα κινητής τηλεφωνίας, καθώς και για υπηρεσίες τοποθεσίας, έχοντας ως είσοδο δεδομένα που συνέλλεξε από την εκάστοτε συσκευή σε συγκεκριμένο χρονικό διάστημα. Πιο συγκεκριμένα χρησιμοποιώντας μια εφαρμογή η οποία εκτελούνταν στο παρασκήνιο σε μια έξυπνη συσκευή για διάστημα μιας εβδομάδας και συλλέγοντας δεδομένα κάθε 1 λεπτό, δημιουργήθηκε ένας μεγάλος όγκος πληροφορίας ο οποίος στη συνέχεια χρησιμοποιήθηκε ως είσοδος στο μοντέλο πρόβλεψης προκειμένου να βρεθεί ένα μοτίβο χρήσης της συσκευής από τον χρήστη, έτσι ώστε να είναι τελικά εφικτή η πρόβλεψη των αναγκών του για υπηρεσίες δεδομένων κινητής τηλεφωνίας και τοποθεσίας.

Στο άρθρο [9] μέσω της έρευνας που έγινε, οι συγγραφείς προσπάθησαν να βρουν λύση στο πρόβλημα της αυτονομίας της μπαταρίας μιας έξυπνης συσκευής δίνοντας έμφαση στο λογισμικό. Αυτό που θέλησαν να πετύχουν στην ουσία είναι η καλύτερη αξιοποιήση των συστατικών του υλικού που χρησιμοποιεί μια εφαρμογή, ανάλογα με την αλληλεπίδραση που έχει ο χρήστης με τη συγκεκριμένη εφαρμογή. Για παράδειγμα μια εφαρμογή μηνυμάτων φορτώνει όλα τα διαθέσιμα μηνύματα, από όλες τις επαφές, παρόλο που ο χρήστης μπορεί να επικοινωνεί με ελάχιστες από αυτές σε καθημερινή βάση. Αυτό σημαίνει ότι η ενέργεια που καταναλώνεται είναι μεγαλύτερη από την πραγματική ανάγκη του χρήστη. Μέσω ενός μηχανισμού προσαρμογής της εφαρμογής στις ανάγκες του χρήστη πετυχαίνεται τελικά σημαντική μείωση της κατανάλωσης ενέργειας, χωρίς παράλληλα να επηρέαζεται η εμπειρία χρήσης της συσκευής.

Στο άρθρο [10] έγινε προσπάθεια να δημιουργηθεί ένα διαφορετικό μοντέλο, το οποίο είχε επίσης ως σκοπό την εξοικονόμηση μπαταρίας. Συγκεκριμένα καταγράφοντας, ανάλογα με διάφορες παραμέτρους όπως για παράδειγμα το διαθέσιμο δίκτυο, τον χρόνο ή την τοποθεσία και τις εφαρμογές που χρησιμοποιούνταν σε περιόδους συνεχόμενης χρήσης της συσκευής, μπορούσαν να

προκύψουν πολλές πληροφορίες για τις προτιμήσεις του χρήστη καθώς και για το πως αλληλεπιδρά με τη συσκευή. Χρησιμοποιώντας ως είσοδο στο μοντέλο την πληροφορία αυτή μπορεί να προβλεφθεί κάθε χρονική στιγμή η εφαρμογή που θα επιθυμεί ο χρήστης να χρησιμοποιήσει καθώς και να υπάρξουν διάφορες προτάσεις σε περίπτωση που επισκεφθεί κάποιο ηλεκτρονικό κατάστημα εφαρμογών περιορίζοντας έτσι αρκετά τον χρόνο αναζήτησης ανάμεσα στο πλήθος των εφαρμογών και πετυχαίνοντας τελικά την αύξηση της διάρκειας ζωής της μπαταρίας.

Με τη χρήση του Mobile Cloud Computing (MCC) οι συγγραφείς στο άρθρο [11] θέλησαν να καταφέρουν τόσο την εξοικονόμηση ενέργειας όσο και υπολογιστικών πόρων, αφού μέσα απο κάποιες παραμέτρους που θα λαμβάνει υπόψη το μοντέλο πρόβλεψης που έφτιαξαν θα αποφασίζεται αν μια εφαρμογή θα τρέξει τοπικά στη συσκευή ή θα εκτελείται στο MCC. Σε παρόμοιο μήκος κύματος στο άρθρο [12] προσπάθησαν να βρουν την καλύτερη δυνατή αξιοποίηση του MCC ανάλογα με το διαθέσιμο δίκτυο, το είδος της εφαρμογής, τις προτιμήσεις του χρήστη αλλά και την ίδια τη συσκευή έτσι ώστε να εξοικονομούνται υπολογιστικοί πόροι και ταυτόχρονα και μπαταρία.

Στην προσπάθεια να μειώσουν την κατανάλωση μπαταρίας, οι συγγραφείς του άρθρου [13] επικεντρώθηκαν στην εύρεση ενός ενεργειακά αποδοτικού μοντέλου εντοπισμού θέσης, όπου θα αποφασίζεται το πότε θα ενεργοποιούνται ή απενεργοποιούνται οι υπηρεσίες τοποθεσίας, ή θα τροποποιείται η συχνότητα με την οποία λαμβάνονται τοποθεσίες. Στο άρθρο [14] πιο συγκεκριμένα προτείνουν έναν μηχανισμό με τον οποίο θα απενεργοποιείται η καταγραφή τοποθεσιών ή θα μειώνεται ο ρυθμός με τον οποίο το GPS (Global Position System) καταγράφει τοποθεσίες ανάλογα με το αν ο χρήστης βρίσκεται σε εσωτερικό ή εξωτερικό χώρο.

Όπως επιβεβαιώνεται από την έρευνα που πραγματοποιήθηκε στο άρθρο [15], η χρήση της αναγνώρισης προτύπων όσον αφορά τον τρόπο που χρησιμοποιείται η κινητή συσκευή δείχνει ότι ο κάθε χρήστης έχει ένα ξεχωριστό μοτίβο χρήσης και ανεξάρτητο από τους υπόλοιπους. Με αυτόν τον τρόπο εφαρμόζονται πιο αποδοτικά οι αλγόριθμοι εξοικονόμησης ενέργειας για κάθε χρήστη χωρίς όμως να διαταράσσονται τα επίπεδα ικανοποίησής του, όπως έγινε και στη συγκεκριμένη μελέτη που επικεντρώθηκε στην ανάλυση του μοτίβου χρήσης ενός smartphone καθώς και τις επιπτώσεις που έχει στη διαχείριση της ενέργειας αλλά και του δικτύου κινητής τηλεφωνίας.

Πολλές ακόμα έρευνες, όπως στα άρθρα [16]-[17] χρησιμοποιούν τα δεδομένα που συλλέγουν από αληθινούς χρήστες κινητών συσκευών έτσι ώστε να προκύψει κάποιο μοτίβο χρήσης που τελικά θα έχει ως σκοπό την εξοικονόμηση ενέργειας. Αξιοσημείωτη είναι και η προσπάθεια στο άρθρο [18], όπου οι συγγραφείς δημιούργησαν ένα ολόκληρο σύστημα, το οποίο αναλαμβάνει τη βελτιστοποίηση της κατανάλωσης ενέργειας μιας έξυπνης συσκευής χωρίς να χρειάζεται η παραμικρή παρεμβολή του χρήστη.

Μια διαφορετική προσέγγιση όσον αφορά την εξοικονόμηση ενέργειας έγινε στο άρθρο [19]. Βασικός σκοπός της συγκεκριμένης έρευνας ήταν να μειωθεί όσο το δυνατόν περισσότερο η κατανάλωση της μπατάριας μέσα από την κατάλληλη επιλογή

δικτύου. Ανάλογα με την εκτιμώμενη κίνηση του κάθε δικτύου αλλά και λαμβάνοντας υπόψη τον όγκο δεδομένων που μεταφέρονται με κάθε δράση του χρήστη, επιλέγεται το καλύτερο δυνατό δίκτυο. Σε παρόμοια λογική στο άρθρο [20] αναπτύχθηκε ένα σύστημα το οποίο κάνει τη χρήση εφαρμογών περισσότερο αποδοτική όσον αφορά την κατανάλωση ενέργειας. Αυτό επιτυγχάνεται μέσα από την ανίχνευση των λιγότερο κοστοβόρων δικτύων για μεταφόρα δεδομένων που πηγάζει από τη χρήση μιας εφαρμογής.

Την μείωση της κατανάλωσης ενέργειας μέσα από την καλύτερη χρήση των υπηρεσιών τοποθεσίας θέλησαν να πετύχουν οι συγγραφείς στα άρθρα [21]-[22]. Οι συγγραφείς του άρθρου [21] προσπάθησαν να το πετύχουν αυτό μέσα από την καλύτερη διαχείρηση των δεδομένων τοποθεσίας που μπορούν να ληφθούν μέσα από τις κεραίες του δικτύου. Από την άλλη μερία στο άρθρο [22] έγινε μια προσπάθεια εντοπισμού της τοποθεσίας μέσα από τη συνεχή επεξεργαρσία των δεδομένων που προσφέρουν οι αισθητήρες της συσκευής, όπως για παράδειγμα το γυροσκόπιο. Ακόμη μια έρευνα που αφορούσε τις υπηρεσίες τοποθεσίας και τη μείωση της κατανάλωσης ενέργειας έγινε στο άρθρο [23]. Οι συγγραφείς του συγκεκριμένου άρθρου επιδίωξαν να αναπτύξουν ένα μοντέλο που προβλέπει πότε είναι αναγκαία η χρήση του GPS ανάλογα με το ιστορικό τοποθεσίας και τα δεδομένα που λαμβάνονται από τον αισθητήρα επιτάχυνσης της συσκευής. Με αυτόν τον τρόπο αποφεύγεται η άσκοπη χρήση της συγκεκριμένης υπηρεσίας που κοστίζει σημάντικα σε ενέργεια.

Μέσα από την ανίχνευση κίνησης της συσκευής και κατ' επέκταση του χρήστη γίνεται προσπάθεια για εξοικονόμηση ενέργειας στα άθρα [24]-[25]. Πιο συγκεκριμένα από τα δεδομένα που συλλέγονται από τους αισθητήρες της συσκευής είναι εφικτή η πρόβλεψη των χρονικής στιγμών που ο χρήστης βρίσκεται στον τόπο κατοικίας του. Αυτό έχει ως αποτέλεσμα τη χρήση του τοπικού δικτύου αλλά και την απενεργοποίηση των διάφορων υπηρεσίων που δεν είναι αναγκαίες τις δεδομένες χρονικές στιγμές.

Οι συγγραφείς του άρθρου [26] επέλεξαν την καλύτερη δυνατή αξιοποίηση του υλικού της συσκευής ως λύση για την εξοικονόμηση ενέργειας. Κύριος σκοπός της συγκεκριμένης έρευνας είναι η εύρεση των ελάχιστων δυνατών απαιτήσεων σε πόρους χωρίς παράλληλα να μειώνεται η γενική απόδοση της συσκευής, κάτι που όπως είναι κατανοητό μπορεί να αυξήσει σημαντικά τη διάρκεια ζωής της μπαταρίας. Παράλληλα στη μελέτη που έγινε στο άρθρο [27] γίνεται προσπάθεια για την καλύτερη αξιοποιήση των εφαρμογών που χρειάζονται ενεργοποιημένες τις υπηρεσίες τοποθεσίας. Αυτό επιτυγχάνεται με την ανάπτυξη ενός λογισμικού το οποίο προσαρμόζει τα αιτήματα της κάθε εφαρμογής για δεδομένα τοποθεσίας όποτε κρίνεται αναγκάιο, μειώνοντας έτσι σημαντικά την κατανάλωση ενέργειας.

Στις έρευνες που έγιναν στα άρθρα [28]-[29], οι συγγραφείς επιχειρούν να διακρίνουν διάφορες καταστάσεις του χρήστη μέσα στην καθημερινότητα του, καθώς και μεταβάσεις μεταξύ των καταστάσεων μέσα από τα δεδομένα που παρέχουν οι διάφοροι αισθητήρες της συσκευής. Μέσα από τις καταστάσεις που βρίσκεται ο χρήστης είναι εφικτή στη συνέχεια η καλύτερη αξιοποίηση των πόρων της συσκευής με αποτέλεσμα να αυξάνεται σημαντικά η διάρκεια ζωής της μπαταρίας της συσκευής. Τέλος, στη μελέτη που έγινε στο άρθρο [30] αναλύεται η χρήση αλγορίθμων μηχανικής μάθησης για την πρόβλεψη της τοποθεσίας κάτι που μπορεί

να φανεί ιδιαίτερα χρήσιμο σε διάφορα μοντέλα με στόχο την καλύτερη αξιοποίηση των πόρων της συσκευής και τελικά την όσο το δυνατόν μικρότερη κατανάλωση ενέργειας.

Κεφάλαιο 3

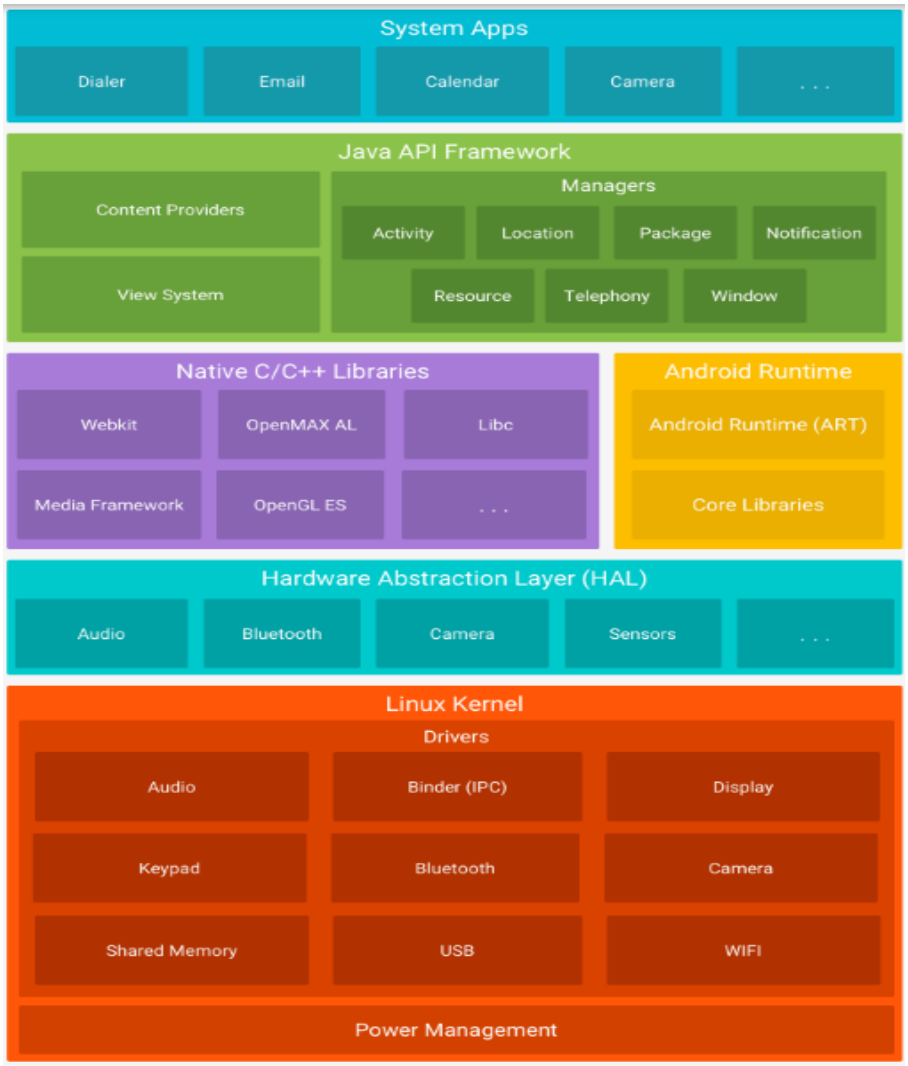
Ανάπτυξη εφαρμογής Android

3.1 Η αρχιτεκτονική του Android

Σε γενικές γραμμές, το Android όπως και κάθε άλλο λειτουργικό σύστημα αποτελεί τον διαμεσολαβητή μεταξύ του υποκείμενου υλικού (hardware) και του διαθέσιμου λογισμικού (software), που στην προκειμένη περίπτωση είναι οι εγκατεστημένες εφαρμογές που εμφανίζονται στην οθόνη. Οι εφαρμογές αυτές σπάνια επικοινωνούν άμεσα με το υλικό της συσκευής. Κάθε φορά που απαιτείται μια τέτοια επικοινωνία γίνεται χρήση μιας σειράς καλά καθορισμένων διεπαφών (interfaces) του συστήματος, κάτι που σημαίνει ότι η καθεμία εφαρμογή παραμένει ανεπηρέαστη από αλλαγές στο υλικό. Με αυτόν τον τρόπο είναι εφικτή η λειτουργία των εφαρμογών σε συσκευές με διαφορετικά υλικά.

Οι τεχνολογίες του λειτουργικού συστήματος Android μπορούν να συγκεντρωθούν στα παρακάτω στρώματα [31]:

1. Linux Kernel.
2. Hardware Abstraction Layer (HAL).
3. Android Runtime.
4. Native Libraries.
5. Java API Framework.
6. System Αρps.



Εικόνα 5: Στρώματα αρχιτεκτονικής Android [31].

3.1.1 Το στρώμα Πυρήνας Linux (Linux Kernel)

Όπως έχει αναφερθεί και σε προηγούμενο κεφάλαιο το λειτουργικό σύστημα Android είναι βασισμένο σε μια τροποποιημένη έκδοση του πυρήνα Linux. Ακόμα και το περιβάλλον εκτέλεσης Android (Android Runtime) το οποίο αναλύεται παρακάτω βασίζεται σε μεγάλο βαθμό στον πυρήνα του Linux για λειτουργίες που αφορούν πολυνηματικότητα (multithreading) ή διαχείρηση μνήμης.

Παράλληλα με τη χρήση του πυρήνα Linux, το λειτουργικό σύστημα Android είναι σε θέση να εκμεταλλευτεί στο έπακρο χαρακτηριστικά που έχουν να κάνουν με την ασφάλεια, επιτρέποντας έτσι στους κατασκευαστές έξυπνων συσκευών να δημιουργήσουν κατάλληλα προγράμματα οδήγησης υλικού (hardware drivers) για έναν ήδη γνωστό πυρήνα.

3.1.2 Το στρώμα αφαίρεσης Υλικού (Hardware Abstraction Layer)

Το συγκεκριμένο επίπεδο παρέχει όλες τις διεπαφές εκείνες που αφορούν τις δυνατότητες που έχει το υλικό και οι οποίες μπορούν να χρησιμοποιήθούν στο ανώτερο επίπεδο Java API (Application Programming Interface) framework. Το στρώμα αφαίρεσης υλικού αποτελείται από πολλά αρθρώματα (modules) βιβλιοθηκών καθένα από τα οποία είναι υπεύθυνο για την ενσώματωση ενός συστατικού του υλικού μέσω της κατάλληλης διεπαφής. Ένα τέτοιο άρθρωμα είναι και η κάμερα του κινητού. Όταν ένα πλαίσιο (framework) API επιθυμεί την πρόσβαση στο υλικό της συσκευής, το λειτουργικό σύστημα Android φορτώνει το καταλλήλο άρθρωμα από τη βιβλιοθήκη για το εκάστοτε συστατικό υλικού.

3.1.3 Το στρώμα Περιβάλλον Εκτέλεσης Android (Android Runtime)

Στις συσκευές που έχουν εγκατεστημένη έκδοση Android 5.0 ή μεγαλύτερη (δηλαδή επίπεδο API μεγαλύτερο ή ισο του 21), κάθε μια από τις υπάρχουσες εφαρμογές εκτελείται σε ξεχωριστή διεργασία και χρησιμοποιώντας ξεχωριστό στιγμιότυπο του περιβάλλοντος εκτέλεσης Android. Το συγκεκριμένο περιβάλλον είναι σχεδιασμένο να εκτελεί πολλαπλές εικονικές μηχανές σε συστήματα με χαμηλή μνήμη μέσω της εκτέλεσης αρχείων DEX (Dalvik EXecutable). Τα αρχεία αυτά είναι της μορφής bytecode και είναι ειδικά σχεδιασμένα για Android έτσι ώστε να καταναλώνουν όσο το δυνατόν λιγότερη μνήμη. Κάποια ήδη υπάρχοντα εργαλεία όπως το Jack, χρησιμοποιούνται για τη μεταγλώττιση αρχείων Java σε DEX bytecode, τα οποία στη συνέχεια μπορούν να εκτελεστούν σε οποιαδήποτε πλατφόρμα Android.

Τα κυριότερα χαρακτηριστικά που περιλαμβάνει το περιβάλλον εκτέλεσης Android είναι τα ακόλουθα:

- Έγκαιρη και ακριβή μεταγλώττιση.
- Βελτιστοποιημένη διαδικασία συλλογής σκουπιδιών (garbage collection).
- Για τις νεότερες εκδόσεις Android (έκδοση 9.0 και πάνω ή αλλιώς επίπεδο API μεγαλύτερο ή ίσο του 28) υποστηρίζεται η μετατροπή των αρχείων (που είναι σε μορφή DEX) ενός πακέτου εφαρμογής σε μια πιο συμβατή μορφή κώδικα μηχανής.

Αξίζει να σημειωθέι ότι στις εκδόσεις Android πριν από την 5.0 το περιβάλλον εκτέλεσης ήταν το Dalvik. Επίσης το Android διαθέτει έναν πυρήνα από βιβλιοθήκες εκτέλεσης, οι περισσότερες από τις οποίες προσφέρουν πολλά σημαντικά χαρακτηριστικά της γλώσσας προγραμματισμού Java και τις οποίες χρησιμοποιεί στη συνέχεια το πλαίσιο Java API.

3.1.4 Το στρώμα Τοπικές Βιβλιοθήκες C/C++ (Native C/C++ Libraries)

Πολλά συστατικά και υπηρεσίες του λειτουργικού συστήματος Android, όπως για παράδειγμα το περιβάλλον εκτέλεσης ή το στρώμα αφαίρεσης υλικού που περιγράφηκαν παραπάνω, έχουν αναπτυχθεί με τη χρήση βιβλιοθηκών που είναι γραμμένες σε C/C++. Το λειτουργικό σύστημα Android παρέχει πλαίσια Java API ικανά να εκμεταλλευτούν τις λειτουργίες αυτών των βιβλιοθήκων και να τις χρησιμοποιήσουν σε μια εφαρμογή. Για παράδειγμα μέσα απο την πρόσβαση στο OpenGL ES χρησιμοποιώντας το πλαίσιο Java OpenGL API, είναι δυνατή η σχεδίαση αλλά και ο χειρισμός δισδιάστατων και τρισδιάστατων γραφικών μέσα στην εφαρμογή.

3.1.5 Το στρώμα Πλαίσιο Java API (Java API Framework)

Όλο το εύρος των χαρακτηριστικών του λειτουργικού συστήματος Android καθίσταται διαθέσιμο στον καθένα μέσα από τις διάφορες διεπαφές προγραμματισμού εφαρμογών (Application Programming Interface – API), οι οποίες είναι γραμμένες σε γλώσσα Java. Μέσα από τη χρήση τέτοιων διεπαφών σχηματίζονται όλα τα απαραίτητα κομμάτια τα οποία είναι απαραίτητα για τη δημιουργία μιας εφαρμογής. Αυτό επιτυγχάνεται μέσω της επαναχρησιμοποίησης σημαντικών συστατικών και υπηρεσιών του συστήματος Android, τα οποία περιλαμβάνουν τα παρακάτω:

- Ένα πλούσιο σε επιλογές και επεκτάσιμο σύστημα όψεων (View System), το οποίο επιτρέπει τη δημιουργία μιας διεπαφής χρήστη μιας εφαρμογής (User Interface) περιλαμβάνοντας λίστες, σημεία εισαγωγής κειμένου, κουμπιά, ακόμα και ενσωματωμένο περιηγητή ιστού.
- Ένα διαχειριστή πηγών (Resource Manager), ο οποίος παρέχει πρόσβαση σε μη προγραμματιζόμενα αρχεία που αφορούν για παράδειγμα τα γραφικά ή τη στοίχιση των εικονιδίων της οθόνης.
- Ένα διαχειριστή ειδοποιήσεων (Notification Manager), ο οποίος είναι υπεύθυνος για την εμφάνιση όλων των ειδοποιήσεων στη γραμμή κατάστασης.
- Ένα διαχειριστή δραστηριοτήτων (Activity Manager), ο οποίος διαχειρίζεται τον κύκλο ζωής όλων των εφαρμογών και παρέχει μια στοίβα περιήγησης στις εφαρμογές που ήδη εκτελούνται.
- Παρόχους περιεχομένου (Content Providers), οι οποίοι επιτρέπουν στις εφαρμογές να έχουν πρόσβαση σε δεδομένα από άλλες εφαρμογές ή ακόμα και να μοιράζονται τα δικά τους δεδομένα.

3.1.6 Το στρώμα Εφαρμογές Συστήματος (System Apps)

Το λειτουργικό σύστημα Android διατίθεται σε καθεμία από τις διαθέσιμες εκδόσεις με κάποιες προεγκατεστημένες εφαρμογές συστήματος, όπως είναι η αποστολή μηνυμάτων κειμένου και ηλεκτρονικού ταχυδρομείου, οι επαφές, η περιήγηση στο διαδίκτυο, το ημερολόγιο αλλά και πολλές άλλες. Το γεγονός αυτό βέβαια δεν εμποδίζει σε καμία περίπτωση τον εκάστοτε χρήστη να εγκαταστήσει οποιαδήποτε εφαρμογή τρίτου τύπου (third party app) επιθυμεί, ορίζοντας έτσι για παράδειγμα ένα διαφορετικό προεπιλεγμένο περιηγητή ιστού, ή ακόμα και διαφορετικό προεπιλεγμένο πληκτρολόγιο.

Το συγκεκριμένο στρώμα πέρα από τις εφαρμογές που παρέχει στον χρήστη δίνει και πολλές δυνατότητες στους προγραμματιστές εφαρμογών Android. Αν για παράδειγμα κάποιος θέλει να ενσωματώσει στην εφαρμογή του τη δυνατότητα αποστολής μηνύματος ηλεκτρονικού ταχυδρομείου δε χρειάζεται να προγραμματίσει τη συγκεκριμένη λειτουργία, παρά μόνο να χρησιμοποιήσει την ήδη εγκατεστημένη εφαρμογή του κάθε χρήστη για αυτό το σκοπό.

3.2 Συστατικά Μέρη μιας Android Εφαρμογής

Τα συστατικά μέρη (Components) μιας εφαρμογής αποτελούν τους βασικούς πυλώνες για την ανάπτυξή της. Κάθε τέτοιο συστατικό αποτελεί ένα σημείο από το οποίο μπορεί το σύστημα ή ο χρήστης να εισέλθει στην εφαρμογή, ενώ παράλληλα δεν είναι λιγα τα συστατικά μέρη που εξαρτώνται απο άλλα.

Υπάρχουν τέσσερα βασικά είδη τέτοιων συστατικών μερών [32]:

1. Δραστηριότητες (Activities).
2. Υπηρεσίες (Services).
3. Δέκτες εκπομπής (Broadcast Receivers).
4. Πάροχοι Περιεχομένου (Content Providers).

3.2.1 Δραστηριότητα - Activity

Μια δραστηριότητα είναι στην ουσία το σημείο από το οποίο ο χρήστης μπορεί να αλληλεπιδράσει με την εκάστοτε εφαρμογή. Αποτελεί πρακτικά μια οθόνη μαζι με ένα γραφικό περιβάλλον μέσα από το οποίο ο χρήστης βλέπει το ανάλογο περιέχομενο και αν είναι εφικτό προχωράει και σε διάφορες δράσης (πχ πάτημα κουμπιού). Μια εφαρμογή μπορεί να αποτελείται από πολλές δραστηριότητες οι οποίες αλληλεπιδρούν μεταξύ τους πολλές φορές για να προσφέρουν την κατάλληλη εμπειρία χρήσης. Για παράδειγμα σε μια εφαρμογή ηλεκτρονικού ταχυδρομείου η λίστα με τα διαθέσιμα μηνύματα, η σύνθεση ενός νέου μηνύματος ηλεκτρονικού ταχυδρομείου αλλα και η ανάγνωση ενός τέτοιου μηνύματος αποτελούν διαφορετικές

δραστηριότητες, ανεξάρτητες μεταξύ τους, οι οποίες συνεργάζονται για να βελτιστοποιήσουν την εμπειρία χρήσης.

Μια δραστηριότητα διευκολύνει τις παρακάτω σημαντικές αλληλεπιδράσεις μεταξύ συστήματος και εφαρμογής:

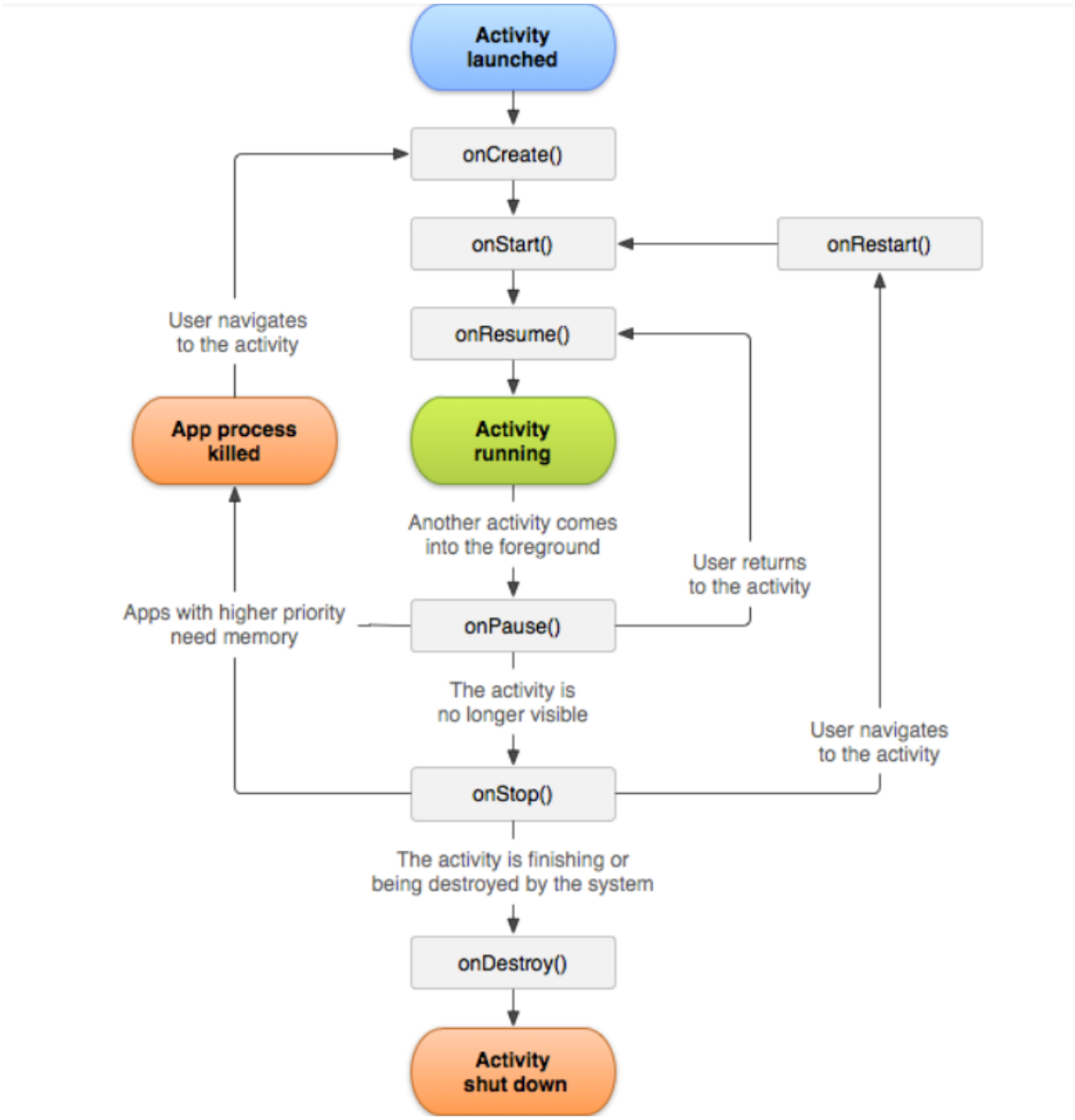
- Παρακολούθηση της δραστηριότητας του χρήστη κάθε χρονική στιγμή. Μέσα από το περιεχόμενο της οθόνης το σύστημα είναι σε θέση να συνεχίζει την εκτέλεση της διεργασίας που φιλοξενεί τη συγκεκριμένη δραστηριότητα.
- Γνώση για τις προηγούμενες δραστηριότητες που εκτελέστηκαν. Με αυτόν τον τρόπο το σύστημα γνωρίζει ότι ίσως ο χρήστης χρειαστεί δεδομένα από τέτοιες δραστηριότητες με αποτέλεσμα να υπάρχει υψηλότερη προτεραιότητα στις ανάλογες διεργασίες για επανεκτέλεση.
- Σε περίπτωση που μία διεργασία τερματιστεί, το σύστημα φροντίζει έτσι ώστε η εφαρμογή να συνεχίσει κανονικά τη λειτουργία της και να είναι σε θέση ο χρήστης να επανέλθει σε μία δραστηριότητα με την προηγούμενη κατάσταση αποθηκευμένη.
- Παρέχεται η δυνατότητα στις εφαρμογές να ενσωματώνουν ροές χρηστών μεταξύ τους αλλά και στο σύστημα η δυνατότητα να συντονίζει τις ροές αυτές, όπως είναι για παράδειγμα ένα παιχνίδι που φιλοξενεί πολλούς χρήστες οι οποίοι αλληλεπιδρούν μεταξύ τους.

Καθώς ο χρήστης περιηγείται μέσα σε μια εφαρμογή, τα στιγμιότυπα της κάθε δραστηριότητας αλλάζουν συνεχώς καταστάσεις κατά τη διάρκεια του κύκλου ζωής τους (Activity Lifecycle). Η κλάση της δραστηριότητας προσφέρει πολλές μεθόδους μέσα από τις οποίες είναι σε θέση να γνωρίζει ότι η κατάσταση έχει αλλάξει. Οι συγκεκριμένες μέθοδοι αναλύονται παρακάτω:

- **onCreate():** η μέθοδος αυτή καλείται όταν το σύστημα δημιουργεί τη δραστηριότητα.
- **onStart():** η συγκεκριμένη μέθοδος καλείται όταν έχει ξεκινήσει η δραστηριότητα, δηλαδή όταν γίνεται ορατή στο χρήστη.
- **onResume():** μέσα από αυτή τη μέθοδο η δραστηριότητα έρχεται στο προσκήνιο και είναι σε θέση ο χρήστης να αλληλεπιδράσει με την εφαρμογή.
- **onPause()**: η μέθοδος αυτή είναι η πρώτη ένδειξη ότι ο χρήστης είναι έτοιμος να αφήσει τη δραστηριότητα, χωρίς όμως να σημαίνει οτι πρέπει να καταστραφεί. Ουσιαστικά καλείται όταν η δραστηριότητα δεν είναι πια στο προσκήνιο.
- **onStop():** με τη χρήση της συγκεκριμένης μεθόδου η δραστηριότητα σταματάει να είναι πλέον ορατή στο χρήστη
- **onDestroy():** η μέθοδος αυτή καλείται λίγο πριν καταστραφεί η δραστηριότητα. Οι λόγοι που μπορεί να καλεστεί είναι είτε επειδή η

δραστηριότητα τελειώνει, είτε επειδή γίνεται κάποια αλλαγή σε κάποια ρύθμιση, όπως είναι για παράδειγμα η περιστρεφόμενη οθόνη.

- **onRestart():** η συγκεκριμένη μέθοδος μπορεί να καλεστεί όταν πρόκειται να επανέλθει στο προσκήνιο η δραστηριότητα μετα τη διακοπή της, δηλαδή μετα τη μέθοδο onStop().



Εικόνα 6: Κύκλος ζωής μιας δραστηριότητας [33].

3.2.1.1 Διατάξεις - Layouts

Οι διαταξείς (Layouts) είναι υπεύθυνες για τη δομή μια διεπαφής χρήστη, επομένως όπως είναι κατανοητό αποτελούν σημαντικό κομμάτι για το πως θα εμφανίζεται μια δραστηριότητα στο χρήστη.

Όλα τα στοιχεία μέσα σε μια διάταξη κατασκευάζονται χρησιμοποιώντας μια ιεραρχία αντικειμένων όψεων (View) και ομάδων όψεων (View Groups). Μια όψη σχεδιάζει συνήθως οτιδήποτε μπορεί να δει ο χρήστης και να αλληλεπιδράσει μαζί του, ενώ μια ομάδα όψεων ευθύνεται για τη δομή της διάταξης των όψεων

Τα αντικείμενα όψεων αποκαλούνται συνήθως γραφικά (widgets) και μπορούν να αποτελέσουν μια από τις πολλές υποκλάσεις, όπως είναι για παράδειγμα τα κουμπιά (Buttons). Τα αντικείμενα των ομάδων όψεων απ' την άλλη, αποκαλούνται συνήθως διατάξεις και μπορούν να πάρουν έναν από τους πολλούς τύπους που ορίζουν μια διαφορετική δομή διάταξης, όπως για παράδειγμα ConstraintLayout.

3.2.2 Υπηρεσία - Service

Μια υπηρεσία είναι ένα στοιχείο εφαρμογής που μπορεί να εκτελεί εκτεταμένες εργασίες στο παρασκήνιο και δεν παρέχει διεπαφή χρήστη. Ένα άλλο στοιχείο της εφαρμογής μπορεί να ξεκινήσει μία υπηρεσία και συνεχίζει να εκτελείται στο παρασκήνιο, ακόμη και αν ο χρήστης μεταβαίνει σε άλλη εφαρμογή. Επιπρόσθετα, ένα στοιχείο μπορεί να δεσμεύσει μια υπηρεσία για να αλληλεπιδράσει μαζί του και να πραγματοποιήσει ακόμη και επικοινωνία μεταξύ διεργασιών. Για παράδειγμα, μία υπηρεσία μπορεί να χειριστεί συναλλαγές δικτύου, να παίξει μουσική, να εκτελέσει ανάγνωση και εγγραφή αρχείων ή να αλληλεπιδράσει με έναν παροχέα περιεχομένου. Όλα αυτά τα επιτυγχάνει απο το παρασκήνιο.

Υπάρχουν τρία διαφορετικά είδη υπηρεσιών:

- **Foreground**
  Μια υπηρεσία προσκηνίου εκτελεί κάποια λειτουργία που είναι αξιοσημείωτη για τον χρήστη. Οι συγκεκριμένες υπηρεσίες πρέπει να εμφανίζουν μια ειδοποιήση στο χρήστη για όσο χρονικό διάστημα εκτελούνται, ενώ συνεχίζουν να εκτελούνται ακόμα και όταν ο χρήστης δεν αλληλεπιδρά με την εφαρμογή.
- **Background**
  Μια υπηρεσία παρασκηνίου εκτελεί μια ενέργεια που δεν παρατηρείται άμεσα από τον χρήστη. Για παράδειγμα, εάν μία εφαρμογή χρησιμοποίησε μία υπηρεσία για να συμπιέσει το αποθηκευτικό χώρο της, αυτό θα ήταν συνήθως μία υπηρεσία παρασκηνίου.
- **Bound**
  Μία δεσμευμένη υπηρεσία προσφέρει μία διεπαφή πελάτη-διακομιστή (client-

server) που επιτρέπει στα συστατικά μέρη να αλληλεπιδρούν με την υπηρεσία, να στέλνουν αιτήματα, να λαμβάνουν αποτελέσματα.

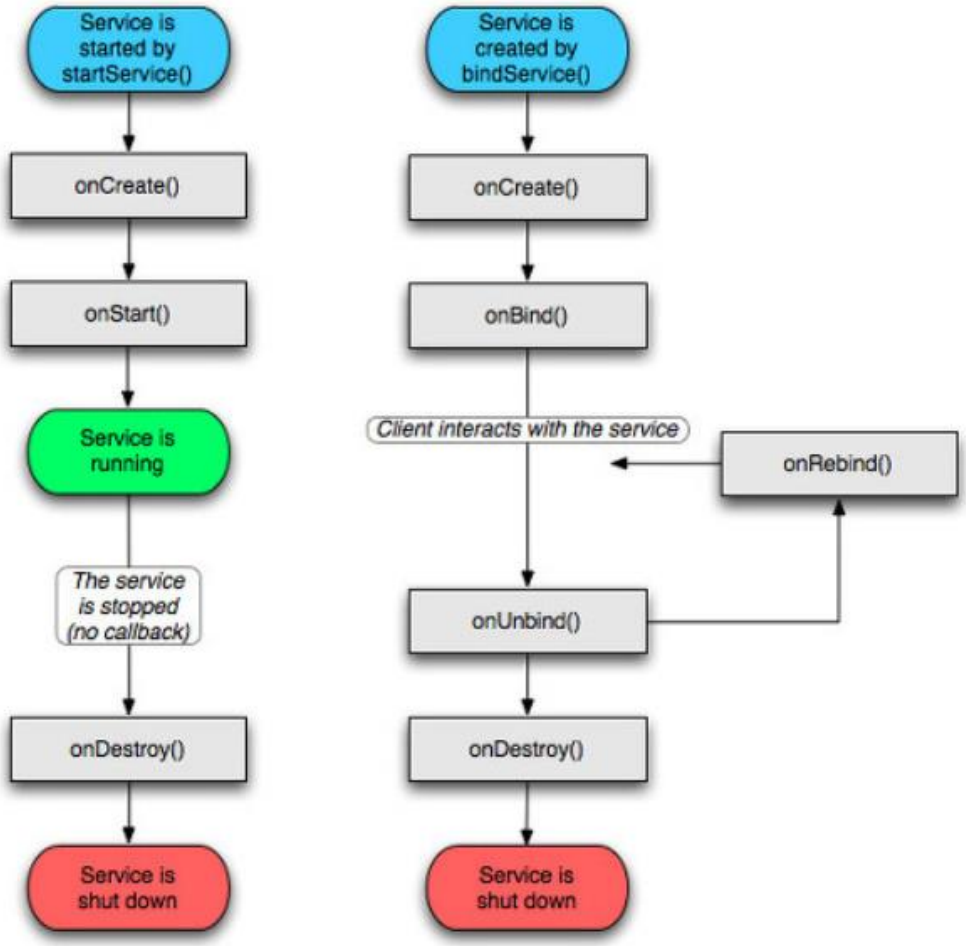
Για τη δημιουργία μιας υπηρεσίας, πρέπει να δημιουργηθεί μία υποκλάση της κλάσης Service ή να χρησιμοποιηθεί μία από τις υπάρχουσες υποκλάσεις της. Στην ενσωμάτωση μιας υπηρεσίας στην εφαρμογή, πρέπει να αντικατασταθούν ορισμένες μέθοδοι επανάκλησης που χειρίζονται βασικές πτυχές του κύκλου ζωής των υπηρεσιών και να παρασχεθεί ένας μηχανισμός που επιτρέπει στα εξαρτήματα να συνδέονται με την υπηρεσία, εάν χρειάζεται.

Αυτές είναι οι πιο σημαντικές μέθοδοι επανάκλησης που πρέπει να παρακαμφθούν:

- **onStartCommand():**
  Το σύστημα επικαλείται αυτή τη μέθοδο καλώντας τη μέθοδο startService() όταν ένα άλλο στοιχείο (όπως μια δραστηριότητα) ζητά την εκκίνηση της υπηρεσίας. Όταν εκτελείται αυτή η μέθοδος, η υπηρεσία εκκινείται και μπορεί να εκτελεστεί στο παρασκήνιο επ' αόριστον. Αν εφαρμοστεί η συγκεκριμένη εντολή, είναι στην ευθύνη του προγραμματιστή να σταματήσει την υπηρεσία όταν ολοκληρωθεί η εργασία της, κάνοντας χρήση των μεθόδων stopSelf() ή stopService().
- **onBind()**:
  Το σύστημα επικαλείται αυτή τη μέθοδο καλώντας την εντολή bindService() όταν ένα άλλο στοιχείο θέλει να δεσμεύσει την υπηρεσία. Κατά την εφαρμογή αυτής της μεθόδου, πρέπει να δοθεί μία διεπαφή που χρησιμοποιούν οι πελάτες για να επικοινωνήσουν με την υπηρεσία, επιστρέφοντας έναν IBinder. Πρέπει πάντα να ενσωματώνεται αυτή η μέθοδος, ωστόσο, αν δεν είναι επιθυμητό να επιτρέπεται η σύνδεση, θα πρέπει να επιστραφεί το μηδέν (null).
- **onCreate():**
  Το σύστημα κάνει χρήση αυτής της μεθόδου για την εκτέλεση διαδικασιών που εκτελούνται μόνο μία φορά κατά την αρχική δημιουργία της υπηρεσίας (πριν κληθεί οποιαδήποτε απο τις onStartCommand() και onBind()). Εάν η υπηρεσία εκτελείται ήδη, αυτή η μέθοδος δεν καλείται.
- **onDestroy():**
  Το σύστημα επικαλείται αυτή τη μέθοδο όταν η υπηρεσία δεν χρησιμοποιείται πλέον και καταστρέφεται. Η υπηρεσία θα πρέπει να έχει ενσωματωμένη αυτή την εντολή για να απελευθερώσει τυχόν πόρους που έχουν δεσμευτεί, όπως τα νήματα, τους εγγεγραμμένους ακροατές (listeners) ή τους δέκτες. Αυτή είναι η τελευταία κλήση που λαμβάνει η υπηρεσία.

Το σύστημα Android σταματά μία υπηρεσία μόνο όταν η μνήμη είναι αρκετά περιορισμένη και πρέπει να ανακτήσει τους πόρους του συστήματος για τη δραστηριότητα που έχει εστιάσει να χρησιμοποιήσει ο χρήστης. Εάν η υπηρεσία συνδέεται με μια δραστηριότητα που έχει την εστίαση του χρήστη, είναι λιγότερο

πιθανό να τερματιστεί, ενώ αν η υπηρεσία δηλωθεί ότι εκτελείται στο προσκήνιο επίσης σπάνια τερματίζεται. Εάν η υπηρεσία ξεκινήσει και είναι μακρά σε διάρκεια, το σύστημα της δίνει μικρότερη προτεραιότητα στον κατάλογο των διεργασιών στο παρασκήνιο με την πάροδο του χρόνου και η υπηρεσία γίνεται ιδιαίτερα ευάλωτη στο να τερματιστεί απο το σύστημα.



Εικόνα 7: Κύκλος ζωής μιας υπηρεσίας [34].

3.3 Η γλώσσα προγραμματισμού Java

Η Java είναι μία αντικειμενοστρεφής γλώσσα προγραμματισμού, η οποία χαρακτηρίζεται ως γλώσσα γενικού σκοπού, με βασικότερο μέλημα ο μεταγλωτισμένος κώδικας σε Java να τρέχει σε οποιαδήποτε πλατφόρμα την υποστηρίζει χωρίς να χρειάζεται να μεταγλωτιστεί ξανά. Η συγκεκριμένη γλώσσα αναπτύχθηκε αρχικά απο τον James Gosling στη Sun Microsystems και κυκλοφόρησε στην αγορά το 1995.

3.3.1 Χαρακτηριστικά της γλώσσας Java

Τα βασικά χαρακτηριστικά της γλώσσας Java [35]-[36] είναι τα εξής: απλή, αντικειμενοστρεφής, ανεξάρτητη από την πλατφόρμα αλλά και από την αρχικτεκτονική, ασφαλής, υψηλής απόδοσης, πολυνηματική και δυναμική.

Απλή:

Η Java είναι μια γλώσσα εύκολη στην εκμάθηση ενώ η σύνταξη της χαρακτηρίζεται απλή, καθαρή και κατανοητή. Οι λόγοι που συμβάλλουν στα προαναφερθέντα και καθιστούν τη συγκεκριμένη γλώσσα απλή έχουν να κάνουν με το ότι η σύνταξη της είναι βασισμένη στην πολύ δημοφιλή γλώσσα C++ απορρίπτοντας παράλληλα πολλές πολύπλοκες λειτουργίες της που χρησιμοποιούνται σπάνια. Επιπλέον μέσω της αυτόματης συλλογής σκουπιδιών (automatic Garbage Collection) καθίσταται προαιρετική η αφαίρεση αντικειμένων στα οποία δεν υπάρχουν αναφορές σε διάφορα σημεία του κώδικα.

Αντικειμενοστρεφής

Η γλώσσα Java χαρακτηρίζεται ως αντικειμενοστρεφής γλώσσα προγραμματισμού καθώς είναι σχεδιασμένη πάνω στη λογική των αντικειμένων. Πιο συγκεκριμένα αντικειμενοστρεφής σημαίνει ο συνδυασμός διαφορετικού τύπου αντικειμένων τα οποία ενσωματώνουν τόσο δεδομένα όσο και συμπεριφορές και τα οποία συνδυάζονται μεταξύ τους με σκοπό την παραγωγή ενος λογισμικού.

Ανεξάρτητη απο την πλατφόρμα και την αρχιτεκτονική

Η Java χαρακτηρίζεται ώς ανεξάρτητη από την πλατφόρμα καθώς όπως αναφέρθηκε και παραπάνω ο μεταγλωτισμένος κώδικας μπορεί να τρέξει οπουδήποτε υποστηρίζεται η συγκεκριμένη γλώσσα χωρίς να χρειάζεται ξανά μεταγλώττιση. Το ίδιο δε συμβαίνει με άλλες γλώσσες, όπως η C και η C++ που πρέπει να μεταγλωτίζονται κάθε φορά που εκτελούνται σε διαφορετική πλατφόρμα. Ο κώδικας της Java αφού μεταγλωτιστεί μετατρέπεται στο λεγόμενο κώδικα byte (bytecode) ο οποίος είναι ανεξάρτητος απο την πλατφόρμα εκτέλεσης και μπορεί να τρέξει για παράδειγμα τόσο σε Windows όσο και Linux, macOS, Solaris και πολλά άλλα.

Παράλληλα η Java είναι ανεξάρτητη της αρχιτεκτονικής. Αυτό σημαίνει ότι δεν υπάρχουν εξαρτήσεις για τα χαρακτηριστικά της Java ανάλογα με την αρχιτεκτονική του συστήματος που θα τη φιλοξενήσει. Στη γλώσσα C για παράδειγμα ο ακέραιος τύπος μεταβλητών καταλαμβάνει 2 bytes μνήμης σε αρχιτεκτονικές 32-bit και 4 bytes σε 64-bit αρχιτεκτονικές. Στη Java κάτι τέτοιο δε συμβαίνει καθώς ο συγκεκριμένος τύπος μεταβλητών καταλαμβάνει 4 bytes ανεξάρτητα από την αρχιτεκτονική του

συστήματος.

Ασφαλής

Η Java είναι ιδιαίτερα δημοφιλής για την ασφάλεια της. Με τη χρήση της συγκεκριμένης γλώσσας παρέχεται η δυνατότητα παραγωγής προγραμμάτων ελεύθερων από ιούς. Κάποιοι από τους λόγους που καθιστούν τη Java ασφαλή γλώσσα έχουν να κάνουν με το ότι εκτελείται μέσα σε εικονική μηχανή. Παράλληλα μέσω του φορτωτή κλάσεων (classloader) ο οποίος είναι υπεύθυνος να φορτώσει τις κλάσεις της Java στην εικονική της μηχανή (Java Virtual Machine – JVM) παρέχεται ένα επιπλέον επίπεδο ασφάλειας αφού γίνεται διαχωρισμός των πακέτων που προέρχονται απο το τοπικό σύστημα αρχείων από αυτά που προέρχονται από πηγές του δικτύου. Τέλος μέσω του διαχειριστή ασφαλείας (Security Manager) αποφασίζεται σε ποιους πόρους έχει πρόσβαση μία κλάση, όπως είναι για παράδειγμα το διάβασμα και το γράψιμο αρχείων του σκληρού δίσκου, ένα χαρακτηριστικό που ενισχύει ακόμα περισσότερο την ασφάλεια της συγκεκριμένης γλώσσας.

Υψηλής Απόδοσης

Σε σχέση με άλλες γλώσσες προγραματισμού η Java χαρακτηρίζεται υψηλής απόδοσης και γρήγορη εξαιτίας του κώδικα που παράγει. Ο κώδικας byte (bytecode) της Java που παράγεται και εκτελείται στις διάφορες πλατφόρμες είναι πολύ κοντά στον πηγαίο κώδικα, κάτι που βοηθάει σε μεγάλο βαθμό την ταχύτητα εκτέλεσης.

Πολυνηματική

Ένα νήμα αντιμετωπίζεται συνήθως σαν ένα ξεχωριστό πρόγραμμα το οποίο μπορεί να εκτελεστεί ταυτόχρονα. Η Java δίνει τη δυνατότητα να παραχθούν προγράμματα που θα ασχολούνται με διάφορες λειτουργίες την ίδια χρονική στιγμή. Κάτι τέτοιο επιτυγχάνεται ορίζοντας πολλαπλά νήματα. Το κύριο πλεονέκτημα του πολυνηματισμού είναι ότι δεν δεσμεύεται ξεχωριστή μνήμη για κάθε νήμα, αλλά όλα τα νήματα μοιράζονται μία κοινή μνήμη.

Δυναμική

Η Java είναι μία δυναμική γλώσσα καθώς υποστηρίζει το δυναμικό φόρτωμα κλάσεων, δηλαδή να φορτώνονται οι κατάλληλες κλάσεις όταν αυτό απαιτείται. Παράλληλα υποστηρίζεται η δυναμική μεταγλώττιση καθώς και η αυτόματη διαχείριση μνήμης.

3.4 Περιβάλλοντα και Εργαλεία Ανάπτυξης Εφαρμογών

Οι εφαρμογές που αναπτύσσονται για το λογισμικό Android γράφονται στη γλώσσα προγραμματισμού Java. Τα εργαλεία του Software Development Kit - SDK μεταγλωττίζουν τον κώδικα μαζί με δεδομένα και αρχεία πόρων σε ένα πακέτο Android (Android package), το οποίο είναι ένα συμπιεσμένο αρχείο με την κατάληξη .apk. Ο κώδικας σε ένα .apk θεωρείται ως μια εφαρμογή και είναι το αρχείο που χρησιμοποιεί μια συσκευή για να εγκαταστήσει την εφαρμογή.

Για την ανάπτυξη επομένως μια ολοκληρωμένης εφαρμογής πέρα από τα εργαλεία ανάπτυξης, χρειάζεται και ενα ολοκληρωμένο περιβάλλον ανάπτυξης που να προσφέρει όσο το δυνατόν περισσότερες δυνατότητες στον υπεύθυνο για την ανάπτυξη της εφαρμογής. Ένα τέτοιο περιβάλλον είναι και το Android Studio.

3.4.1 Software Development Kit (SDK).

Ένα εργαλείο ανάπτυξης λογισμικού (SDK) αποτελεί το σύνολο των εργαλείων, βιβλιοθηκών, δειγμάτων κώδικα ή άλλων οδηγών που επιτρέπουν στους προγραμματιστές τη δημιουργία εφαρμογών λογισμικού σε μία συγκεκριμένη πλατφόρμα. Τα εργαλεία ανάπτυξης λογισμικού αποτελούν τις πηγές προέλευσης για σχεδόν κάθε πρόγραμμα με το οποίο μπορεί να αλληλεπιδράσει ένας σύγχρονος χρήστης.

3.4.2 Η εικονική μηχανή Java (Java Virtual Machine – JVM)

Η εικονική μηχανή Java αποτελεί πρακτικά μία εικονική μηχανή στην οποία εκτελούνται κλάσεις αρχείων java. Με βάση τον ορισμό της εικονικής μηχανής σημαίνει ότι στην ουσία πρόκειται για ένα στιγμιότυπο μιας πραγματικής μηχανής όπως είναι για παράδειγμα ο εξυπηρετητής (server) ενός συγκεκριμένου προγράμματος. Ανεξάρτητα από το λειτουργικό σύστημα ή το υποκείμενο υλικό (hardware) του εκάστοτε συστήματος, η εικονική μηχανή JVM δημιουργεί ένα κατάλληλο περιβάλλον για να εκτελεστούν τα προγράμματα, χωρίς ωστόσο να να δημιουργέι και ένα εικονικό λειτουργικό σύστημα, όπως θα έκανε μια πραγματική εικονική μηχανή.

Ο φορτωτής κλάσεων της εικονικής μηχανής Java, είναι υπεύθυνος να φορτωθούν οι κλάσεις στη μνήμη και να γίνουν διαθέσιμες για εκτέλεση. Κάθε εικονική μηχανή έχει έναν τετοιο. Αφού ολοκληρωθεί η διαδικασία φόρτωσης κλάσεων, η εικονική μηχανή ξεκινάει την εκτέλεση του κώδικα κάθε μίας απο αυτές. Η μηχανή εκτέλεσης είναι το συστατικό της εικονικής μηχανής που χειρίζεται αυτή τη λειτουργία, κάτι που όπως είναι εύκολα κατανοητό την καθιστά απαιραίτητη για τη σωστή λειτουργία της εικονικής μηχανής.

Η εκτέλεση ενός κώδικα, όπως είναι φυσικό περιλαμβάνει και τη διαχείριση της πρόσβασης στους πόρους του συστήματος. Η μηχανή εκτέλεσης παίζει το ρόλο του διαμεσολαβητή ανάμεσα στο εκτελέσιμο πρόγραμμα (μαζί με ό,τι απαιτήσεις για δίκτυο, μνήμη και πρόσβαση στο σύστημα αρχείων μπορεί να έχει αυτό), και το λειτουργικό σύστημα το οποίο είναι υπεύθυνο να παρέχει τους απαιτούμενους πόρους.

3.4.3 Συλλέκτης Απορριμάτων (Garbage Collector)

Ο συλλέκτης απορριμάτων είναι ένα πρόγραμμα που επιτυγχάνει την αυτόματη διαχείριση της μνήμης και οδηγεί στην αποδέσμευση αντικειμένων από την ίδια τη Java χωρίς να χρειάζεται η παρεμβολη του προγραμματιστή. Η Java υποστηρίζει τη δυναμική δέσμευση μνήμης από τα αντικείμενα της.

Κατα τη δημιουργία ενός αντικειμένου δεσμέυεται αυτόματα μνήμη, η οποία παραμένει σε αυτή την κατάσταση για όσο υπάρχουν αναφορές στο αντικέιμενο αυτό. Όταν πάψουν να υπάρχουν τέτοιες αναφορές γίνεται η υπόθεση ότι το αντικείμενο δεν είναι πλέον χρήσιμο και επομένως η δεσμευμένη μνήμη μπορεί να ανακτηθεί, χωρίς ωστόσο να καταστρέφεται το αντικείμενο αφού η Java υποστηρίζει την επαναδέσμευση μνήμης αυτόματα.

Η τεχνική που πετυχαίνει τα παραπάνω ονομάζεται Συλλογή Σκουπιδιών (Garbage Collection). Προγράμματα που δεν επαναδεσμεύουν μνήμη οδηγούνται τελικά σε τερματισμό (crash) όταν δεν υπάρχει άλλη διαθέσιμη μνήμη στο σύστημα. Τα προγράμματα αυτά έχουν τις λεγόμενες διαροές μνήμης (memory leaks).

Η συλλογή σκουπιδιών στη Java πραγματοποιείται κατά τη διάρκεια ζωής ενός προγράμματος, εξουδετερώνοντας έτσι την ανάγκη για επαναδέσμευση μνήμης και κατ’ επέκταση τις διαροές μνήμης.

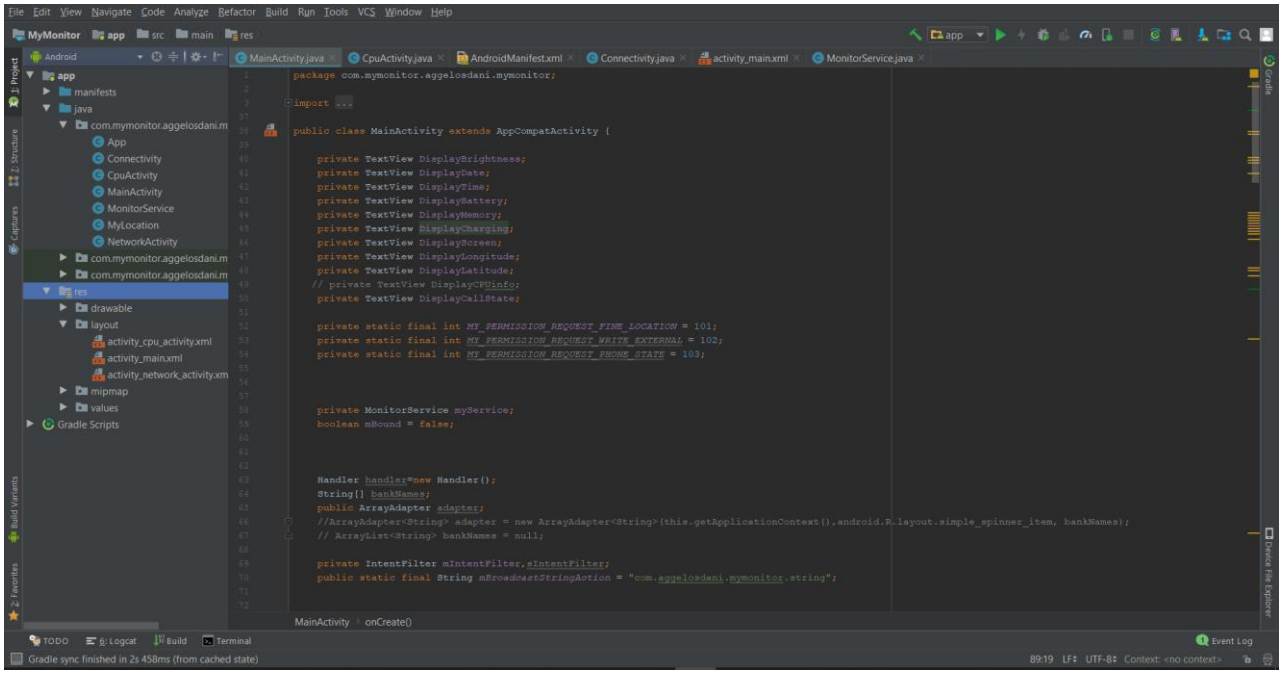
3.4.4 Περιβάλλον ανάπτυξης Android Studio

Για την ανάπτυξη εφαρμογών σε λειτουργικό σύστημα Android, είναι απαραίτητο να υπάρχει ένας υπολογιστής στον οποίο θα είναι δυνατή η εγκατάσταση του ολοκληρωμένου περιβάλλοντος ανάπτυξης Android Studio [37]. Το Android Studio είναι το επίσημο ολοκληρωμένο περιβάλλον ανάπτυξης (IDE - Integrated Development Environment) για το λειτουργικό σύστημα Android, το οποίο βασίζεται στο λογισμικό IntelliJ IDEA της JetBrains και έχει σχεδιαστεί ειδικά για αυτόν το σκοπό, δηλαδή για ανάπτυξη Android εφαρμογών. Επίσης το το συγκεκριμένο περιβάλλον αντικατέστησε το Eclipse Android Development Tools, το οποίο αποτελούσε το αρχικό περιβάλλον ανάπτυξης εφαρμογών Android.

Το περιβάλλον ανάπτυξης Android Studio παρέχει όλα τα εργαλεία που είναι αναγκαία για τη δημιουργία μίας εφαρμογής αλλα και για τη διαχείριση της. Μερικά από αυτά τα εργαλεία βοηθούν στη σχεδίαση και την υλοποίηση της διεπαφής του χρήστη (User Interface), στη διόρθωση σφαλμάτων που μπορεί να βρεθούν στον κώδικα (debugging) και σε πολλά άλλα. Όταν ο προγραμματιστής επιλέγει να εκτελέσει την εφαρμογή του έχει δύο επιλογές. Είτε να χρησιμοποιήσει τον Android Emulator, είτε να χρησιμοποιήσει μία κανονική Android συσκευή. Ο Androíd Emulator, ο οποίος περιέχεται στο πακέτο εγκατάστασης του προαναφερθέντος περιβάλλοντος ανάπτυξης, είναι ένας προσομοιωτής συσκευών Android που παρέχει στον προγραμματιστή τη δυνατότητα να βεβαιωθεί ότι η εφαρμογή που έχει δημιουργήσει λειτουργεί με τον τρόπο που ο ίδιος επιθυμεί. Αν η εφαρμογή λειτουργεί με τον επιθυμητό τρόπο, μπορεί η εκτέλεσή της να δοκιμαστεί σε επόμενο στάδιο απευθείας σε μία πραγματική συσκευή Android που είναι συνδεδεμένη με τον υπολογιστή μέσω καλωδίου usb, ή στην οποία έχει εγκατασταθεί εξαρχής το εκτελέσιμο αρχείο APK της εφαρμογής. Όπως είναι κατανοητό το καλύτερο περιβάλλον για να πραγματοποιηθεί η δοκιμή της εφαρμογής είναι σε μια κανονική συσκευή με λειτουργικό σύστημα Android.

Η βασική γλώσσα προγραμματισμού που χρησιμοποιείται στο Android Studio είναι. η Java. Επιπλέον, το Android Studio περιλαμβάνει τα εξής εργαλεία:

- visual layout editor: παρέχει τη δυνατότητα στον προγραμματιστή να σχεδιάσει. όλες τις διατάξεις της εφαρμογής χρησιμοποιώντας απλά τον κέρσορα του υπλογιστή και αποφεύγοντας με αυτόν τον τρόπο την υλοποίηση κώδικα σε xml μορφή.
- APK Analyzer: μειώνει όσο το δυνατόν περισσότερο το μέγεθος της εφαρμογής και παρακολουθεί κατ’ επέκταση τα στοιχεία του APK αρχείου.
- Intelligent code editor: παρέχει τη δυνατότητα προτάσεων και διορθώσεων στον προγραμματιστή κατά τη διάρκεια της συγγραφής του κώδικα.
- Flexible build system (build gradle): επιτρέπει τη δημιουργία του κατάλληλου εκτελέσιμου αρχείου έτσι ώστε να είναι δυνατή η εκτέλεση της εφαρμογής σε διαφορετικές Android συσκευές.
- Realtime profilers: παρέχει στοιχεία και στατιστικά για τη δραστηριότητα της μνήμης, του φόρτου εργασίας της Κεντρικής Μονάδας Επεξεργασίας καθώς και του δικτύου, δίνοντας έτσι στον προγραμματιστή μια γενικότερη εικόνα για τους πόρους που χρησιμοποιεί κατα την ανάπτυξη της εφαρμογης.



Εικόνα 8: Περιβάλλον ανάπτυξης Android Studio.

Κεφάλαιο 4

Ανάλυση Δεδομένων

4.1 Μηχανική Μάθηση

Η Μηχανική Μάθηση αποτελεί μια εφαρμογή της Τεχνητής Νοημοσύνης, η οποία επιτρέπει σε ένα σύστημα να εκπαιδεύεται αυτόματα καθώς και να βελτιώνεται χωρίς να είναι απαραίτητος ο συνεχής προγραμματισμός του [38]. Ο κύριος τρόπος με τον οποίο χρησιμοποιείται η μηχανική μάθηση αφορά την ανάπτυξη υπολογιστικών προγραμμάτων τα οποία έχουν πρόσβαση σε διάφορα δεδομένα. Τα συγκεκριμένα δεδομένα χρησιμοποιούνται στη συνέχεια από το εκάστοτε πρόγραμμα έτσι ώστε να εκπαιδευτεί πάνω σε αυτά.

Η διαδικασία εκμάθησης ξεκινάει με παρατηρήσεις ή δεδομένα, όπως είναι παραδείγματα, προσωπικές παρατηρήσεις, ή οδηγίες, έτσι ώστε να είναι επιτυχής η αναζήτηση μοτίβων στα εισερχόμενα δεδομένα και να παρθούν καλύτερες αποφάσεις με βάση αυτά. Ο απώτερος σκοπός αυτής της διαδικασίας είναι να επιτραπεί στους υπολογιστές να εκπαιδευόνται αυτόματα χωρίς την ανθρώπινη ανάμειξη καθώς και να προσαρμόζονται ανάλογα με τις περιστάσεις.

Η μηχανική μάθηση παρέχει τη δυνατότητα ανάλυσης μεγάλου όγκου δεδομένων. Παρόλο που είναι εφικτή η γρήγορη και ακριβής παραγωγή αποτελεσμάτων με σκοπό να βρεθούν κερδοφόρες ευκαιρίες ή να γίνει ανάλυση κινδύνων, είναι πιθανό να χρειάζεται να δοθεί αρκετό χρονικό διάστημα σε τέτοιου είδους προγράμματα έτσι ώστε να εκπαιδευτούν σωστά.

4.4.1 Μη Επιβλεπόμενη Μάθηση

Η μη επιβλεπόμενη μάθηση χρησιμοποιείται όταν τα δεδομένα που χρησιμοποιούνται δεν είναι ουτε κατηγοριοποιημένα, ούτε επισημασμένα [38]. Ο κύριος σκοπός της είναι να μελετήσει πως μπορεί το εκάστοτε σύστημα να παράξει μία συνάρτηση, η οποία να περιγράφει μία κρυφή δομή από τα μη επισημασμένα δεδομένα. Σκοπός του συστήματος επομένως δεν είναι να παράξει μία σωστή έξοδο, αλλά να εξερευνήσει τα δοσμένα δεδομένα έτσι ώστε να προκύψουν συμπεράσματα και αλληλουχίες που θα είναι σε θέση να περιγράψουν τις προαναφερθείσες δομές των μη επισημασμένων δεδομένων.

4.1.2 Επιβλεπόμενη Μάθηση

Η επιβλεπόμενη μάθηση απο την άλλη, είναι σε θέση να εφαρμόσει όλη την πληροφορία που έχει εξάγει στο παρελθόν σε νέα δεδομένα, χρησιμοποιώντας επισημασμένα παραδείγματα για να κάνει τις ζητούμενες προβλέψεις [38]. Ξεκινώντας από την ανάλυση ενός συνόλου δεδομένων εκμάθησης, οι χρησιμοποιούμενοι αλγόριθμοι παράγουν μία συνάρτηση με σκοπό να κάνουν προβλέψεις για τις τιμές εξόδου. Το σύστημα είναι σε θέση να παρέχει νέες μεταβλητές στόχου (target variables) για κάθε καινούρια είσοδο. Τέλος οι αλγόριθμοι μηχανικής μάθησης είναι σε θέση να συγκρίνουν την παραγόμενη έξοδο με την προβλεπόμενη έξοδο έτσι ώστε να γίνονται οι κατάλληλες συγκρίσεις και να τροποποιούνται για να προσφέρουν ένα καλύτερο αποτέλεσμα.

Στην συγκεκριμένη διπλωματική εργασία χρησιμοποιείται η επιβλεπόμενη μάθηση και συγκεκριμένα το πρόβλημα της ταξινόμησης (classification).

4.2 Ταξινόμηση

Η ταξινόμηση είναι μια ευρέως χρησιμοποιούμενη τεχνική που έχει ως στόχο την κατασκευή ενός μοντέλου με σκοπό την πρόβλεψη της αξίας ενός εξαρτώμενου χαρακτηριστικού μέσα από μερικές ανεξάρτητες μεταβλητές [38]. Ένα πρόβλημα ταξινόμησης είναι όταν η μεταβλητή εξόδου είναι μία διακριτή κατηγορία, όπως "κόκκινο" ή "μπλε" για παράδειγμα. Ένα μοντέλο ταξινόμησης επιχειρεί να καταλήξει σε κάποιο συμπέρασμα από τις παρατηρούμενες τιμές, συνεπώς λαμβάνοντας υπόψη μία ή περισσότερες εισόδους, ένα μοντέλο ταξινόμησης θα προσπαθήσει στη συνέχεια να προβλέψει την αξία ενός ή περισσοτέρων αποτελεσμάτων.

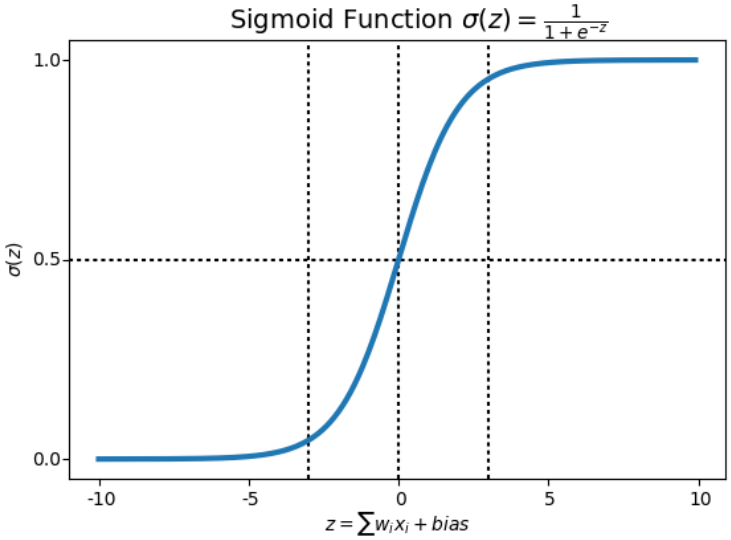
4.3 Αλγόριθμοι Ταξινόμησης

Για τους σκοπούς της συγκεκριμένης εργασίας χρησιμοποιήθηκαν ποικίλοι αλγόριθμοι για το πρόβλημα της ταξινόμησης. Έγινε χρήση λογιστικής παλινδρόμησης, K-πλησιέστερων γειτόνων, δέντρων απόφασης, τυχαίων δασών καθώς και SVM, Kernel SVM.

4.3.1 Λογιστική Παλινδρόμηση

Η λογιστική παλινδρόμηση (Logistic Regression) αποτελεί έναν αλγόριθμο μηχανικής μάθησης που χρησιμοποιείται σε προβλήματα ταξινομησής. Στην ουσία αποτελεί έναν αλγόριθμο πρόβλεψης, ο οποίος βασίζεται κυρίως στην έννοια της πιθανότητας.

Η σημαντικότερη διαφορά της λογιστικής παλινδρόμησης με τη γραμμική παλινδρόμηση, είναι οτι στην πρώτη περίπτωση χρησιμοποιείται μια πιο σύνθετη συνάρτηση κόστους, η οποία ονομάζεται σιγμοειδής συνάρτηση (Sigmoid function).



Εικόνα 9: Σιγμοειδής Συνάρτηση,

Η συγκεκριμένη συνάρτηση χρησιμοποιείται για να μετατρέπει τις παραγόμενες τιμές σε πιθανότητες. Αυτό το καταφέρνει με το να μετατρέπει οποιαδήποτε πραγματική τιμή σε μία τιμή ανάμεσα στο 0 και το 1. Το συγκεκριμένο διάστημα στο οποίο πρέπει να περιοριστεί η συνάρτηση κόστους είναι και ο κύριος λόγος που χρησιμοποιείται λογιστική παλινδρόμηση, καθώς οι γραμμικές συναρτήσεις είναι δυνατόν να πάρουν και τιμές μεγαλύτερες του 1 ή μικρότερες του 0 [39].

4.3.2 Κ-Πλησιέστεροι Γείτονες (KNN)

Ο αλγόριθμος των Κ-Πλησιέστερων Γειτόνων (K-Nearest Neighbor) αποτελεί έναν αλγόριθμο ταξινόμησης, με κύριο μέλημα του να αποφασίσει σε ποια ομάδα ανήκει ένα δεδομένο, εξετάζοντας τα δεδομένα που βρίσκονται γύρω του.

Ο συγκεκριμένος αλγόριθμος στην ουσία εξετάζει ένα σημείο σε ένα πλέγμα και στη συνέχεια προσπαθεί κατηγοριοποιήσει το συγκεκριμένο σημείο σε μία ομάδα Α ή Β ανάλογα με τις καταστάσεις των υπόλοιπων σημείων που βρίσκονται γύρω του. Για να το πετύχει αυτό χρησιμοποιεί τις λεγόμενες συναρτήσεις απόστασης, οι οποίες φαίνονται παρακάτω:

- **Ευκλείδια Απόσταση**

$$\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$$

- **Απόσταση Manhattan**

$$\sum_{i=1}^{k}|x_i - y_i|$$

- **Απόσταση Minkowski**

$$\left(\sum_{i=1}^{k}(|x_i - y_i|)^q\right)^{1/q}$$

Αξίζει να σημειωθεί οτι οι συγκεκριμένες συναρτήσεις μπορούν να χρησιμοιποιηθούν μόνο στην περίπτωση που οι μετάβλητες παίρνουν συνεχόμενες τιμές. Αν οι μεταβλητές παίρνουν διακριτές τιμές τότε χρησιμοποιείται η συνάρτηση απόστασης Hamming, η οποία φαίνεται στον παρακάτω τύπο:

- **Απόσταση Hamming**

$$D_H = \sum_{i=1}^{k}|x_i - y_i|,$$

$$\text{x = y} \Rightarrow D = 0,$$

$$x \neq y \Rightarrow D = 1$$

Ο αλγόριθμος KNN χαρακτηρίζεται ως αλγόριθμος «τεμπέλικης» μάθησης, αφού δεν παράγει ένα μοντέλο για το δοσμένο σύνολο δεδομένων, παρά μόνο υπολογίζει αποστάσεις οταν του ζητείται μεταξύ του εξεταζόμενου σημείου και των υπολοίπων γειτόνων του [40].

4.3.3 Δέντρα απόφασης

Τα Δέντρα Απόφασης (Decision Trees), αποτελούν εργαλείο παραγωγής προβλέψεων το οποίο έχει εφαρμογή σε πολλούς διαφορετικούς τομείς. Σε γενικές γραμμές τα δέντρα απόφασης βαζίζονται στην αλγοριθμική προσέγγιση αναγνώρισης τρόπων

διαχωρισμού ενός συνόλου δεδομένων με βάση διαφορετικές συνθήκες. Αποτελούν μια ευρέως δημοφιλή και χρησιμοποιούμενη μέθοδο για επιβλεπόμενη μάθηση σε προβλήματα ταξινόμησης. Ο κυριότερος στόχος αυτού του αλγορίθμου είναι η δημιουργία ενός μοντέλου που θα προβλέπει την τιμή μιας μεταβλητής στόχου, μέσω της εκμάθησης απλών κανόνων απόφασης που θα προκύπτουν από τα χαρακτηριστικά του συνόλου των δεδομένων.

Τα δέντρα απόφασης βασίζονται στην τεχνική των γράφων με τη μορφή δέντρου, όπου προκύπτει μία ερώτηση με βάση κάποιο συγκεκριμένο χαρακτηριστικό. Οι άκρες του γράφου αποτελούν την απάντηση στην ερώτηση ενώ τα φύλλα αντιπροσωπεύουν την πραγματική ετικέτα εξόδου.

Τα δέντρα απόφασης κατηγοριοποιούν τα παραδείγματα ταξινομώντας τα από τη ρίζα του δέντρου μέχρι κάποιο κόμβο φύλλου. Κάθε κόμβος του δέντρου αποτελεί μία περίπτωση ελέγχου για κάποιο χαρακτηριστικό και κάθε ακμή που ξεκινάει από αυτόν τον κόμβο αποτελεί μια πιθανή απάντηση στη συγκεκριμένη περίπτωση. Η διαδικασία αυτή είναι αναδρομική και επαναλαμβάνεται για κάθε υποδέντρο που έχει ρίζες στους νέους κόμβους [41].

4.3.4 Τυχάιο Δάσος

Το τυχαίο δάσος (Random Forest), όπως υποδηλώνει και το όνομα του, αποτελείται από ένα μεγάλο αριθμό ατομικών δέντρων απόφασης τα οποία λειτουργούν ως σύνολο. Κάθε μεμονωμένο δέντρο μέσα στο τυχαίο δάσος παράγει μία πρόβλεψη στο πρόβλημα κατηγοριοποίησης και η πρόβλεψη με τις περισσότερες ψήφους καταλήγει στο τέλος να αποτελεί και την έξοδο του αλγορίθμου αυτού.

Ο λόγος που το τυχαίο δάσος έχει τόσο μεγάλη εφαρμογή στην επιστήμη των δεδομένων έχει να κάνει με το ότι ένας μεγάλος αριθμός από σχετικά μη συσχετιζόμενα μοντέλα (δέντρα στην προκειμένη περίπτωση), τα οποία λειτουργούν ως σύνολο, θα παράξουν καλύτερο αποτέλεσμα από τα επιμέρους μεμονωμένα μοντέλα.

Η χαμηλή συσχέτιση μεταξύ των μοντέλων είναι και ο σημαντικότερος παράγοντας για την επιτυχία της πρόβλεψης. Το παράδοξο αυτό γεγονός έχει να κάνει με το ότι τα επιμέρους δέντρα αλληλοκαλύπτονται ώς προς το αποτέλεσμα της πρόβλεψης. Ενώ για παράδειγμα κάποια δέντρα θα δώσουν λάθος αποτέλεσμα, πολλά θα είναι εκείνα που θα δώσουν σωστή πρόοβλεψη με αποτέλεσμα το σύνολο των δέντρων να κινείται προς τη σωστή κατεύθυνση.

Οι σημαντικότεροι παράγοντες επομένως για να λειτουργεί σωστά ένας αλγόριθμος τυχαίου δάσους είναι οι ακόλουθοι:

1. Πρέπει να υπάρχει κάποια πραγματική πληροφορία στα χαρακτηριστικά που αποτελούν το σύνολο των δεδομένων εκμάθησης, έτσι ώστε τα μοντέλα που

κατασκευάζονται χρησιμοποιώντας τα συγκεκριμένα χαρακτηριστικά να μην παράγουν τυχαίες προβλέψεις.
2. Οι προβλέψεις αλλά και τα σφάλματα που γίνονται απο τα μεμονωμένα δέντρα απόφασης πρέπει να έχουν χαμηλή συσχέτιση μεταξύ τους [42].

4.3.5 Μηχανές Διανυσμάτων Υποστήριξης - SVMs

Οι μηχανές διανυσμάτων υποστήριξης (Support Vector Machines - SVMs) αποτελούν ένα μοντέλο επιβλεπόμενης μηχανικής μάθησης, που χρησιμοποιείται τόσο στο πρόβλημα της ταξινόμησης, όσο και της παλινδρόμησης, με κύριο σκοπό να χωρίσει τα δεδομένα σε δύο κατηγορίες. Ο συγκεκριμένος αλγόριθμος εκπαιδεύεται σε σειρές δεδομένων που είναι ήδη κατηγοριοποιημένες, δημιουργώντας έτσι το μοντέλο όπως έχει αρχικά εκπαιδευτεί και στη συνέχεια ταξινομεί κάθε νέο σημείο δεδομένων σε μία απο τις δύο κατηγορίες. Το γεγονός αυτό καθιστά τις μηχανές διανυσμάτων υποστήριξης ένα είδος μη δυαδικού γραμμικού ταξινομητή.

Πιο συγκεκριμένα, ένα τέτοιο μοντέλο χαρτογραφεί τα δεδομένα στο χώρο έτσι ώστε τα δεδομένα των δυο κατηγοριών να χωρίζονται από ένα σαφές κενό που είναι όσο το δυνατόν μεγαλύτερο. Τα καινούρια δεδομένα έτσι χαρτογραφούνται και αυτά στον ίδιο χώρο και πρόβλεπεται σε ποια κατηγορία θα ανήκουν ανάλογα με τη μεριά του χάσματος στην οποία έχουν καταλήξει [43].

4.4 Η γλώσσα προγραμματισμού Python

Η Python είναι γλώσσα προγραμματισμού η οποία χρησιμοποιεί διερμηνευτή αντι για μεταγλωττιστή. Δημιουργήθηκε το 1990 από τον Ολλανδό προγραμματιστή Guido van Rossum, ακολουθώντας την εμπειρία που είχε πάνω στη γλώσσα εκπαιδευτικού σκοπού ABC στο πανεπιστήμιο CWI. Η Python διαφέρει από άλλες γλώσσες προγραμματισμού, καθώς δίνει προτεραιότητα στην αναγνωσιμότητα του κώδικα και στη χρήση όστο το δυνατόν μικρότερων αρχείων [44].

Η Python χρησιμοποιεί δυναμικούς τύπους δεδομένων καθώς και την τεχνική συλλογής σκουπιδιών. Υποστηρίζει πλήρως τον αντικειμενοστρεφή και δομημένο προγραμματισμό καθώς και σε μεγάλο βαθμό το λειτουργικό και προσανατολισμένο σε μορφή προγραμματισμό. Με αυτόν τον τρόπο καθίσταται εφικτή η παραγωγή κώδικα ιδιαίτερα ευπροσάρμοστου και εφαρμόσιμου σε μια ευρεία ποικιλία περιπτώσεων χρήσης.

Η βασική βιβλιοθήκη της Python θεωρείται ένα απο τα βασικότερα πλεονεκτήματα της. Αυτό το χαρακτηριστικό επιτρέπει στους προγραμματιστές να αναπτύσσουν ολοκληρωμένα προγράμματα γρήγορα και χωρίς να χρειάζεται να βασίζονται σε μεγάλο βαθμό σε πακέτα άλλων κατασκευαστών για τις βασικές ανάγκες μιας συγκεκριμένης εφαρμογής. Χαρακτηριστικό είναι το γεγονός οτι ο κατάλογος πακέτων Python Package Index (PyPI) καταγράφει πάνω από 200.000 πακέτα που παρέχουν διάφορες λειτουργίες.

Παρακάτω παρουσιάζονται οι βιβλιοθήκες της Python, ο τύπος του αρχείου αποθήκευσης δεδομένων, καθώς και το προγραμματιστικό περιβάλλον που χρησιμοποιήθηκε για την ανάλυση των δεδομένων [45].

4.4.1 SciPy

Η SciPy είναι ένα λογισμικό ελεύθερου λογισμικού (open source) που έχει εφαρμογές στα μαθηματικά καθώς και στην επιστήμη των υπολογιστών. Η συγκεκριμένη βιβλιοθήκη μπορεί να αποδειχθεί χρήσιμη αν βασικό μέλημα είναι ο χειρισμός αριθμών με τη βοήθεια ενός υπολογιστή ή η προβολή των αποτελεσμάτων των προαναφερθέντων αριθμών.

4.4.2 Numpy

Η NumPy (Numerical Python) αποτελεί επίσης μια βιβλιοθήκη ελεύθερου λογισμικού η οποία προσφέρεται για τον εύκολο και γρήγορο χειρισμό πολυδιάστατων πινάκων. Αξιοσημείωτο είναι οτι η βιβλιοθήκη SciPy σχεδιάστηκε έτσι ώστε να ενσωματώνεται μαζί με πίνακες Numpy.

4.4.3 Pandas

Το Pandas είναι ακόμη μια βιβλιοθήκη ελεύθερου λογισμικού που προσφέρει γρήγορες και ευέλικτες δομές δεδομένων, οι οποίες βρίσκουν εφαρμογή κυρίως σε χρονοσειρές. Το συγκεκριμένο πακέτο αποτελεί ένα σημαντικό εργαλείο για την ανάλυση δεδομένων με τη χρήση της γλώσσας Python.

4.4.4 Scikit-learn

Το Scikit-learn αποτελεί άλλη μια βιβλιοθήκη της Python η οποία παρέχει αλγορίθμους τόσο επιβλεπόμενης όσο και μη επιβλεπόμενης μάθησης. Σχεδιάστηκε βασιζόμενη στις παραπάνω τεχνολογίες (SciPy, Numpy, Pandas) καθώς και στη Matplotlib. Μέσω της συγκεκριμένης βιβλιοθήκης είναι εφικτή η επίλυση προβλημάτων ταξινόμησης, παλινδρόμησης, ομαδοποίησης, επιλογής μοντέλου καθώς και προεπεξεργασίας των δεδομένων.

4.4.5 Matplotlib

Η συγκεκριμένη βιβλιοθήκη ελεύθερου λογισμικού χρησιμοποιείται για την απεικόνιση σε γραφήματα μεγάλου όγκου δεδομένων χωρίς την ανάγκη για μεγάλο σε έκταση κώδικα προκειμένου να την ενσωματώσει κάποιος. Βασίζεται κυρίως σε πίνακες Numpy ενώ είναι εφικτό να λειτουργεί με το ευρύτερο πακέτο της SciPy.

4.4.6 CSV

Το CSV αποτελεί ένα αρχείο κειμένου το οποίο χρησιμοποιείται για την απεικόνιση μεγάλων στηλών δεδομένων, οι οποίες χωρίζονται μεταξύ τους με κόμμα (Comma Separated Values - CSV).

4.4.7 Spyder

Το Spyder είναι ένα ανοικτού κώδικα ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) για επιστημονικό προγραμματισμό στη γλώσσα Python. Το Spyder ενσωματώνεται με μια σειρά προεξέχοντων πακέτων όπως είναι η Python, συμπεριλαμβανομένων των NumPy, SciPy, Matplotlib, Pandas, IPython, SymPy και Cython, καθώς και άλλων λογισμικών ανοιχτού κώδικα.

Αρχικά δημιουργήθηκε και αναπτύχθηκε από τον Pierre Raybaut το 2009, από το 2012 το Spyder διατηρείται και βελτιώνεται συνεχώς από μια ομάδα προγραμματιστών της Python και την επιστημονική κοινότητα.

Το Spyder είναι επεκτάσιμο με plugins ενσωματωμένα είτε απο άλλους παρόχους, περιλαμβάνει υποστήριξη για διαδραστικά εργαλεία για την επιθεώρηση δεδομένων και ενσωματώνει μέσα διασφάλισης της ποιότητας του κώδικα καθώς και μέσα ενδοσκόπησης όπως Pyflakes, Pylint και Rope. Διατίθεται σε πολλαπλές πλατφόρμες μέσω Anaconda, σε Windows, MacOS μέσω MacPorts και σε μεγάλες διανομές Linux όπως το Arch Linux, το Debian, το Fedora, το Gentoo Linux, το openSUSE και το Ubuntu.

Κεφάλαιο 5

Συλλογή, Προετοιμασία και Επεξεργασία Δεδομένων

Η παρούσα διπλωματική αποτελείται από δύο κομμάτια υλοποίησης. Στο πρώτο κομμάτι πραγματοποιήθηκε υλοποίηση εφαρμογής σε γλώσσα προγραμματισμού Java. Η εφαρμογή αυτή χρησιμοποιήθηκε για τη συλλογή δεδομένων από κινητό τερματικό, τα οποία δεδομένα αποθηκεύονταν σε αρχείο τύπου csv. Στο δεύτερο κομμάτι έγινε χρήση της γλώσσας προγραμματισμού Python, η οποία συγκεκριμένα χρησιμοποιήθηκε για την επεξεργασία και ανάλυση των συλλεχθέντων δεδομένων καθώς και για την εφαμοργή των αλγορίθμων μηχανικής μάθησης.

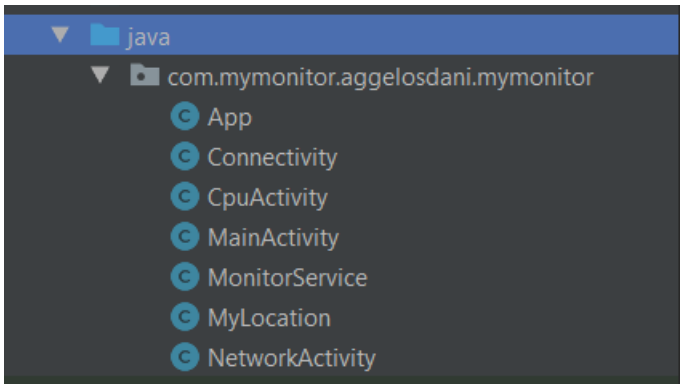
Στη συγκεκριμένη ενότητα περιγράφεται η σχεδίαση και η λειτουργικότητα της εφαρμογής Android που υλοποιήθηκε στα πλαίσια της Διπλωματικής και η οποία αποτελεί και το εργαλείο συλλογής των δεδομένων καθώς και η διαδικασία ανάκτησης, αποθήκευσης, επεξεργασίας δεδομένων όπως επίσης και η διαδικασία εξαγωγής χαρακτηριστικών (features) για την εφαρμογή των αλγορίθμων μηχανικής μάθησης.

5.1 Δραστηριότητες και Διατάξεις

Η συγκεκριμένη εφαρμογή αποτελείται από τρεις διαφορετικές διεπαφές χρήστη. Για τη δημιουργία μιας διεπαφής χρήστη ο προγραμματιστής είναι αναγκαίο να χρησιμοποιήσει Δραστηριότητες, ο ορισμός των οποίων εχει περιγραφεί στο Κεφάλαιο 3. Στη συγκεκριμένη εφαρμογή υπάρχουν τρεις διαφορετικές οθόνες, το οποίο μεταφράζεται σε τρεις διαφορετικές δραστηριότητες. Το αρχείο της κάθε δραστηριότητας περιέχει τον κώδικα σε java που τρέχει όταν γίνεται χρήση της εκάστοτε οθόνης. Παράλληλα. για κάθε μία από αυτες τις τρεις οθόνες υπάρχουν και τρία διαφορετικά layouts τα οποία περιέχουν όλα τα γραφικά αντικείμενα που φορτώνονται στην οθόνη κάθε φορά που τρέχει κάποια δραστηριότητα. Οι προαναφερθείσες διεπαφές χρήστη (οθόνες) είναι οι εξής: main screen, cpu screen και networks screen.

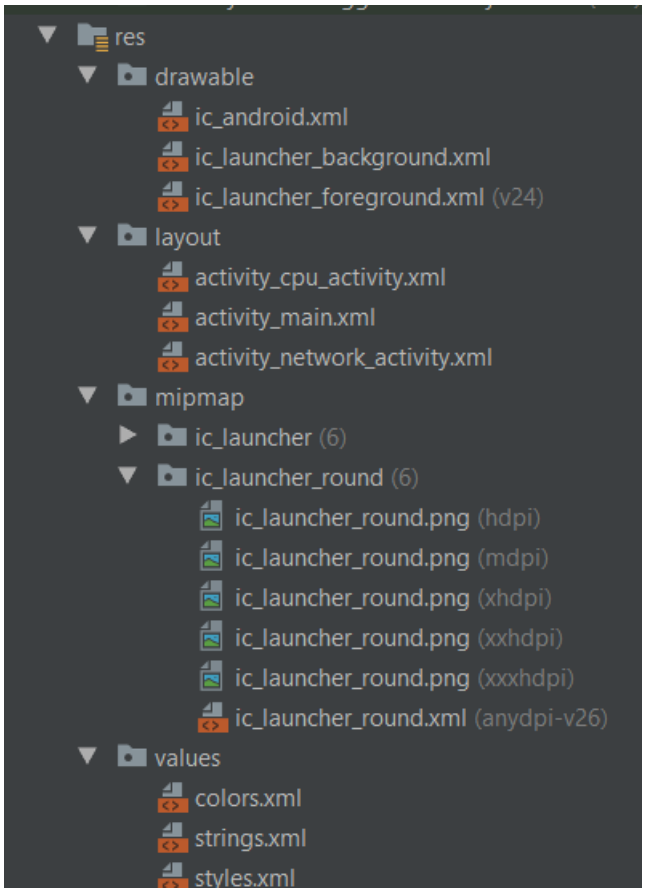
Όλες οι κλάσεις των δραστηριοτήτων καθώς και όλα τα αρχεία σε java που αφορούν την εκτέλεση και τη λειτουργικότητα της εφαρμογής περιέχονται στον φάκελο “java”

του Android Project (Εικόνα 10).



Εικόνα 10: Κλάσεις Δραστηριοτήτων.

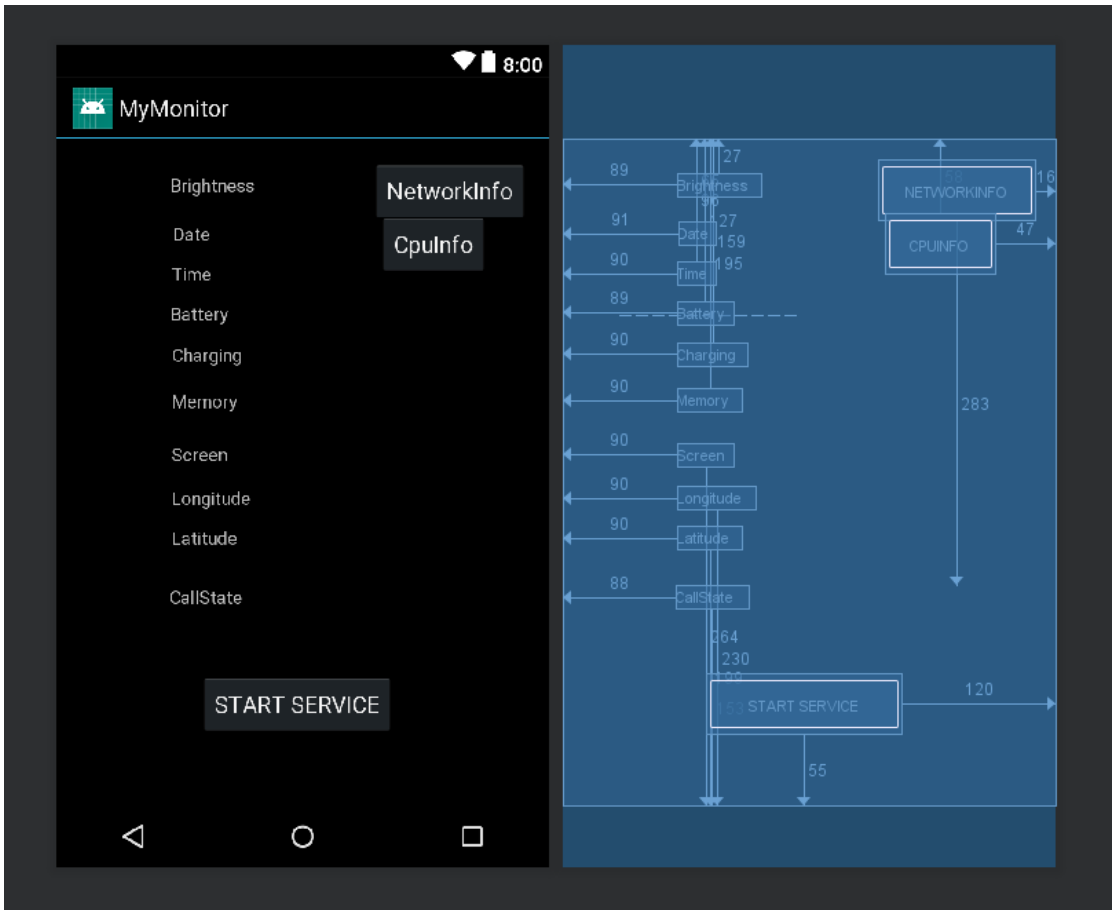
Όλες οι διατάξεις βρίσκονται μέσα στον ομότιτλο φάκελο ο οποίος είναι τοποθετημένος στον φάκελο "res"του Android Project.



Εικόνα 11: Σύνολο διατάξεων.

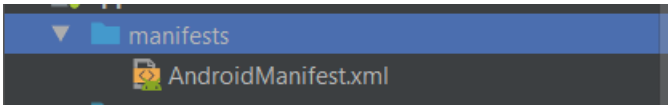
Παράλληλα μέσα στον φάκελο "res" υπάρχει επίσης ο φάκελος "drawable" που

περιέχει όλες τις εικόνες που ειναι διαθέσιμες και εμφανίζονται στην εφαρμογή και μάλιστα σε διαφορετικές αναλύσεις ώστε να είναι δυνατή πάντοτε η επιλογή της καταλληλότερης ανάλογα με τις διαστάσεις της οθόνης της συσκευής στην οποία εκτελείται η εφαρμογή. Παρόμοια, ο φάκελος “mipmap” περιέχει και αυτός τα εικονίδια της εφαρμογής σε πολλές αναλύσεις, για τον ίδιο λόγο που συμβαίνει αυτό και στον φάκελο “drawable”, ενώ υπάρχει και ο φάκελος “values” μέσα στον οποίο περιέχονται τα βασικά χρώματα της εφαρμογής σε αρχεία της μορφής .xml, ώστε να καθίσταται δυνατή η τροποποίηση τους για όλη την εφαρμογή από το αρχείο αυτό. Στον ίδιο φάκελο περιέχονται βασικές συμβολοσειρές της εφαρμογής καθώς επίσης και τα στυλ και θέματα που χρησιμοποιούνται σε διάφορα τμήματα της εφαρμογής.



Εικόνα 12: Αρχική οθόνη – Δομή διάταξης όψεων.

Κάθε δραστηριότητα καθώς και όλες οι βασικές πληροφορίες που την αφορούν δηλώνονται στο αρχείο AndroidManifest.xml, το οποίο βρίσκεται στον φάκελο “manifests”.



Εικόνα 13: Φάκελος Manifest αρχείων.

Σημαντική είναι η δήλωση της δραστηριότητας που εκτελείται κατά την εκκίνηση της εφαρμογής. Η παράλειψη της δήλωσης της συγκεκριμένης δραστηριότητας στο αρχείο αυτό σημαίνει και τη μη δυνατή εκτέλεση της εφαρμογής.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.mymonitor.aggelosdani.mymonitor">

    <uses-permission android:name="android.permission.FOREGROUND_SERVICE"/>
    <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
    <uses-permission android:name="android.permission.WRITE_INTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
    <uses-permission android:name="android.permission. ACCESS_COARSE_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.CHANGE_WIFI_STATE" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
    <uses-permission android:name="android.permission.READ_PHONE_STATE" />
    <uses-feature android:name="android.hardware.location.gps" />
    <uses-permission android:name="android.permission.WAKE_LOCK" />
    <application
        android:name=".App"
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <service
            android:name=".MonitorService"
            android:exported="false" />

        <activity android:name=".NetworkActivity"
            android:parentActivityName=".MainActivity">
            <!-- The meta-data tag is required if you support API level 15 and lower -->
            <meta-data
                android:name="android.support.PARENT_ACTIVITY"
                android:value=".MainActivity" />


        </activity>
```

```
        <activity android:name=".CpuActivity"
            android:parentActivityName=".MainActivity">
            <!-- The meta-data tag is required if you support API level 15 and lower -->
            <meta-data
                android:name="android.support.PARENT_ACTIVITY"
                android:value=".MainActivity" />


        </activity>
    </application>

</manifest>
```

Κάθε διάταξη (Layout) που φορτώνεται κατά την έναρξη μιας δραστηριότητας χαρακτηρίζεται από τα σύνολα των όψεων που περιέχει καθώς και από τον τρόπο που αυτές οι όψεις συνδέονται μεταξύ τους και απεικονίζονται στο χώρο, δηλαδή στην οθόνη της έξυπνης συσκευής.

Στην εν λόγω εφαρμογή έγινε κατά βάση η χρήση της διάταξης Constraint Layout (Εικόνα 12) το οποίο προτείνεται από τους ίδιους τους δημιουργούς ως το πιο αποδοτικό και αποκρίσιμο. Το Constraint Layout πρόκειται για έναν πιο αυστηρό τρόπο απεικόνισης αφού πρέπει όλες οι χρησιμοποιούμενες όψεις να συνδέονται με δεσμούς τόσο μεταξύ τους όσο και με τα πλαίσια της οθόνης, έτσι ώστε να διατηρούν μια προκαθορισμένη θέση ανεξάρτητη απο το μέγεθος της οθόνης.

5.2 Χρήση Βασικών Στοιχείων Android

Μέσα στη διάταξη της κάθε οθόνης μπορούν να περιέχονται άλλα μοτίβα απεικόνισης ή άλλα βασικά στοιχεία που απαρτίζουν το μεγαλύτερο κομμάτι των οθονών όλων των Android εφαρμογών. Αυτά περιγράφονται συνοπτικά παρακάτω:

- **TextView**

  Η όψη αυτή αποτελεί στην ουσία οποιαδήποτε μορφή κειμένου εμφανίζεται στην οθόνη της έξυπνης συσκευής του χρήστη. Τα πλαίσια μέσα στα οποία εμφανίζεται το συγκεκριμένο κείμενο μπορούν να έχουν μια αρχικοποιημένη τιμή, η οποία ορίζεται σε αρχείο τύπου .xml και μπορεί να μεταβάλλεται δυναμικά μέσα απο τον κώδικα που εκτελείται οταν μια δραστηριότητα είναι στο προσκήνιο.

- **Buttons**

  Τα κουμπιά αποτελούν ένα βασικό τρόπο αλληλεπίδρασης του χρήστη με την εφαρμογή. Το πλαίσιο μέσα στο οποίο ορίζεται η ιδιότητα του κουμπιού μπορεί να είναι κείμενο καθώς και εικόνα. Μια βασική λειτουργία των κουμπιών είναι για παράδειγμα η έναρξη μιας καινούριας δραστηριότητας που στα μάτια του

χρήστη θα σημαίνει το φόρτωμα μιας καινούριας οθόνης στην έξυπνη συσκευή του.

- **EditText**

  Το EditText είναι ένα πεδίο που επιτρέπει στο χρήστη την εισαγωγή κειμένου. Το κείμενο αυτό στη συνέχεια θα αποθηκευτεί σε μια μεταβλητή και θα χρησιμοποιηθεί απο την εφαρμογή.

- **ImageView**

  Η συγκεκριμένη όψη επιτρέπει την εμφάνιση μιας εικόνας ανάλογα με τη δραστηριότητα που εκτελείται εκείνη τη στιγμή. Μέσω αυτής της όψης είναι δυνατή για παράδειγμα και η απεικόνιση του λογότυπου της εφαρμογής.

  Spinners

  Τα Spinners αποτελούν ένα στοιχείο απεικόνισης που επιτρέπει τη χρήση μια λίστας drop-down. Η τιμή που εμφανίζεται στην οθόνη είναι συνήθως η πρώτη διαθέσιμη απο τη λίστα, ενώ ταυτόχρονα παρέχεται η δυνατότητα στο χρήστη, αγγίζοντας το συγκεκριμένο πεδίο με τη χρήση της αφής να επιλέξει κάποια διαφορετική τιμή απο αυτές που φορτώνονται δυναμικά. Η νέα τιμή που θα επιλέξει ο χρήστης θα είναι και αυτή που θα εμφανίζεται τελικά στην οθόνη.

- **ScrollView**

  Η συγκεκριμένη όψη χρησιμοποιείται για να παρέχεται η δυνατότητα στο χρήστη να διατρέχει ένα σύνολο δεδομένων στην οθόνη τα οποία δεν είναι εφικτό να απεικονιστούν στο σύνολο των υπάρχοντων ιντσών της οθόνης. Η συγκεκριμένη όψη μπορεί να συνδυαστεί με πολλούς διαφορετικούς τρόπους απεικόνισης των δεδομένων.

- **ListView**

  Το ListView αποτελεί μια όψη του Android λειτουργικού που επιτρέπει την ομαδοποιήση διαφορετικών αντικειμένων και την απεικόνικη τους σε μια κάθετη λίστα την οποία μάλιστα μπορεί ο χρήστης να διατρέξει μέσω της αφής που παρέχει η έξυπνη συσκευή. Τα συγκεκριμένα αντικείμενα εισέρχονται αυτόματα στη λίστα με τη χρήση ενός Adapter, χάρη στον οποίο είναι εφικτή η άντληση δεδομένων απο διαφορετικές δομές, όπως για παράδειγμα ένα πίνακα ή μια βάση δεδομένων.

- **GridView**

  Το GridView αποτελεί μια όψη που επιτρέπει την απεικόνιση ενός πλέγματος δυο διαστάσεων στην οθόνη καθώς και η δυνατότητα να διατρέξει ο χρήστης το συγκεκριμένο πλέγμα. Ο τρόπος με τον οποίο συλλέγονται τα δεδομένα είναι μέσω του ListAdapter.

Στη συγκεκριμένη εφαρμογή χρησιμοποιήθηκαν TextViews και Buttons καθώς η βασική λειτουργικότητα της εφαρμογής στηρίζεται όχι τόσο στην απεικόνιση

στοιχείων στην οθόνη της έξυπνης συσκευής αλλά στις διεργασίες που εκτελούνται στο παρασκήνιο.

5.3 Επικοινωνία μεταξύ Δραστηριοτήτων

Οι διεπαφές χρήστη που διαθέτει η συγκεκριμένη εφαρμογή είναι αναγκαίο κάποιες φορές να αλληλεπιδρούν μεταξύ τους. Για να επιτευχθεί αυτό σημαίνει ότι θα πρέπει να αναπτυχθεί μια επικοινωνία ανάμεσα στις δραστηριότητες. Η επικοινωνία μεταξύ των δραστηριοτήτων πραγματοποιείται μέσω των **Intents**.

Intent είναι μια δομή δεδομένων η οποία μπορεί να μεταφέρεται μεταξύ διαφόρων αντικειμένων Android (Android objects) και η οποία διατηρεί μια αφηρημένη περιγραφή μιας λειτουργίας που πρόκειται να εκτελεστεί. Μια τέτοια λειτουργία είναι και η ανταλλαγή πληροφοριών μεταξύ δυο δραστηριοτήτων.

```
Intent intent = new Intent(Activity1.this, Activity2.class);
intent.putExtra(key, value);
startActivity(intent);
```

Με το σύνολο των εντολών που φαίνεται παραπάνω επιτυγχάνεται η δημιουργία μιας νέας δραστηριότητας μέσα απο την κύρια δραστηριότητα της εφαρμογής (MainActivity). Με αυτόν τον τρόπο δημιουργείται μια νέα διεπαφή χρήστη (οθόνη), η οποία αντικαθιστά την τρέχουσα χωρίς όμως να καταστρέφεται η παρούσα διεπαφή καθώς ο χρήστης μπορεί να επανέλθει οποιαδήποτε στιγμή πατώντας το κουμπί «Back» της συσκευής.

Η εντολή «intent.putExtra(key, value);» που υπάρχει στο παραπάνω σετ εντολών αφορά την προώθηση ενός μηνύματος στη νέα δραστηριότητα που δημιουργείται. Η παράμετρος «key» αποτελεί το αναγνωριστικό κλειδί του μηνύματος ενώ η παράμετρος «value» το περιεχόμενο του μηνύματος που αποστέλλεται. Για να λάβει η καινούρια δραστηριότητα το μήνυμα που στέλνεται χρησιμοποιεί το παρακάτω σετ εντολών.

```
// Getting all extras
Bundle extras = getIntent().getExtras();

// Getting your int (second param is the default value if null)
int value = extras.getIntExtra("Key1", 0);
```

Η εντολή «Bundle extras= getIntent().getExtras();» η οποία συνήθως εκτελείται στην εκκίνηση της δραστηριότητας, δηλαδη στη μέθοδο onCreate() αφορά την λήψη όλων των διαθέσιμων δεδομένων, ενώ η δεύτερη εντολή «int value = extras.getIntExtra("Key1", 0);» αφορά την ανάγνωση ενός συγκεκριμένου μηνύματος

(μέσω του εκάστοτε αναγνωριστικού κλειδιού), καθώς και την αποθήκευση της συγκεκριμένης πληροφορίας σε μια μεταβλητή.

5.4 Υπηρεσίες

Η εφαρμογή αποτελείται απο μία βασική υπηρεσία η οποία τρέχει στο παρασκήνιο και είναι υπεύθυνη για τη συλλογή και αποθήκευση των δεδομένων. Καλείται μέσα από την MainActivity με το πάτημα του κουμπιού “StartService”. Από τη στιγμή που καλείται η υπηρεσία εμφανίζεται μία μπάρα ειδοποίησης στην οθόνη της συσκευής η οποία ενημερώνει τον χρήστη ότι έχει ξεκινήσει η καταγραφή των δεδομένων, ενώ είναι ανεξάρτητη από την κύρια δραστηριότητα και συνεχίζει την καταγραφή δεδομένων ακόμα και αν σταματήσει να λειτουργεί η βασική εφαρμογή. Η διαδικασία αυτή διαρκεί έως ότου περάσει το απαραίτητο χρονικό διάστημα.

Η συγκεκριμένη υπηρεσία είναι μία δεσμευμένη υπηρεσία (bound service). Μια υπηρεσία είναι δεσμευμένη όταν μία συνιστώσα μιας εφαρμογής δεσμεύεται σε αυτήν καλώντας τη μέθοδο bindService(). Επίσης, μια δεσμευμένη υπηρεσία προσφέρει μία διεπαφή πελάτη-διακομιστή που επιτρέπει στα εξαρτήματα να αλληλεπιδρούν με την υπηρεσία, να στέλνουν αιτήματα, να λαμβάνουν αποτελέσματα και ακόμη να το κάνουν αυτό σε όλες τις διεργασίες με επικοινωνία μεταξύ διεργασιών. Τέλος, μία δεσμευμένη υπηρεσία εκτελείται μόνο για όσο χρονικό διάστημα δεσμεύεται ένα άλλο στοιχείο της εφαρμογής. Οι πολλαπλές συνιστώσες μπορούν να συνδεθούν με την υπηρεσία ταυτόχρονα, αλλά όταν όλες αυτές αποσυνδεθούν η υπηρεσία καταστρέφεται.

```
    private ServiceConnection mConnection = new ServiceConnection() {

        @Override
        public void onServiceConnected(ComponentName className,IBinder service) {
            // We've bound to LocalService, cast the IBinder and get LocalService instance
            MonitorService.LocalBinder binder = (MonitorService.LocalBinder) service;
            myService = binder.getService();
            mBound = true;
            myService.flag=true;

        }

        @Override
        public void onServiceDisconnected(ComponentName arg0) {
            mBound = false;
            myService.flag=false;
        }
    };
}
```

Ο λόγος που η υπηρεσία είναι ανεξάρτητη απο την κύρια εφαρμογή και μπορεί να προχωρήσει στην καταγραφή των δεδομένων αλλα και στην αποθήκευση τους την κατάλληλη στιγμή ειναι το wakelock. Το wakelock είναι ένας μηχανισμός ο οποίος

μπορεί να κρατήσει τον επεξεργαστή ενεργό έτσι ώστε να επιτελείται η λειτουργία της υπηρεσίας και να μην δέχεται τερματισμό από τον επεξεργαστή εξαιτίας της συνεχόμενης χρήσης των πόρων. Το wakelock αποκτάται τη στιγμή που δημιουργείται η υπηρεσία δηλαδή με το πάτημα του κουμπιού 'start service' από την κύρια οθόνη και ελευθερώνεται τη στιγμή που τελειώνει το χρονικό διάστημα στο οποίο γίνεται η συλλογή δεδομένων. Αυτό επιτυγχάνεται με τις εντολές wakeLock.acquire() και wakeLock.release(). Για να οριστεί ενα wakelock πρέπει αρχικά να δημιουργηθεί ενα στιγμιότυπο της κλάσης PowerManager έτσι ώστε να αποκτηθεί ο έλεγχος της κατάστασης ισχύος της συσκευής, το οποίο επιτυγχάνεται με τις ακόλουθες εντολές:

```
PowerManager powerManager = (PowerManager) getSystemService(POWER_SERVICE);
wakeLock =
powerManager.newWakeLock(PowerManager.PARTIAL_WAKE_LOCK,"MyApp::MyWakelockTag")
;
```

Προκειμένου να οριστεί το ακριβές χρονικό διάστημα για το οποίο η υπηρεσία επιτελεί τις λειτουργίες της, χρησιμοποιήθηκε η κλάση CountDownTimer. Μέσω αυτής της κλάσης, η οποία παίρνει σαν παραμέτρους δυο ακέραιους αριθμούς που δείχνουν τα χιλιοστά του δευτερολέπτου μπορεί να οριστεί μία αντίστροφη μέτρηση μέχρι μία συγκεκριμένη ώρα στο μέλλον καθώς και ο χτύπος του ρολογιού, δηλαδή το κάθε πότε στο συνολικό αυτό χρόνο θα πραγματοποιείται μια δράση. Αυτό γίνεται εφικτό μέσω της μεθόδου onTick(). Κάθε χτύπος του ρολογιού είναι συγχρονισμένος έτσι ώστε να εξασφαλίζεται ότι η κλήση της μεθόδου onTick() δε θα συμβεί αν δεν έχει ολοκληρωθεί πλήρως η αμέσως προηγούμενη. Μόλις ολοκληρωθεί η αντίστροφη μέτρηση καλείται η μέθοδος onFinish() όπου η υπηρεσία αφήνει ελεύθερο το wakelock, κλείνει το αρχείο καταγραφής των δεδομένων και τερματίζει τη λειτουργία της.

5.4.1 Επικοινωνία Υπηρεσίας με τις Δραστηριότητες

Η υπηρεσία επιτυγχάνει την επικοινωνία με τις υπόλοιπες δραστηριότητες μέσω των δεκτών εκπομπής. Οι δέκτες εκπομπής απλά ανταποκρίνονται σε μηνύματα εκπομπής από άλλες εφαρμογές ή από το ίδιο το σύστημα. Αυτά τα μηνύματα συνήθως ονομάζονται γεγονότα ή προθέσεις. Για παράδειγμα, οι εφαρμογές μπορούν επίσης να ξεκινήσουν εκπομπές για να επιτρέψουν σε άλλες εφαρμογές να γνωρίζουν ότι κάποια δεδομένα έχουν ληφθεί στη συσκευή και είναι διαθέσιμα για χρήση από αυτούς, οπότε πρόκειται για δέκτη εκπομπής που θα παρακολουθήσει αυτή την επικοινωνία και θα ξεκινήσει κατάλληλες ενέργειες.

Ένας δέκτης εκπομπής εφαρμόζεται ως υποκατηγορία της κλάσης BroadcastReceiver και παρακάμπτει τη μέθοδο onReceive(), όπου κάθε μήνυμα λαμβάνεται ως παράμετρος ενός intent. Με την εγγραφή ενός δέκτη εκπομπής στο αρχείο AndroidManifest.xml μια εφαρμογή είναι ικανή να αναμένει τη λήψη συγκεκριμένων μηνυμάτων.

Κατά τη διάρκεια της αντίστροφης μέτρησης που ορίστηκε παραπάνω, η υπηρεσία στέλνει σε κάθε χτύπο του ρολογιού όλα τα δεδομένα που συλλέγει ομαδοποιώντας τα με βάση κάποιο αναγνωριστικό. Το αναγνωριστικό αυτό διαφέρει σε κάθε δραστηριότητα έτσι ώστε να λάβουν τα κατάλληλα μηνύματα στη μέθοδο onReceive() και να τα εμφανίσουν τελικά στην οθόνη.

5.5 Συλλογή Δεδομένων

Η υπηρεσία που αναφέρθηκε παραπάνω καταγράφει δεδομένα για συνολικό χρονικό διάστημα 7 ημερών και με ρυθμό καταγραφής 1 λεπτο. Τα δεδομένα αυτά είναι τα εξής:

- Ημερομηνία
- Ώρα
- Στιγμή της ημέρας
- Χρήση του επεξεργαστή
- Φωτεινότητα
- Μπαταρία
- Κατάσταση φόρτισης
- Κατάσταση οθόνης
- Μνήμη
- Κατάσταση κλήσεων
- Υπηρεσίες Τοποθεσίας - GPS
- Σύνδεση στο Δίκτυο
- Κίνηση Συσκευής
- Ποσότητα Φωτός

5.5.1 Ημερομηνία

Αποτελεί την ημέρα καταγραφής.

```
public String getDate() {
   Calendar c = Calendar.getInstance();
   DecimalFormat df = new DecimalFormat("00");
   return new StringBuilder()
         .append(df.format(c.get(Calendar.YEAR))).append("-")
         .append(df.format(c.get(Calendar.MONTH) + 1)).append("-")
         .append(df.format(c.get(Calendar.DATE))).toString();

}
```

5.5.2 Ώρα

Είναι η δεδομένη χρονική στιγμή κατα την οποία γίνεται η καταγραφή.

```
public String getTime() {
   Calendar c = Calendar.getInstance();
   DecimalFormat df = new DecimalFormat("00");
   if(c.get(Calendar.HOUR_OF_DAY)>=5 && c.get(Calendar.HOUR_OF_DAY)<12)
      timeofday="Morning";
   else if(c.get(Calendar.HOUR_OF_DAY)>=12 && c.get(Calendar.HOUR_OF_DAY)<17)
      timeofday="Afternoon";
   else if(c.get(Calendar.HOUR_OF_DAY)>=17 && c.get(Calendar.HOUR_OF_DAY)<21)
      timeofday="Evening";
   else
      timeofday="Night";
   return new StringBuilder()
         .append(df.format(c.get(Calendar.HOUR_OF_DAY))).append(":")
         .append(df.format(c.get(Calendar.MINUTE))).append(":")
         .append(df.format(c.get(Calendar.SECOND))).toString();
}
```

5.5.3 Στιγμή της ημέρας

Αποτελεί τη στιγμή της ημέρας ανάλογα με την ώρα και χωρίζεται σε τέσσερις διαφορετικές καταστάσεις: Πρωί, Μεσημέρι, Απόγευμα, Βράδυ.

5.5.4 Χρήση του επεξεργαστή

Αποτελεί το ποσοστό χρήσης του επεξεργαστή με το βασικό περιορισμό οτι αυτή η μέτρηση αφορά συσκευές Android που χρησιμοποιούν εκδόσεις μικρότερες της 7^{ης} (Nougat).

```
private float readUsage() {
   try {
      RandomAccessFile reader = new RandomAccessFile("/proc/stat", "r");
      String load = reader.readLine();

      String[] toks = load.split(" +");  // Split on one or more spaces

      long idle1 = Long.parseLong(toks[4]);
      long cpu1 = Long.parseLong(toks[2]) + Long.parseLong(toks[3]) + Long.parseLong(toks[5])
            + Long.parseLong(toks[6]) + Long.parseLong(toks[7]) + Long.parseLong(toks[8]);

      try {
         Thread.sleep(360);
      } catch (Exception e) {}

      reader.seek(0);
      load = reader.readLine();
      reader.close();

      toks = load.split(" +");

      long idle2 = Long.parseLong(toks[4]);
      long cpu2 = Long.parseLong(toks[2]) + Long.parseLong(toks[3]) + Long.parseLong(toks[5])
            + Long.parseLong(toks[6]) + Long.parseLong(toks[7]) + Long.parseLong(toks[8]);
```

```
            return (float)(cpu2 - cpu1)*100 / ((cpu2 + idle2) - (cpu1 + idle1));

        } catch (IOException ex) {
            ex.printStackTrace();
        }

        return 0;
    }
```

5.5.5 Φωτεινότητα

Αποτελεί το επίπεδο φωτεινότητας της οθόνης της συσκευής σε ποσοστιαία κλίμακα.

```
public int getBrightness() {
    try {
        Bright = Settings.System.getInt(getContentResolver(), Settings.System.SCREEN_BRIGHTNESS);
        bri = (Bright * 100.0f) / 255;
        //Log.d(TAG,"Brightness"+Integer.toString(Bright));
    } catch (Settings.SettingNotFoundException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
    return Math.round(bri);

}
```

5.5.6 Μπαταρία

Πρόκειται για το επίπεδο της μπαταρίας του κινητού σε ποσοστιαία κλίμακα.

```
public float getBatteryLevel() {
    Intent batteryIntent = registerReceiver(null, new
IntentFilter(Intent.ACTION_BATTERY_CHANGED));
    int level = batteryIntent.getIntExtra(BatteryManager.EXTRA_LEVEL, -1);
    int scale = batteryIntent.getIntExtra(BatteryManager.EXTRA_SCALE, -1);

    // Error checking that probably isn't needed but I added just in case.
    if (level == -1 || scale == -1) {
        return 50.0f;
    }

    return ((float) level / (float) scale) * 100.0f;
}
```

5.5.7 Κατάσταση φόρτισης

Το χαρακτηριστικό αυτό δείχνει αν η συσκευή βρίσκεται σε κατάσταση φόρτισης ή όχι.

```
public String getChargingState() {

    IntentFilter filter = new IntentFilter(Intent.ACTION_BATTERY_CHANGED);
```

```
    Intent batteryStatus = registerReceiver(null, filter);

    int chargeState = batteryStatus.getIntExtra(BatteryManager.EXTRA_STATUS, -1);
    String strState;

    switch (chargeState) {
        case BatteryManager.BATTERY_STATUS_CHARGING:
        case BatteryManager.BATTERY_STATUS_FULL:
            strState = "Charging";
            break;
        default:
            strState = "Discharging";
    }
    return strState;

}
```

5.5.8 Κατάσταση οθόνης

Πρόκειται για την κατάσταση της οθόνης δηλαδή αν είναι ενεργή ή ανενεργη.

```
public String getScreenStatus() {
    PowerManager pm = (PowerManager) getSystemService(Context.POWER_SERVICE);
    String screenonff;
    if (pm.isScreenOn()) {
        screenonff = "Screen on";

    }else{
        screenonff= "Screen off";
    }
    return screenonff;
}
```

5.5.9 Μνήμη

Το χαρακτηριστικό αυτό αποτελεί τη διαθέσιμη προσωρινή μνήμη (RAM) της συσκευής.

```
public String getMemory() {

    ActivityManager.MemoryInfo mi = new ActivityManager.MemoryInfo();
    ActivityManager activityManager = (ActivityManager) getSystemService(ACTIVITY_SERVICE);
    activityManager.getMemoryInfo(mi);
    double availableMegs = mi.availMem / 0x100000L;
    double usedMegs = mi.totalMem / 0x100000L - availableMegs;
    //Percentage can be calculated for API 16+
    double percentAvail = Math.round(mi.availMem / (double) mi.totalMem * 100.0);
    double percentUsed = Math.round((double) (mi.totalMem - mi.availMem) / (double) mi.totalMem *
100.0);
    return String.valueOf(percentUsed);
}
```

5.5.10 Κατάσταση κλήσεων

Το χαρακτηριστικό αυτό μας δείχνει αν η συσκευή δέχεται κάποια κλήση τη δεδομένη χρονική στιγμή, αν ο κάτοχος της συσκευής βρίσκεται σε κάποια κλήση ή αν δε συμβαίνει τίποτα απο τα δυο προαναφερθέντα συμβάντα.

```
PhoneStateListener callStateListener = new PhoneStateListener() {
   public void onCallStateChanged(int state, String incomingNumber)
   {
      if(state==TelephonyManager.CALL_STATE_RINGING){
         callState="Phone Is Riging";
      }
      if(state==TelephonyManager.CALL_STATE_OFFHOOK){
         callState="Phone is Currently in A call";
      }

      if(state==TelephonyManager.CALL_STATE_IDLE){
         callState="Phone is neither ringing nor in a call";
      }
   }
};
```

5.5.11 Υπηρεσίες Τοποθεσίας - GPS

Πρόκειται ουσιαστικά για την υπηρεσία τοποθεσίας. Πέρα από το γεωγραφικό μήκος και πλάτος το χαρακτηρικστικό αυτό μας ενημερώνει αν η συσκευή έχει ενεργοποιημένη την υπηρεσία τοποθεσίας. Σε περίπτωση που είναι ανενεργή αποθηκεύεται η τελευταία καταγεγραμμένη τοποθεσία.

```
private class LocationListener implements android.location.LocationListener
{
   Location mLastLocation;

   public LocationListener(String provider)
   {
      Log.e(TAG, "LocationListener " + provider);
      mLastLocation = new Location(provider);
   }

   @Override
   public void onLocationChanged(Location location)
   {
      Log.e(TAG, "onLocationChanged: " + location);
      mLastLocation.set(location);
      longitude=mLastLocation.getLongitude();
      latitude=mLastLocation.getLatitude();
   }

   @Override
   public void onProviderDisabled(String provider)
   {
      Log.e(TAG, "onProviderDisabled: " + provider);
      GPS=0;
```

```
    }

    @Override
    public void onProviderEnabled(String provider)
    {
        Log.e(TAG, "onProviderEnabled: " + provider);
        GPS=1;
    }

    @Override
    public void onStatusChanged(String provider, int status, Bundle extras)
    {
        Log.e(TAG, "onStatusChanged: " + provider);
    }
}
```

5.5.12 Σύνδεση στο Δίκτυο

Πρόκειται για την κατάσταση της σύνδεσης της συσκευής στο δίκτυο, δηλαδή αν υπάρχει κάποια σύνδεση ή οχι, τι είδους σύνδεση είναι καθώς και την ταχύτητα της.

```
if (serviceConnectivity.isConnected(context)) {
    connected = true;
    connect="Connected";

} else {
    connected=false;
    connect="Not Connected";
}
if (connected) {
    if (serviceConnectivity.isConnectedWifi(context)){
        type="Wifi";
        MobileData=0;
    }
    else{
        type="Mobile Data";
        MobileData=1;

    }
    if (serviceConnectivity.isConnectedFast(context)){
        speed="Fast Connection";
    }
    else {
        speed="Slow Connection";
    }
} else {
    type="There is no Internet Connection";
    speed="There is no Internet Connection";
}
```

5.5.13 Κίνηση Συσκευής

Αποτελεί τη θέση της συσκευής, όσον αφορά το αν ο χρήστης βρίσκεται σε κίνηση ή η συσκευή είναι σταθερή σε κάποιο σημείο.

```
@Override
public void onSensorChanged(SensorEvent event) {
   if (event.sensor.getType() == Sensor.TYPE_ACCELEROMETER){
      mGravity = event.values.clone();
      // Shake detection
      float x = mGravity[0];
      float y = mGravity[1];
      float z = mGravity[2];
      mAccelLast = mAccelCurrent;
      mAccelCurrent = (float)Math.sqrt(x*x + y*y + z*z);
      float delta = mAccelCurrent - mAccelLast;
      mAccel = mAccel * 0.9f + delta;
      // Make this higher or lower according to how much
      // motion you want to detect
      if(mAccel > 0.01){
         // do something
         device="Device Moving";
      }
      else{
         device="Device Not Moving";
      }

   }
   if(event.sensor.getType() == Sensor.TYPE_LIGHT){
         mLightQuantity = event.values[0];


   }


}
```

5.5.14 Ποσότητα Φωτός

Πρόκειται για τη φωτεινότητα που αντιλαμβάνεται ο αισθητήρας φωτός που βρίσκεται στο μπροστινό μέρος της συσκευής και ουσιαστικά παρέχει πληροφορίες για το φως που υπάρχει στο χώρο που βρίσκεται η συσκευή.

```
@Override
public void onSensorChanged(SensorEvent event) {

   if(event.sensor.getType() == Sensor.TYPE_LIGHT){
         mLightQuantity = event.values[0];


   }
}
```

5.6 Αποθήκευση Δεδομένων

Τα δεδομένα που αναφέρθηκαν παραπάνω αποθηκεύονται σε αρχείο της μορφής CSV όπου κάθε πεδίο κειμένου από αυτά που συλλέγονται διαχωρίζεται συνήθως με κόμμα (,).

```
String entry =  getDate() + "," + getTime() + "," + String.valueOf(readUsage())  + "," + timeofday + ","
      + Integer.toString(getBrightness()) + "," + String.valueOf(getBatteryLevel()) + "," +
getChargingState() + ","
      + getScreenStatus() + "," + callState + "," + getMemory()+ ","  +  String.valueOf(longitude) + ","
      + String.valueOf(latitude)  + "," +  connect + "," + type + "," + speed + "," + device + "," +
      String.valueOf(mLightQuantity) + "," + String.valueOf(GPS) + "," + String.valueOf(MobileData);



try
{

   fos.write(entry.getBytes());
   String lineSeparator = System.getProperty("line.separator");
   fos.write(lineSeparator.getBytes());
   //fos.close();
}
catch (FileNotFoundException e) {
   e.printStackTrace();
}
catch (IOException e) {
   e.printStackTrace();
}
```

Κεφάλαιο 6

Πειραματική Μελέτη

6.1 Πρόβλεψη υπηρεσιών με χρήση αλγορίθμων μηχανικής μάθησης

Στην ενότητα αυτή θα γίνει χρήση των αλγορίθμων μηχανικής μάθησης, όπως αυτοί περιγράφηκαν στο κεφάλαιο 4 με χρήση της γλώσσας προγραμματισμού Python για την πρόβλεψη των απαιτήσεων του χρήστη για δεδομένα κινητής τηλεφωνίας καθώς και υπηρεσιών τοποθεσίας. Για την αξιολόγηση του συγκεκριμένου μοντέλου χρησιμοποιήθηκε ως μετρική η ακρίβεια (accuracy) η οποία ορίζεται σύμφωνα με τον ακόλουθο τύπο [46] :

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

όπου TN είναι οι σωστές αρνητικές προβλέψεις (True Negative), TP οι σωστές θετικές προβλέψεις (True Positive), FP οι λανθασμένες θετικές προβλέψεις (False Positive) και FN οι λανθασμένες αρνητικές προβλέψεις (False Negative).

Τα δεδομένα που συλλέχθηκαν μέσω της εφαρμογής σε διάστημα 7 ημέρων και καταγράφοντάς τα κάθε 1 λεπτό θα χρησιμοποιηθούν για να γίνει μια κατηγοριοποίηση κατάστασης (state classification) του κάθε χρήστη. Τα δεδομένα αυτά προέρχονται από μία συσκευή. Πιο συγκεκριμένα έχοντας ως εξαρτημένες μεταβλητές τα δεδομένα κινητής τηλεφωνίας και τις υπηρεσίες τοποθεσίας προκύπτουν 4 καταστάσεις από τον κάθε πιθανό συνδυασμό των συγκεκριμένων μεταβλητών σύμφωνα με τον παρακάτω πίνακα.

State	Data Required	Location Required
1	**No**	**No**
2	**No**	**Yes**
3	**Yes**	**No**
4	**Yes**	**Yes**

Πίνακας 2: Συνδυασμός καταστάσεων χρήστη.

Κάθε κατάσταση δείχνει τις απαιτήσεις της συσκευής σε πόρους κάθε χρονική στιγμή. Γνωρίζοντας τις απαιτήσεις αυτές κάθε χρονική στιγμή μπορεί να επιτευχθεί τελικά η καλύτερη διαχείριση των πορών, το οποίο θα έχει ως αποτελεσμά τελικά την εξοικονόμηση της ενέργειας.

6.2 Το σύνολο δεδομένων (dataset)

Για την εφαρμογή των αλγορίθμων μηχανικής μάθησης χρειάστηκε να δημιουργήσουμε ένα σύνολο δεδομένων, η δημιουργία του οποίου έγινε μέσω της εφαρμογής που περιγράφηκε σε προηγούμενο κεφάλαιο. Στην ουσία δημιουργήθηκε ένα dataframe με 10080 γραμμές όσα και τα λεπτά που αντιστοιχούν σε 7 μέρες καταγραφής δεδομένων, με καθεμία να πραγματοποιείται κάθε ένα λεπτό. Επίσης δημιουργήθηκαν 18 στήλες όσα και τα χαρακτηριστικά που συλλέχθηκαν:

- Στήλη 1: αφορά ημερομηνία καταγραφής (Date)
- Στήλη 2: αφορά την ώρα καταγραφης (Time)
- Στήλη 3: αφορά τη χρήση του επεξεργαστή (Μη διαθέσιμο για εκδόσεις Android μεγαλύτερες απο 7.0) (CPU Usage)
- Στήλη 4: αφορά τη στιγμή της ημέρας (Time of Day)
- Στήλη 5: αφορά το επίπεδο φωτεινότητας (Brightness)
- Στήλη 6: αφορά το ποσοστό της μπαταρίας που απομένει (Battery Level)
- Στήλη 7: αφορά την κατάσταση φόρτισης (Charging State)
- Στήλη 8: αφορά την κατάσταση της οθόνης (Screen State)
- Στήλη 9: αφορά την κατάσταση της συσκευής όσον αφορά τις εισερχόμενες-εξερχόμενες κλήσεις (Calling State)
- Στήλη 10: αφορά το γεωγραφικό μηκος (Longitute)
- Στήλη 11: αφορά το γεωγραφικό πλάτος (Latitude)
- Στήλη 12: αφορά τη σύνδεση με το δίκτυο (Network Connection)
- Στήλη 13: αφορά τον τύπο της σύνδεσης (Connection Type)
- Στήλη 14: αφορά την ταχύτητα της σύνδεσης (Connection Speed)
- Στήλη 15: αφορά την κίνηση της συσκευής (Device Movement)
- Στήλη 16: αφορά το φως που αντιλαμβάνεται ο αισθητήρας φωτός της συσκευής (Ambient Light)
- Στήλη 17: αφορά την κατάσταση των υπηρεσιών τοποθεσίας της συσκευής (GPS Status)
- Στήλη 18: αφορά την κατάσταστη των δεδομένων κινητής τηλεφωνίας της συσκευής (Mobile Data Status)

Για την εφαρμογή των αλγορίθμων μηχανικής μάθησης όπως αυτοί που παρουσιάστηκαν στο Κεφάλαιο 4, χρειάστηκε να χωρίσουμε το σύνολο δεδομένων που δημιουργήσαμε σε δεδομένα εκπαίδευσης (train data) και δεδομένα δοκιμής (test

data). Λόγω του μεγάλου πλήθους των δεδομένων εφαρμόστηκε η μέθοδος k-fold cross validation, όπου στην προκειμένη περίπτωση k=10. Με τη συγκεκριμένη μέθοδο το σύνολο δεδομένων χωρίζεται σε 10 τμήματα, κάθε ένα απο τα οποία κάποια στιγμή θα αποτελέσει το σύνολο πάνω στο οποίο θα δοκιμαστεί το μοντέλο πρόβλεψης, ενώ τα υπόλοιπα αποτελούν τα σύνολα που εκπαιδεύουν το μοντέλο. Η διαδικασία αυτή επαναλαμβάνεται έως ότου όλα τα τμήματα να έχουν χρησιμοποιηθεί σαν σύνολα ελέγχου δεδομένων.

```
from sklearn.model_selection import KFold, cross_val_score
kf = KFold(n_splits=10)
kf

def get_score (model, X_train, X_test, y_train, y_test):
    model.fit(X_train,y_train)
    return model.score(X_test,y_test)
```

Ο κάθε αλγόριθμος στη συνέχεια εκπαιδεύτηκε στα δεδομένα εκμάθησης (train data) και εκτιμήθηκε στα δεδομένα ελέγχου (test data) με σκοπό την πρόβλεψη των δυο μεταβλητών στόχου. Ο συνδυασμός των δυο αυτών προβλέψων μας δίνει και την κατάσταση στην οποία βρισκέται ο χρήστης.

Παρακάτω παρουσιάζεται το dataframe για τα πρώτα λεπτά στα οποία τρέχει η εφαρμογή:

Date	Time	Cpu Usage	Time of Day	Brightness	Battery Level	Charging State	Screen State	Longitude	Cpu Usage	Longitude	Latitude	Connection status	Connection Type	Connection Speed	Device Status	Ambient Light
13/11/2019	9:30:36	0.0	Morning	38	93.0	Discharging	Screen off	Phone is neither ringing nor in a call	59.0	237.684.302	3.798.148.353	Connected	Mobile Data	Slow Connection	Device Not Moving	4.0
13/11/2019	9:31:36	0.0	Morning	38	93.0	Discharging	Screen off	Phone is neither ringing nor in a call	58.0	2.376.719.857	3.798.267.511	Connected	Mobile Data	Slow Connection	Device Not Moving	4.0
13/11/2019	9:32:36	0.0	Morning	23	93.0	Discharging	Screen on	Phone is neither ringing nor in a call	60.0	2.376.825.907	3.798.327.819	Connected	Mobile Data	Slow Connection	Device Moving	44.0
13/11/2019	9:33:36	0.0	Morning	33	93.0	Discharging	Screen on	Phone is neither ringing nor in a call	63.0	2.376.841.958	3.798.336.541	Connected	Mobile Data	Slow Connection	Device Moving	99.0
13/11/2019	9:34:36	0.0	Morning	60	92.0	Discharging	Screen off	Phone is neither ringing nor in a call	62.0	237.675.083	379.853.632	Connected	Mobile Data	Slow Connection	Device Moving	4.0
13/11/2019	9:35:36	0.0	Morning	60	92.0	Discharging	Screen off	Phone is neither ringing nor in a call	62.0	2.376.842.032	3.798.763.372	Connected	Mobile Data	Slow Connection	Device Not Moving	4.0
13/11/2019	9:36:36	0.0	Morning	47	92.0	Discharging	Screen on	Phone is neither ringing nor in a call	63.0	2.376.926.094	3.798.937.846	Connected	Mobile Data	Slow Connection	Device Moving	73.0
13/11/2019	9:37:36	0.0	Morning	41	92.0	Discharging	Screen off	Phone is neither ringing nor in a call	63.0	2.377.668.769	3.799.373.312	Connected	Mobile Data	Slow Connection	Device Moving	18.0
13/11/2019	9:38:36	0.0	Morning	46	92.0	Discharging	Screen on	Phone is neither ringing nor in a call	64.0	237.787.649	379.954.026	Connected	Mobile Data	Slow Connection	Device Not Moving	60.0
13/11/2019	9:39:36	0.0	Morning	46	92.0	Discharging	Screen on	Phone is neither ringing nor in a call	67.0	2.378.419.415	3.799.889.482	Connected	Mobile Data	Slow Connection	Device Not Moving	174.0
13/11/2019	9:40:36	0.0	Morning	49	92.0	Discharging	Screen on	Phone is neither ringing nor in a call	68.0	237.860.792	3.800.024.193	Connected	Mobile Data	Slow Connection	Device Not Moving	123.0

Εικόνα 14: Στιγμιότυπο των δεδομένων.

6.3 Εξαγωγή Χαρακτηριστικών

Από το παραπάνω σύνολο δεδομένων που δημιουργήθηκε χρειάστηκε να επιλεχθούν εκείνα τα χαρακτηριστικά τα οποία θα αποτελέσουν την καλύτερη δυνατή είσοδο για το σύνολο των αλγορίθμων που χρησιμοποιήθηκαν έτσι ώστε η πρόβλεψη να είναι όσο το δυνατόν πιο ακριβής. Τα χαρακτηριστικά αυτά θα αποτελέσουν τις ανεξάρτητες μεταβλητές για την εκτίμηση – πρόβλεψη της εξαρτημένης τιμής που στην προκειμένη περίπτωση οι τιμές αυτές είναι η χρήση ή μη των υπηρεσιών δεδομένων τηλεφωνίας καθώς και τοποθεσίας. Για την επιλογή των συγκεκριμένων χαρακτηριστικών χρειάστηκε να γίνει αναζήτηση στο σύνολο των δεδομένων και στη συνέχεια να επιλεχθούν τα χαρακτηριστικά που δίνουν ακριβέστερη πρόβλεψη για όλους τους διαθέσιμους αλγορίθμους.

Οι ανεξάρτητες αυτές μετάβλητες είναι οι εξής:

- Η στιγμή της ημέρας (Time of Day), η οποία έχει διαθέσιμες 4 τιμές: Morning, Afternoon, Evening, Night
- Η κατάσταση της οθόνης (Screen State), η οποία παίρνει δύο τιμες: Ανοιχτή, Κλειστή
- Το γεωγραφικό μήκος (Longitude), το οποίο παίρνει τιμές στο σύνολο των πραγματικών αριθμών
- Το γεωγραφικό πλάτος (Latitude), το οποίο αποτελεί επίσης έναν πραγματικό αριθμό
- Η κατάσταση της σύνδεσης στο διαδίκτυο (Connection State), η οποία έχει δύο διαθέσιμες τιμές: Ναι, Όχι
- Η κίνηση της συσκευής (Device Movement), η οποία δηλώνει το αν κινείται η συσκευή ή βρίσκεται σταθερή (για παράδειγμα πάνω σε ένα έπιπλο) ανάλογα με τα δεδομένα που λαμβάνονται απο τους αισθητήρες της συσκευής καθώς και ορίζοντας ένα συντελεστή ευαισθησίας για το τι μπορεί να θεωρηθεί κίνηση και τι όχι. Η συγκεκριμένη μεταβλητή παίρνει επίσης τιμές Ναι ή Οχι.
- Η αντίληψη φωτός από τη συσκευή (Ambient Light). Πρόκειται για μία μεταβλητή η οποία λαμβάνει ακέραιες τιμές και δηλώνει μέσα από αυτό τον αριθμό, το πόσο φως αντιλαμβάνεται ο αισθητήρας που βρίσκεται στο μπροστινό μέρος της συσκευής. Όσο πιο κοντά στο μηδέν είναι ο αριθμός αυτός σημαίνει ότι τόσο πιο σκοτεινός είναι ο χώρος που βρίσκεται η συσκευή είτε αυτό είναι ένα δωμάτιο χωρίς φως είτε αυτό είναι η τσέπη του χρήστη, κάτι το οποίο μπορεί να φανεί πολύ χρήσιμο στη βελτίωση της ακρίβειας των προβλέψεων.

6.4 Κανονικοποίηση Χαρακτηριστικών

Εξαιτίας της διαφορετικότητας των χαρακτηριστικών ως προς τις τιμές που μπορούν να λάβουν εμφανίστηκε η ανάγκη για κανονικοποίηση ορισμένων. Πιο συγκεκριμένα, τα χαρακτηριστικά που χρειάστηκαν να μετατραπούν είναι η στιγμή της ημέρας, η κατάσταση της οθόνης, η κατάσταση της σύνδεσης στο διαδίκτυο καθώς και η κατάσταση κίνησης της συσκευής. Ο λόγος που χρειάζονται κανονικοποίηση τα συγκεκριμένα χαρακτηριστικά έχει να κάνει με τον τύπο των διαθέσιμων τιμών που μπορούν να λάβουν.

Προκειμένου να έχουν όλα τα χαρακτηριστικά σαν τύπο μεταβλητής αριθμό θα μετατρέψουμε την κάθε μεταβλητή σε Ν αριθμό μεταβλητών, όπου Ν θα είναι το πλήθος των διαφορετικών τιμών που μπορεί να πάρει η συγκεριμένη μεταβλητή. Για παράδειγμα το χαρακτηριστικό «Στιγμή της ημέρας» έχει τέσσερις διαφορετικές τιμές που μπορεί να πάρει (Morning, Afternoon, Evening, Night). Αυτό σημαίνει οτι το Ν στη συγκεριμένη περίπτωση ισούται με Ν=4. Επομένως αν η μεταβλητή «Στιγμή της

ημέρας» έχει τιμή «Evening» τότε το αποτέλεσμα μετά την κανονικοποίηση θα είναι να έχουμε 4 διαφορετικές μεταβλητές όπως στον ακόλουθο πίνακα:

Feature	Morning	Afternoon	Evening	Night
Στιγμή της ημέρας	0	0	1	0

Πίνακας 3: Παράδειγμα κανονικοποιημένων δεδομένων

Με αντίστοιχο τρόπο θα προκύψει παρόμοιο αποτέλεσμα και για τα υπόλοιπα χαρακτηριστικά. Ο κώδικας με τον οποίο επιτυγχάνεται αυτό φαίνεται παρακάτω:

```
# Encoding categorical data
# Encoding every Independent Variable
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
from sklearn.compose import ColumnTransformer
labelencoder_X = LabelEncoder()
X[:, 0] = labelencoder_X.fit_transform(X[:,0])
X[:, 1] = labelencoder_X.fit_transform(X[:,1])
X[:, 2] = labelencoder_X.fit_transform(X[:,2])
X[:, 3] = labelencoder_X.fit_transform(X[:,3])
transformer = ColumnTransformer(
    transformers=[
        ("OneHot",        # Just a name
         OneHotEncoder(), # The transformer class
         [0,1,2,3]              # The column(s) to be applied on.
         )
    ]
)
X=transformer.fit_transform(X)
```

6.5 Εφαρμογή αλγορίθμων μηχανικής μάθησης και εξαγωγή προβλέψεων

Έπειτα από τις διαδικασίες που ακολουθήθηκαν παραπάνω, δηλαδή της εξαγωγής χαρακτηριστικών από το σύνολο των δεδομένων αλλά και της κανονικοποίησης μέρους αυτών είναι εφικτός πλέον ο διαχωρισμός σε δεδομένα εκμάθησης και δεδομένα ελέγχου. Τα χαρακτηριστικά που αναλύθηκαν παραπάνω θα αποτελέσουν τις ανεξάρτητες μεταβλητές στην προκειμένη προσπάθεια εξαγωγής προβλέψεων, ενώ οι υπηρεσίες τοποθεσίας και οι υπηρεσίες δεδομένων κινητής τηλεφωνίας αποτελούν τις εξαρτημένες μεταβλητές.

Για τις ανάγκες της παρούσα διπλωματικής εργασίας τα δεδομένα χωρίστηκαν τυχαία σε 90% δεδομένα εκμάθησης (train set) και 10% (test set). Ο συγκεκριμένος διαχωρισμός μπορεί να φανεί με τον παρακάτω κώδικα:

```
# 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 = 0.1, random_state = 0)
```

Μετα τον διαχωρισμό των δεδομένων καθίσταται δυνατή η εφαρμογή των αλγορίθμων μηχανικής μάθησης για την εξαγωγή προβλέψεων. Οι αλγόριθμοι που εφαρμόστηκαν στη συγκεκριμένη εργασία όπως αναφέρθηκε και σε προηγούμενο κεφάλαιο είναι οι: Λογιστική Παλινδρόμηση, Κ-Πλησιέστεροι Γείτονες (KNN), Δέντρα απόφασης, Τυχάια δένδρα (Random Forests), Μηχανές Διανυσμάτων Υποστήριξης – SVMs. Οι αλγόριθμοι αυτοί εκπαιδεύτηκαν πάνω στα δεδομένα εκμάθησης και στη συνέχεια εκτιμήθηκαν πάνω στα δεδομένα ελέγχου. Παρακάτω φαίνεται ο κώδικας με τον οποίο εφαρμόστηκαν οι αλγόριθμοι, των οποίων η ανάλυση έχει πραγματοποιηθεί στο Κεφάλαιο 4, για την εξαγωγή των προβλέψεων.

```
# Fitting KNN to the Training set
# Create your classifier here
from sklearn.neighbors import KNeighborsClassifier
classifier = KNeighborsClassifier(n_neighbors=10,metric='minkowski',p=2)
classifier.fit(X_train,y_train)
prediction=classifier.predict(X_test)
from sklearn import metrics
#evaluation(Accuracy)
print("Accuracy:",metrics.accuracy_score(prediction,y_test))
#evaluation(Confusion Metrix)
print("Confusion Metrix:\n",metrics.confusion_matrix(prediction,y_test))
```

```
# Fitting Logistic Regression to the Training set
# Create your classifier here
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression(random_state=0)
classifier.fit(X_train,y_train)
prediction=classifier.predict(X_test)
from sklearn import metrics
#evaluation(Accuracy)
print("Accuracy:",metrics.accuracy_score(prediction,y_test))
#evaluation(Confusion Metrix)
print("Confusion Metrix:\n",metrics.confusion_matrix(prediction,y_test))
```

```
# Fitting SVM to the Training set
# Create your classifier here
from sklearn.svm import SVC
classifier = SVC(kernel='linear',random_state=0)
classifier.fit(X_train,y_train)
prediction=classifier.predict(X_test)
from sklearn import metrics
#evaluation(Accuracy)
print("Accuracy:",metrics.accuracy_score(prediction,y_test))
#evaluation(Confusion Metrix)
print("Confusion Metrix:\n",metrics.confusion_matrix(prediction,y_test))
```

```
# Fitting Decision Tree to the Training set
# Create your classifier here
from sklearn.tree import DecisionTreeClassifier
classifier = DecisionTreeClassifier(criterion='entropy',random_state=0)
classifier.fit(X_train,y_train)
prediction=classifier.predict(X_test)
from sklearn import metrics
#evaluation(Accuracy)
print("Accuracy:",metrics.accuracy_score(prediction,y_test))
#evaluation(Confusion Metrix)
print("Confusion Metrix:\n",metrics.confusion_matrix(prediction,y_test))
```

```
# Fitting RandomForest to the Training set
# Create your classifier here
from sklearn.ensemble import RandomForestClassifier
classifier = RandomForestClassifier(n_estimators=10,criterion='entropy',random_state=0)
classifier.fit(X_train,y_train)
prediction=classifier.predict(X_test)
from sklearn import metrics
#evaluation(Accuracy)
print("Accuracy:",metrics.accuracy_score(prediction,y_test))
#evaluation(Confusion Metrix)
print("Confusion Metrix:\n",metrics.confusion_matrix(prediction,y_test))
```

Αξίζει να σημειωθεί οτι οι παράμετροι που χρησιμοποιήσαν οι συγκεκριμένοι αλγόριθμοι μηχανικής μάθησης έχουν τις προεπιλεγμένες τιμές που έχει θέσει το προγραμματιστικό περιβάλλον Spyder.

Τα αποτελέσματα των προβλέψεων μπορούν να φανούν στους δυο παρακάτω πίνακες για κάθε μια απο τις εξαρτημένες μεταβλητές, δηλαδή για τα δεδομένα κινητής τηλεφωνίας και για τις υπηρεσίες τοποθεσίας.

Τα ποσοστά αυτά αποτελούν το μέσο όρο ακρίβειας των δέκα προβλέψεων που προέκυψαν απο τη διαδικασία 10-fold Cross Validation που ακολουθήθηκε.

Mobile Data:

Algorithm	Logistic Regression	KNN	SVM	Decision Tree	Random Forest
Preciction Accuracy	82.1%	77.8%	81.3%	82.5%	82.6%

GPS:

Algorithm	Logistic Regression	KNN	SVM	Decision Tree	Random Forest
Preciction Accuracy	97.5%	97.3%	95.9%	97.16%	96.74%

Απο τα παραπάνω αποτελέσματα γίνεται αντιληπτό ότι όσον αφορά την εξαρτημένη μεταβλητή «Δεδομένα κινητής τηλεφωνίας», καλύτερη μέση ακρίβεια πρόβλεψης δίνει ο αλγόριθμος 'Τυχαίο Δάσος', μόλις 0.1% καλύτερη απο τα 'Δέντρα Απόφασης'.

Για τη μεταβλητή «Υπηρεσίες Τοποθεσίας», είναι εμφανές ότι όλοι οι αλγόριθμοι δίνουν πολύ καλά αποτελέσματα, με τον καλύτερο από αυτούς να είναι ο αλγόριμος Κ-Πλησιέστερων Γειτόνων.

Κεφάλαιο 7

Επίλογος

Στο συγκεκριμένο κεφάλαιο, το οποίο αποτελεί και το τελευταίο της διπλωματικής εργασίας γίνεται μία ανασκόπηση της συνολικής δουλειάς που έχει γίνει κατά την ανάπτυξη της εφαρμογής για Android λειτουργικό, καθώς επίσης και στην αξιοποίηση της συγκεκριμένης εφαρμογής μέσα από τα δεδομένα που αντλώνται για την εξαγωγή συμπερασμάτων για τον εκάστοτε χρήστη. Παράλληλα αναφέρονται και κάποιες μελλοντικές δυνατότητες που προκύπτουν από τη συγκεκριμένη εργασία στην όσο το δυνατόν καλύτερη αξιοποιηση μιας συσκευής Android, έτσι ώστε να βελτιωθεί ακόμα περισσότερο η εμπειρία χρήσης.

7.1 Σύνοψη και Συμπεράσματα

Είναι εμφανές ότι τα τελευταία χρόνια οι έξυπνες συσκευές αποτελούν αναπόσπαστο κομμάτι της καθημερινότητας του ανθρώπου. Μάλιστα η εξέλιξη της τεχνολογίας έχει οδηγήσει στην ανάπτυξη έξυπνων φορητών συσκευών ικανών να αντικαταστήσουν ακόμα και τον προσωπικό υπολογιστή, προσφέροντας πολλές περισσότερες δυνατότητες από αυτές ενός συμβατικού τηλεφώνου. Η συνεχόμενη τάση εξέλιξης των φορητών έξυπνων κινητών παράλληλα με τις αυξανόμενες υπηρεσίες που προσφέρουν, όπως είναι εύκολα κατανοητό, οδηγεί στην αυξημένη ανάγκη για μεγαλύτερη αυτονομία της συσκευής.

Το αντικείμενο της παρούσας διπλωματικής είναι η ανάπτυξη μιας εφαρμογής για Android συσκευές με σκοπό τη συλλογή δεδομένων χρήσης από τον κάτοχο της συσκευής. Στη συνέχεια τα δεδομένα αυτά επεξεργάζονται και τροφοδοτούν σύγχρονους αλγοριθμούς μηχανικής μάθησης με σκοπό να προκύψουν κάποιες προβλέψεις για τις απαιτήσεις του χρήστη σε δεδομένα κινητής τηλεφωνίας καθώς και υπηρεσίες τοποθεσίας. Απώτερος σκοπός αυτής της μελέτης είναι μέσα από τις προβλέψεις που πραγματοποιούνται να υπάρχει ένα μοντέλο που να μπορεί ανά πάσα στιγμή να προβλέπει αν ο χρήστης χρειάζεται κάποια από τις προαναφερθείσες υπηρεσίες βοηθώντας έτσι στην εξοικονόμηση της μπαταρίας της συσκευής μέσα απο την αποφυγή της άσκοπης χρήσης αυτών των υπηρεσιών.

Πιο συγκεκριμένα η εφαρμογή που αναπτύχθηκε συλλέγει δεδομένα με διάστημα καταγραφής ένα λεπτό. Τα δεδομένα αυτά συλλέγονται για χρονικό διάστημα πέντε ημερών, με την εφαρμογή να τρέχει στο παρασκήνιο όλο αυτό το διάστημα, ενώ ταυτόχρονα αποθηκεύει όλα τα διαθέσιμα δεδομένα σε ένα αρχειο .csv. Ο τρόπος με

τον οποίον επιτυγχάνεται η εκτέλεση της εφαρμογής στο παρασκήνιο είναι η χρήση του χαρακτηριστικού μια εφαρμογής Android που λέγεται Υπηρεσία (Service). Κάνοντας χρήση του συγκεκριμένου χαρακτηριστικού είναι δυνατή η ανεξάρτητη εκτέλεση της εφαρμογής χωρίς να εμποδίζει τον κάτοχο της συσκευής από το να χρησιμοποιεί κανονικά τη συσκευή του. Η κατανάλωση πόρων από τη συγκεριμένη εφαρμογή είναι αμελητέα και με αυτόν τον τρόπο η εμπειρία χρήσης της συσκευής παραμένει ανεπηρέαστη. Η συγκεκριμένη εφαρμογή ασχολείται κυρίως με δεδομένα που αφορούν πτυχές της συσκευής και όχι τόσο των εφαρμογών που έχει εγκαταστήσει ο κάθε χρήστης, κάτι το οποίο είναι εύκολα κατανοήτο αν αναλογιστεί κανείς ότι η εφαρμογή αποκτάει πρόσβαση σε δεδομένα που έχουν να κάνουν με τους διάφορους αισθητήρες που διαθέτει η συσκευή (πχ αισθητήρας φωτός) ή με το αν δέχεται κάποια κλήση τη στιγμή της καταγραφής η συσκευή.

Το δεύτερο σκέλος της εργασίας αφορά την αξιοποίηση των αποθηκευμένων δεδομένων. Μέσα από την επεξεργασία των δεδομένων αυτών προκύπτουν κάποιες προβλέψεις, με αρκετά υψηλό ποσοστό, που αφορούν την ανάγκη του χρήστη για υπηρεσίες δεδομένων κινητής τηλεφωνίας και τοποθεσίας. Για την ακρίβεια από το συνολικό σετ δεδομένων εξάγονται κάποια χαρακτηριστικά τα οποία έχουν μεγαλύτερη συσχέτιση με τον σκοπό της μελέτης και στη συνέχεια μέσα από κάποια προεπεξεργασία των χαρακτηριστικών αυτών τροφοδοτούνται πέντε αλγόριθμοι μηχανικής μάθησης οι οποίοι δίνουν και το αποτέλεσμα της ζητούμενης πρόβλεψης. Από τα πειράματα που πραγματοποιήθηκαν παρατηρήθηκε ότι το καλύτερο δυνατό αποτέλεσμα στο σύνολο των προβλέψεων δίνει ο αλγόριθμος Κ-Πλησιέστερων Γειτόνων (KNN).

Ως τελικό συμπέρασμα, διαπιστώνεται ότι απο την δραστηριότητα του κάθε χρήστη ενός έξυπνου κινητού Android, η οποία μπορεί να καταγραφεί με τη χρήση κατάλληλου λογισμικού μπορούν να εξαχθούν πολλά συμπεράσματα τα οποία βοηθάνε στη μεγαλύτερη αυτονομία της συσκευής και στην καλύτερη εμπειρία χρήσης. Η συγκεριμένη μελέτη μάλιστα μπορεί να αξιοποιηθεί σε όλες τις διαθέσιμες συσκευές Android ανεξάρτητα από την έκδοση του λογισμικού που χρησιμοποιείται.

7.2 Μελλοντικές Επεκτάσεις

Μέσα από τη μελέτη που πραγματοποιήθηκε στη συγκεκριμένη διπλωματική εργασία προκύπτει ένα μεγάλο εύρος πιθανών επεκτάσεων. Δεδομένου ότι οι χρήστες έξυπνων συσκευών Android αποτελούν τη συντριπτική πλειοψηφία της αγοράς είναι εύκολα κατανοητό ότι η καταγραφή της δραστηριότητας μπορεί να επεκταθεί και σε διαφορετικές πτυχές. Μία τέτοια είναι και η καταγραφή των εφαρμογών που χρησιμοποιεί κατα κύριο λόγο ο χρήστης και πιο συγκεκριμένα η δεδομένη χρονική στιγμή που χρησιμοποιεί την εκάστοτε εφαρμογή. Μέσα από τα δεδομένα που προκύπτουν από τη συγκεκριμένη μελέτη είναι δυνατή η αναγνώριση ενός μοτίβου χρήσης των διαθέσιμων εφαρμογών καθώς και οι προτιμήσεις του χρήστη σε είδη εφαρμογών σε συνδυασμό με τη χρονική στιγμή. Με αυτόν τον τρόπο είναι δυνατή η

ανάπτυξη ενός μοντέλου πρόβλεψης το οποίο θα παρέχει στον χρήστη προτάσεις εφαρμογών για εκτέλεση ανάλογα με τη στιγμή της ημέρας βελτιώνοντας έτσι σε σημαντικό βαθμό την εμπειρία χρήσης καθώς και την εξοικονόμηση ενέργειας.

Παράλληλα μια διαφορετική επέκταση που θα μπορούσε να λάβει χώρα είναι η κατηγοροποίηση των χρηστών μέσα από την αναγνώριση του μοτίβου χρήσης της συσκευής. Με αυτόν τον τρόπο θα ήταν δυνατή η εφαρμογή διαφορετικών αλγορίθμων χρήσης ανάλογα με την κατηγορία που ανήκει ο χρήστης και η εξαγωγή ασφαλέστερων συμπερασμάτων.

Μία ακόμα επέκταση που αφορά κυρίως το δεύτερο σκέλος της διπλωματικής έχει να κάνει με όλους τους χρήστες κινητών συσκευών. Για παράδειγμα εφόσον ήταν διαθέσιμο ένα σύνολο δεδομένων από συσκευές με λειτουργικό σύστημα iOS θα ήταν εξίσου δυνατή η πρόβλεψη των αναγκών του χρήστη για τις διάφορες διαθέσιμες υπηρεσίες και κατά συνέπεια η μεγαλύτερη αυτονομία της συσκευής.

Τέλος, όσον αφορά το κομμάτι των προβλέψεων ανοιχτό παραμένει το ενδεχόμενο χρησιμοποίησης διαφορετικών παραμέτρων στον κάθε αλγόριθμο που χρησιμοποιήθηκε ή ακόμα και η χρήση διαφορετικών αλγορίθμων μηχανικής μάθησης για το συγκεκριμένο πρόβλημα.

Λεξικό Όρων

Στον παρακάτω πίνακα παρουσιάζεται η αγγλική και ελληνική ορολογία ορισμένων χαρακτηριστικών που εμφανίζονται στο συγκεκριμένο κείμενο. Η σειρά με την οποία εμφανίζονται στον πίνακα είναι και η σειρά με την οποία συναντώνται οι συγκεκριμένοι όροι μέσα στο κείμενο.

Αγγλικός Όρος	**Ελληνικός Όρος**
Smartphone	Έξυπνη Συσκευή
Personal Computer – PC	Προσωπικός Υπολογιστής
Internet of Things	Διαδίκτυο των Πραγμάτων
Cloud	Υπολογιστικό Νέφος
Tablet	Έξυπνη Ταμπλέτα
SmartTV	Έξυπνη Τηλεόραση
Hardware	Υλικό
Framework	Πλαίσιο
Interface	Διεπαφή
Linux Kernel	Πυρήνας λειτουργικού συστήματος Linux
Android Runtime	Περιβάλλον εκτέλεσης Android
Multithreading	Πολυνηματικότητα
Hardware Drivers	Προγράμματα οδήγησης υλικού
Hardware Abstraction Layer	Στρώμα Αφαίρεσης Υλικού
Application Programming Interface - API	Διεπαφή Προγραμματισμού Εφαρμογών
Module	Άρθρωμα
Garbage collection	Διαδικασία Συλλογής Σκουπιδιών
Native C/C++ Libraries	Τοπικές Βιβλιοθήκες C/C++
Java API Framework	Πλαίσιο Java API
View System	Σύστημα όψεων
User Interface	Διεπαφή Χρήστη
Resource Manager	Διαχειριστής Πηγών
Notification Manager	Διαχειριστης Ειδοποιήσεων
Activity Manager	Διαχειριστης Δραστηριοτήτων
Content Providers	Πάροχοι Περιεχομένου
System Apps	Εφαρμογές Συστήματος
Component	Συστατικό
Activities	Δραστηριότητες
Services	Υπηρεσίες
Broadcast Receivers	Δέκτες εκπομπής
Content Providers	Πάροχοι Περιεχομένου
Activity Lifecycle	Κύκλος Ζωής Δραστηριότητας
Layouts	Διατάξεις
View	Όψη
View Groups	Ομάδες Όψεων
Widgets	Γραφικά
Buttons	Κουμπιά

Listeners	Ακροατές
Bytecode	Κώδικας Byte
Classloader	Φορτωτής Κλάσεων
Java Virtual Machine – JVM	Εικονική Μηχανή της Java
Security Manager	Διαχειριστής Ασφάλειας
Android package	Πακέτο Android
Server	Εξυπηρετητής
Crash	Τερματισμός
Memory Leaks	Διαρροές Μνήμης
IDE - Integrated Development Environment	Ολοκληρωμένο περιβάλλον ανάπτυξης
Debugging	Διόρθωση Σφαλμάτων
Target variables	Μεταβλητές Στόχου
Classification	Ταξινόμηση
Logistic Regression	Λογιστική Παλινδρόμηση
Sigmoid function	Συνάρτηση Σίγμα
K-Nearest Neighbors (KNN)	Κ-Πλησιέστεροι Γείτονες
Decision Trees	Δέντρα απόφασης
Random Forest	Τυχαίο δάσος
Support Vector Machines – SVMs	Μηχανές Διανυσμάτων Υποστήριξης
Open source	Ελεύθερου Λογισμικού
Comma Separated Values – CSV	Τιμές χωρισμένες με κόμμα
Features	Χαρακτηριστικά
Accuracy	Ακρίβεια
Dataset	Σύνολο Δεδομένων
Train data	Δεδομένα εκμάθησης
Test Data	Δεδομένα Ελέγχου

Βιβλιογραφία

[1] https://www.bankmycell.com/blog/how-many-phones-are-in-the-world

[2] Mitra, R. N., & Agrawal, D. P. (2015). 5G mobile technology: A survey. ICT Express, 1(3), 132–137.

[3] https://www.samsung.com/gr/smartphones/galaxy-s10/buy/

[4] http://gs.statcounter.com/os-market-share/mobile/worldwide

[5] https://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/

[6] https://www.statista.com/statistics/266210/number-of-available-applications-in-the-google-play-store/

[7] Wikipedia, Android (operating system), 2019.

[8] Brad K. Donohoo, Student Member, IEEE, Chris Ohlsen, Student Member, IEEE, Sudeep Pasricha, Member, IEEE, Yi Xiang, Student Member, IEEE, and Charles Anderson, Member, IEEE, IEEE TRANSACTIONS ON MOBILE COMPUTING, VOL. 13, NO. 8, AUGUST 2014.

[9] Cañete, A., Horcas, J.-M., Ayala, I., & Fuentes, L. (2019). Energy Efficient Adaptation Engines for Android Applications. Information and Software Technology, 106220.

[10] H. Cao, M. Lin, Mining smartphone data for app usage prediction and recommendations: A survey, Pervasive and Mobile Computing (2017).

[11] Piotr Nawrocki, Bartlomiej Sniezynski (2017) Autonomous Context-Based Service Optimization in Mobile Cloud Computing, J Grid Computing (2017) 15:343–356 DOI 10.1007/s10723-017-9406-2.

[12] Ahmed, E., Gani, A., Sookhak, M., Hamid, S. H. A., & Xia, F. (2015). Application optimization in mobile cloud computing: Motivation, taxonomies, and open challenges. Journal of Network and Computer Applications, 52, 52–68.

[13] I. Constandache, S. Gaonkar, M. Sayler, R. R. Choudhury, and L. Cox, “EnLoc: Energy-efficient localization for mobile phones,” in *Proc. INFOCOM*, Rio de Janeiro, Brazil, Jun. 2009, pp. 19–25.

[14] C. Lee, M. Lee, and D. Han, “Energy efficient location logging for mobile device,” in *Proc. SAINT*, Seoul, Korea, Oct. 2010, p. 84.

[15] J. Kang, S. Seo, and J. W. Hong, “Usage pattern analysis of smartphones,” in *Proc. APNOMS*, Taipei, Taiwan, Nov. 2011, pp. 1–8.

[16] A. Shye, B. Scholbrock, G. Memik, Into the wild: studying real user activity patterns to guide power optimizations for mobile architectures, in: Proceedings Int. Symp. Microarchitecture, 2009, pp. 168–178.

[17] M. Kennedy, H. Venkataraman, G.M. Muntean, Energy consumption analysis and adaptive energy saving solutions for mobile device applications, in: Jae. Kim (Ed.), Green IT: Technologies and Applications, Springer-Verlag, Berlin, 2011, pp. 173–189.

[18] Pandey, N., Verma, O. P., & Kumar, A. (2018). *Nature Inspired Power Optimization in smartphones. Swarm and Evolutionary Computation.*

[19] H. Petander, "Energy-aware network selection using traffic estimation," in *Proc. MICNET*, New York, NY, USA, Sep. 2009, pp. 55–60.

[20] M. Ra *et al.*, "Energy-delay tradeoffs in smartphone applications," in *Proc. MobiSys*, New York, NY, USA, Jun. 2010, pp. 255–270.

[21] K. Lin, A. Kansal, D. Lymberopoulos, and F. Zhao, "Energyaccuracy trade-off for continuous mobile device location," in *Proc. MobiSys*, New York, NY, USA, Jun. 2010, pp. 285–298.

[22] F. B. Abdesslem, A. Phillips, and T. Henderson, "Less is more: Energy-efficient mobile sensing with senseless," in *Proc. MobiHeld*, New York, NY, USA, Aug. 2009, pp. 61–62.

[23] J. Paek, J. Kim, and R. Govindan, "Energy-efficient rate-adaptive GPS-based positioning for smartphones," in *Proc. MobiSys*, New York, NY, USA, Jun. 2010, pp. 299–314.

[24] I. Shafer and M. L. Chang, "Movement detection for powerefficient smartphone WLAN localization," in *Proc. MSWIM*, New York, NY, USA, Oct. 2010, pp. 81–90.

[25] M. Youssef, M. A. Yosef, and M. El-Derini, "GAC: Energy-efficient hybrid GPS-accelerometer-compass GSM localization," in *Proc. GLOBECOM*, Miami, FL, USA, Dec. 2010, pp. 1–5.

[26] K. Nishihara, K. Ishizaka, and J. Sakai, "Power saving in mobile devices using context-aware resource control," in *Proc. ICNC*, Higashi-Hiroshima, Japan, 2010, pp. 220–226.

[27] Z. Zhuang, K. Kim, and J. P. Singh, "Improving energy efficiency of location sensing on smartphones," in *Proc. MobiSys*, New York, NY, USA, Jun. 2010, pp. 315–330.

[28] Y. Wang *et al*., "A framework of energy efficient mobile sensing for automatic user state recognition," in *Proc. MobiSys*, NewYork, NY, USA, 2009, pp. 179–192.

[29] L. Batyuk, C. Scheel, S. A. Camtepe, and S. Albayrak, "Context-aware device self-configuration using self-organizing maps," in *Proc. OC*, New York, NY, USA, Jun. 2011, pp. 13–22.

[30] T. Anagnostopoulos, C. Anagnostopoulos, S. Hadjiefthymiades, M. Kyriakakos, and A. Kalousis, "Predicting the location of mobile users: A machine learning approach," in *Proc. ICPS*, New York, NY, USA, Jul. 2009, pp. 65–72.

[31] https://developer.android.com/guide/platform

[32] https://developer.android.com/guide/components/fundamentals

[33] https://developer.android.com/reference/android/app/Activity

[34] https://androidclarified.com/android-service-lifecycle-and-working/

[35] Wikipedia, Java (programming language), 2018

[36] https://www.tutorialspoint.com/java/index.htm

[37] https://developer.android.com/studio/releases/index.html

[38] E. Alpaydm, Introduction to Machine Learning, The MIT Press, Massachusetts, 2004.

[39] Peng, C.-Y. J., Lee, K. L., & Ingersoll, G. M. (2002). An Introduction to Logistic Regression Analysis and Reporting. The Journal of Educational Research, 96(1), 3–14.

[40] Saini, I., Singh, D., & Khosla, A. (2013). QRS detection using K-Nearest Neighbor algorithm (KNN) and evaluation on standard ECG databases. Journal of Advanced Research, 4(4), 331–344.

[41] Esposito, F., Malerba, D., Semeraro, G., & Kay, J. (1997). A comparative analysis of methods for pruning decision trees. IEEE Transactions on Pattern Analysis and Machine Intelligence, 19(5), 476–493.

[42] Lin, W., Wu, Z., Lin, L., Wen, A., & Li, J. (2017). An Ensemble Random Forest Algorithm for Insurance Big Data Analysis. IEEE Access, 5, 16568–16575.

[43] Suykens, J. A. K. (2014). Introduction to Machine Learning. Academic Press Library in Signal Processing: Volume 1 - Signal Processing Theory and Machine Learning, 765–773.

[44] https://www.geeksforgeeks.org/history-of-python/

[45] https://towardsdatascience.com/top-10-python-libraries-for-data-science-cd82294ec266

[46]https://developers.google.com/machine-learning/crash-course/classification/accuracy