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

Ανίχνευση Κακόβουλων Χρηστών σε Κοινωνικά Δίκτυα μέσω Μεθόδων Βαθιάς Μάθησης

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

ΛΟΥΚΑ Κ. ΗΛΙΑ

Επιβλέπουσα : Ιωάννα Ρουσσάκη
Επίκουρη Καθηγήτρια Ε.Μ.Π

Αθήνα, Ιούλιος 2020

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

Ανίχνευση Κακόβουλων Χρηστών σε Κοινωνικά Δίκτυα μέσω Μεθόδων Βαθιάς Μάθησης

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

ΛΟΥΚΑ Κ. ΗΛΙΑ

Επιβλέπουσα : Ιωάννα Ρουσσάκη
Επίκουρη Καθηγήτρια

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 20η Ιουλίου 2020.

(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*
.....	.....	.....
Ιωάννα Ρουσσάκη	Μιλτιάδης Αναγνώστου	Συμεών Παπαβασιλείου
Επίκουρη Καθηγήτρια	Καθηγητής Ε.Μ.Π	Καθηγητής Ε.Μ.Π
Ε.Μ.Π		

Αθήνα, Ιούλιος 2020

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

(Υπογραφή)

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

Περίληψη

Αναμφισβήτητα, τα μέσα κοινωνικής δικτύωσης, όπως το Facebook και το Twitter, αποτελούν αναπόσπαστο κομμάτι της καθημερινότητάς μας λόγω των ποικίλων δυνατοτήτων, που προσφέρουν. Ειδικότερα, το Twitter παρέχει τη δυνατότητα στους χρήστες μέσω των tweets, σύντομων κειμένων μήκους έως 280 χαρακτήρες, να εκφράζουν τις απόψεις και τις σκέψεις τους σε θέματα της επικαιρότητας, διαμορφώνοντας κατ' αυτόν τον τρόπο τις τάσεις ή ακόμα και να συνομιλούν με χρήστες από όλο τον κόσμο άμεσα και χωρίς κανένα κόστος. Ωστόσο, το Twitter και γενικότερα τα μέσα κοινωνικής δικτύωσης αποτελούν αντικείμενο έλξης αυτοματοποιημένων λογαριασμών, γνωστών ως bots, οι οποίοι έχουν ως απώτερο στόχο την παραπληροφόρηση του χρήστη μέσω της διάδοσης ψευδών ειδήσεων, την προώθηση συγκεκριμένων προϊόντων και ιδεών καθώς και τη διακίνηση, πολλές φορές, υλικού πορνογραφικού περιεχομένου μέσω των ιστοσελίδων, που δημοσιεύουν στα tweets τους. Καθίσταται, λοιπόν, σαφές ότι αποτελεί αδήριτη ανάγκη η έγκαιρη ανίχνευση των bots.

Αντικείμενο της παρούσας διπλωματικής εργασίας αποτελεί η προστασία των χρηστών του Twitter από κακόβουλους χρήστες. Συγκεκριμένα, προτείνονται δύο μέθοδοι κατηγοριοποίησης των χρηστών του Twitter σε αληθινούς χρήστες και αυτοματοποιημένους λογαριασμούς.

Κατά την πρώτη μέθοδο συλλέγεται ένας μεγάλος αριθμός χαρακτηριστικών ανά χρήστη, που έχουν χρησιμοποιηθεί σε πρόσφατες ερευνητικές εργασίες για την ανίχνευση των bots. Αφού υλοποιούμε διάφορες τεχνικές επιλογής χαρακτηριστικών και μεθόδους δειγματοληψίας με σκοπό την εύρεση του βέλτιστου υποσυνόλου χαρακτηριστικών και τη δημιουργία ενός ομοιόμορφου συνόλου δεδομένων αντίστοιχα, αξιολογούμε την επίδοση του κάθε υποσυνόλου με χρήση αλγορίθμων Μηχανικής Μάθησης.

Κατά τη δεύτερη μέθοδο, κάνοντας χρήση μόνο των tweets των χρηστών υλοποιούμε μία αρχιτεκτονική Βαθιάς Μάθησης, η οποία αποτελείται από Αμφίδρομα Αναδρομικά Νευρωνικά Δίκτυα Μακράς Βραχυπρόθεσμης Μνήμης (BiLSTM) με μηχανισμό προσοχής ακολουθούμενα από dense layers. Με αυτόν τον τρόπο αποφεύγουμε τη χρονοβόρα διαδικασία της εξαγωγής χαρακτηριστικών. Αξιολογούμε την επίδοση του μοντέλου Βαθιάς Μάθησης με διάφορες μετρικές αξιολόγησης.

Και στις δύο μεθόδους χρησιμοποιούμε δύο δημόσια διαθέσιμα σύνολα δεδομένων και παρατηρούμε ότι οι τεχνικές, που εφαρμόζουμε, επιτυγχάνουν ανταγωνιστικές επιδόσεις συγκριτικά με τα αποτελέσματα των ερευνητικών εργασιών περί ανίχνευσης των bots, που έχουν δημοσιευτεί έως τώρα.

Λέξεις - Κλειδιά

Μέσα Κοινωνικής Δικτύωσης, Twitter, Ανίχνευση bots, Μηχανική Μάθηση, Επιστήμη Δεδομένων, Εξόρυξη Δεδομένων, Ανάλυση Δεδομένων, Τεχνητή Νοημοσύνη, Επιλογή χαρακτηριστικών, Μέθοδοι Δειγματοληψίας, Επεξεργασία Φυσικής Γλώσσας, Βαθιά Μάθηση

Abstract

Undoubtedly, social media, such as Facebook and Twitter, constitute a major part of our everyday life due to the great number of possibilities, they offer. More specifically, Twitter provides the opportunity to users to express their thoughts and opinions about hot issues of the everyday life or communicate with other users from all over the world whenever they want and without any cost. They get these opportunities by posting tweets, texts with a limited number of characters up to 280. However, Twitter and generally online social networks (OSNs) are increasingly used by automated accounts, which are widely known as bots and their main purpose is the dissemination of fake news, the promotion of specific ideas and products, the manipulation of the stock market and even the diffusion of explicit material. Therefore, the early detection of bots in social media is quite essential.

The main object of this diploma thesis is the security of social media from the proliferation of malicious users. More specifically, we propose two methods to distinguish legitimate users from automated accounts.

In the first method we collect a great number of features per user, which have been used in recent research papers for bot detection. After implementing feature selection techniques and sampling methods, in order to find the best subset of features and to deal with imbalanced dataset respectively, we evaluate the performance of each subset by using Machine Learning Algorithms.

In the second method we use only the tweets of the users and we propose a deep neural network consisting of bidirectional long short-term memory (BiLSTM) layers with attention mechanism followed by dense layers. In this way, we avoid the computationally expensive procedure of feature extraction. We evaluate the performance of the architecture using several metrics.

In both methods we use two publicly available datasets and we demonstrate that our approaches can achieve competitive performance compared with existing state-of-the-art bot detection systems.

Keywords

Social media, Twitter, Bots detection, Machine Learning, Data Science, Data Mining, Data Analysis, Artificial Intelligence, Feature Selection, Sampling Methods, Natural Language Processing, Deep Learning

Ευχαριστίες

Αρχικά, θα ήθελα να ευχαριστήσω θερμά την επιβλέπουσα καθηγήτρια κ. Ιωάννα Ρουσσάκη τόσο για την ευκαιρία, που μου έδωσε να ασχοληθώ με ένα τόσο ενδιαφέρον θέμα, όσο και για την εμπιστοσύνη και το ενδιαφέρον, που μου έδειξε όλους αυτούς τους μήνες. Οι συμβουλές και οι κατευθύνσεις της ήταν πολύτιμες και καθοριστικές για την εκπόνηση αυτής της διπλωματικής εργασίας.

Στη συνέχεια, θα ήθελα να ευχαριστήσω τους φίλους μου για αυτά τα όμορφα φοιτητικά χρόνια.

Κυρίως, θα ήθελα να ευχαριστήσω την οικογένειά μου για την αγάπη, που μου έχουν προσφέρει όλα αυτά τα χρόνια και την υποστήριξή τους σε όλες μου τις επιλογές.

Πίνακας Περιεχομένων

Ευρετήριο Πινάκων

Ευρετήριο Εικόνων

Κεφάλαιο 1

Εισαγωγή

1.1 Κοινωνικά Δίκτυα και Μέσα Κοινωνικής Δικτύωσης

Στα τέλη του 19ου αιώνα τόσο ο Émile Durkheim όσο και ο Ferdinand Tönnies προωθούσαν την ιδέα των κοινωνικών δικτύων στις θεωρίες τους και την έρευνα των κοινωνικών ομάδων. Συγκεκριμένα, ένα κοινωνικό δίκτυο είναι μία κοινωνική δομή, που αποτελείται από ένα σύνολο παραγόντων, όπως άτομα ή οργανισμούς, αλλά και κοινωνικών αλληλεπιδράσεων μεταξύ των παραγόντων αυτών.

Τα μέσα κοινωνικής δικτύωσης (social media) αποτελούν μία διαδικτυακή πλατφόρμα, που οι άνθρωποι χρησιμοποιούν, προκειμένου να διαμορφώσουν κοινωνικούς δεσμούς με άλλους ανθρώπους, με τους οποίους μοιράζονται παρόμοια ενδιαφέροντα τόσο σε προσωπικό όσο και σε επαγγελματικό επίπεδο. Οι ιστοσελίδες κοινωνικής δικτύωσης επιτρέπουν στους χρήστες να μοιράζονται ιδέες, ψηφιακές φωτογραφίες και βίντεο, δημοσιεύσεις αλλά και να ενημερώνουν τους άλλους για γεγονότα που διαδραματίζονται στον κόσμο. Τους παρέχουν, επίσης, τη δυνατότητα να συνδέονται και να επικοινωνούν με άλλους ανθρώπους, που βρίσκονται σε διαφορετική τοποθεσία. Οι πιο γνωστές ιστοσελίδες κοινωνικής δικτύωσης είναι το Facebook, το Instagram, το Pinterest και το Twitter.

1.2 Twitter

Το Twitter δημιουργήθηκε τον Μάρτιο του 2006 από τους Jack Dorsey, Noah Glass, Biz Stone & Evan Williams και ξεκίνησε τον Ιούλιο του έτους αυτού. Στο Twitter οι χρήστες δημοσιεύουν και αλληλεπιδρούν με άλλους χρήστες με μηνύματα, γνωστά ως "tweets". Όσοι έχουν δημιουργήσει λογαριασμό, μπορούν να δημοσιεύουν και να αναδημοσιεύουν (retweet) tweets. Αρχικά το μήκος του περιεχομένου των tweets ήταν περιορισμένο στους 140 χαρακτήρες, αλλά τον Νοέμβριο του 2017 ο αριθμός αυτός διπλασιάστηκε στους 280 χαρακτήρες. Ο κάθε χρήστης ακολουθεί (Following/Friends) τα άτομα για τα οποία θέλει να ενημερώνεται, όταν δημοσιεύουν ένα tweet και αντιστοίχως ακολουθείται και αυτός από άλλους χρήστες (Followers). Οι χρήστες έχουν τη δυνατότητα να απαντήσουν στα tweets άλλων χρηστών (reply) ή να αναφέρουν άλλους χρήστες (mention). Αυτό το κάνουν πληκτρολογώντας στο περιεχόμενο των tweets τους τον χαρακτήρα @ ακολουθούμενο από το όνομα του χρήστη, που επιθυμούν. Επίσης, οι χρήστες του Twitter έχουν τη δυνατότητα να χρησιμοποιήσουν hashtags στα tweets τους, δηλαδή λέξεις ή φράσεις που ξεκινούν με το σύμβολο # και χρησιμοποιούνται για την ομαδοποίηση tweets με βάση το θέμα τους. Τέλος υπάρχει η δυνατότητα για ανταλλαγή απευθείας ιδιωτικών μηνυμάτων μεταξύ των χρηστών (direct messages).

Επιπλέον, ένας χρήστης μπορεί να αναζητήσει με βάση ένα hashtag ή οποιαδήποτε λέξη/φράση όλα τα tweets, που έχουν γίνει σε αντίστροφη χρονολογική σειρά (στην ίδια λογική με το timeline), άσχετα με το αν ακολουθεί τους λογαριασμούς που δημοσίευσαν αυτά τα tweets. Μάλιστα, στο timeline του ο κάθε χρήστης μπορεί να δει ποια θέματα είναι πιο δημοφιλή εκείνη τη στιγμή (τα λεγόμενα trends), και απευθείας να δει τη συζήτηση πάνω σε αυτά. Τα trends καθορίζονται από αλγορίθμους του Twitter για κάθε χρήστη με βάση αυτούς που ακολουθεί, τα ενδιαφέροντά του και την τοποθεσία του (αν και ο χρήστης μπορεί να επιλέξει να βλέπει τα trends για μια δεδομένη γεωγραφική περιοχή). Σε κάθε περίπτωση, ο χρήστης βλέπει θέματα, τα οποία εμφανίζονται ως hashtag ή λέξεις/φράσεις και τα οποία είναι δημοφιλή εκείνη τη στιγμή, και όχι αυτά που είναι γενικά δημοφιλή σε ένα μεγάλο χρονικό διάστημα, ή σε καθημερινή βάση.

1.3 Bots στο Twitter

Η προέλευση του 'Bot' είναι 'Social Robot'. Ως bot ορίζεται κάθε αυτοματοποιημένος λογαριασμός στα μέσα κοινωνικής δικτύωσης, που ελέγχεται από το λογισμικό αλλά συμπεριφέρεται και ενεργεί σαν αληθινός χρήστης. Στόχος του αποτελεί να επηρεάσει τη συζήτηση, τις απόψεις των άλλων ανθρώπων, να υποστηρίξει συγκεκριμένες ιδέες και να προωθήσει ένα συγκεκριμένο προϊόν.

Συγκεκριμένα, τα bots έχουν χρησιμοποιηθεί, για να επηρεάσουν πολιτικές εκλογές διαστρεβλώνοντας τον ορθά πολιτικό λόγο, για να χειραγωγήσουν το χρηματιστήριο ή ακόμα και για να προκαλέσουν θεωρίες συνωμοσίας κατά των εμβολίων ισχυρίζοντας ότι προκάλεσαν σοβαρές ασθένειες. Δημοσιεύουν tweets με στόχο τη διαφήμιση ενός συγκεκριμένου προϊόντος και την εξαπάτηση των

άλλων χρηστών. Συχνά, μάλιστα, αποτελούν κίνδυνο για τη δημόσια υγεία, ιδίως όταν οι άνθρωποι τείνουν να λάβουν ιατρικές συμβουλές σε μέσα κοινωνικής δικτύωσης και τα bots προωθούν συγκεκριμένα φάρμακα. Προκειμένου, μάλιστα, να μη γίνουν αντιληπτά από τους χρήστες, δημοσιεύουν κάθε φορά διαφορετικές URLs, που παραπέμπουν όμως σε ιστοσελίδες με το ίδιο κακόβουλο περιεχόμενο (λήψεις προγραμμάτων επικίνδυνου περιεχομένου, ναρκωτικά κτλ).

Ο Lutz Finger διατυπώνει πέντε άμεσες χρήσεις για social bots. Ως social bots ορίζονται οι αυτοματοποιημένοι λογαριασμοί, που μιμούνται την ανθρώπινη συμπεριφορά.

- Τα bots μπορούν να κάνουν κάποιον χρήστη να φαίνεται δημοφιλής. Και αυτό, γιατί μπορούν να χρησιμοποιηθούν ως fake followers.
- Spamming: Χρησιμοποιούνται ως διαφημιστικά, προκειμένου να πείσουν τον χρήστη να αγοράσει ένα συγκεκριμένο προϊόν.
- Mischief: Τα bots μπορούν να χρησιμοποιηθούν, για να βλάψουν άλλα άτομα (ανταγωνιστές). Οι Newt Gingrich, Mitt Romney & German Conservative Party (CDU) αποτελούν μόνο ορισμένα παραδείγματα περιπτώσεων, όπου οι ψεύτικοι ακόλουθοι ανακαλύφθηκαν και το γεγονός αυτό τους επηρέασε αρνητικά.
- Επηρεάζουν την κοινή γνώμη. Συγκεκριμένα, επηρεάζουν τις τάσεις (trends), δημοσιεύοντας αμέτρητα μηνύματα παρόμοιου περιεχομένου με διαφορετικές φράσεις, προκειμένου να μη γίνουν αντιληπτά.
- Περιορισμός ελεύθερου λόγου. Συχνά, τα bots δημοσιεύουν μηνύματα, προκειμένου σημαντικές δημοσιεύσεις να εμφανίζονται χαμηλότερα στη σελίδα, αποπροσανατολίζοντας έτσι την κοινή γνώμη.

1.4 Συνεισφορά Διπλωματικής

Σε αντίθεση με προηγούμενες έρευνες, που εστιάζουν περισσότερο στη χρήση περιορισμένου αριθμού χαρακτηριστικών και την εκπαίδευση "παραδοσιακών" αλγορίθμων Μηχανικής Μάθησης, στην εργασία αυτή προτείνουμε δύο μεθόδους κατηγοριοποίησης των χρηστών του Twitter σε humans & bots. Στην πρώτη μέθοδο, αφού κάναμε μία ολοκληρωμένη μελέτη της βιβλιογραφίας, συγκεντρώσαμε έναν μεγάλο αριθμό features, προκειμένου να κατηγοριοποιήσουμε τους χρήστες του Twitter σε αληθινούς ή social bots. Υλοποιήσαμε αρκετές τεχνικές επιλογής χαρακτηριστικών για την εύρεση του βέλτιστου υποσυνόλου χαρακτηριστικών. Στη δεύτερη μέθοδο, προτείνουμε ένα μοντέλο βαθιάς μάθησης, που χρησιμοποιεί μόνο τα tweets χωρίς να απαιτεί δηλαδή τη διαδικασία της εξαγωγής χαρακτηριστικών. Αναλυτικά, η εργασία μας παρουσιάζεται στα επόμενα βήματα:

- Χρησιμοποιήσαμε δύο δημόσια διαθέσιμα datasets, τα οποία περιγράφουν τους λογαριασμούς ως humans ή bots. Λόγω των ποικίλων χαρακτηριστικών, που διαθέτουν, αλλά και του μεγάλου αριθμού λογαριασμών και tweets, αποτελούν ένα αντιπροσωπευτικό δείγμα των χρηστών του Twitter.
- Αρχικά, αφού εφαρμόσαμε τεχνικές επιλογής χαρακτηριστικών (feature selection techniques), βρήκαμε τα κατάλληλα υποσύνολα χαρακτηριστικών, τα οποία αποτέλεσαν είσοδο σε διαφορετικούς αλγορίθμους μηχανικής μάθησης. Στη συνέχεια, συγκρίναμε την επίδοση των αλγορίθμων αυτών με διάφορες μετρικές αξιολόγησης.
- Τέλος, εφαρμόσαμε μία αρχιτεκτονική βαθιάς μάθησης, η οποία δέχεται ως είσοδο μόνο tweets και εξάγει αν το tweet αυτό ανήκει σε human ή bot. Συγκεκριμένα, χρησιμοποιήσαμε 2 επίπεδα Bidirectional LSTMs με Attention Mechanism.

1.5 Οργάνωση Κειμένου

Στο Κεφάλαιο 1 δόθηκε μία σύντομη εισαγωγή και περιγραφή του προβλήματος, που πραγματεύεται η παρούσα εργασία, ενώ ορίστηκε και η συνεισφορά της.

Στο Κεφάλαιο 2 πραγματοποιείται μία πλήρης βιβλιογραφική επισκόπηση των μεθόδων, που έχουν χρησιμοποιηθεί για την κατηγοριοποίηση των χρηστών των social media σε real users & bots.

Στο Κεφάλαιο 3 παρουσιάζεται το θεωρητικό υπόβαθρο των αλγορίθμων Μηχανικής Μάθησης, που χρησιμοποιούνται στην εργασία. Παρουσιάζονται, επίσης, κάποιες τεχνικές επιλογής χαρακτηριστικών καθώς και μέθοδοι υπερδειγματοληψίας/υποδειγματοληψίας για τη δημιουργία ενός ομοιόμορφου συνόλου δεδομένων. Στη συνέχεια, επεξηγούνται οι βασικές αρχές λειτουργίας των Νευρωνικών Δικτύων και των τεχνικών βαθιάς μάθησης (deep learning).

Στο Κεφάλαιο 4 παρατίθενται τεχνικές, που εμπίπτουν στο πεδίο της Επεξεργασίας Φυσικής Γλώσσας (Natural Language Processing - NLP) και αφορούν διάφορους τρόπους αναπαράστασης κειμένου σε μαθηματική μορφή.

Στο Κεφάλαιο 5 παρουσιάζεται το σύνολο δεδομένων (dataset), που χρησιμοποιήθηκε στην παρούσα εργασία.

Στο Κεφάλαιο 6 εξάγουμε τα χαρακτηριστικά από το dataset και εφαρμόζοντας διάφορες τεχνικές επιλογής χαρακτηριστικών, δίνουμε τα κατάλληλα υποσύνολα χαρακτηριστικών ως είσοδο στους αλγορίθμους μηχανικής μάθησης. Στο κεφάλαιο αυτό γίνεται παρουσίαση και σύγκριση των αποτελεσμάτων για όλα τα μοντέλα, που εξετάστηκαν.

Στο Κεφάλαιο 7 παρουσιάζουμε το μοντέλο βαθιάς μάθησης, που υλοποιήσαμε, προκειμένου να συμπεράνουμε αν ένα tweet ανήκει σε real user ή bot.

Στο Κεφάλαιο 8 καταγράφουμε τα βασικά συμπεράσματα της εργασίας, καθώς και προτάσεις για μελλοντική έρευνα.

Κεφάλαιο 2

Συναφής Βιβλιογραφία

Στο Κεφάλαιο αυτό, θα γίνει μία πλήρης βιβλιογραφική επισκόπηση των μεθόδων, που έχουν χρησιμοποιηθεί για την ανίχνευση των bots στα μέσα κοινωνικής δικτύωσης. Θα πραγματοποιηθεί, επίσης, μία συγκριτική μελέτη των μετρικών αξιολόγησης - αποτελεσμάτων, που πέτυχε η κάθε εργασία, η οποία θα μας οδηγήσει στην κατασκευή των δικών μας μοντέλων, που θα βελτιώσουν τις μετρικές αυτές.

Οι F.Benevenuto et al. [1] συνέκριναν δύο προσεγγίσεις, προκειμένου να ανιχνεύσουν spam χρήστες και tweets στο Twitter. Αρχικά, χρησιμοποίησαν χαρακτηριστικά βασισμένα στον χρήστη (user-based features), για να κατηγοριοποιήσουν τους χρήστες σε spammers & non-spammers και κάνοντας χρήση του SVM classifier πέτυχαν ακρίβεια ίση με 84.6 %. Στη συνέχεια, χρησιμοποίησαν χαρακτηριστικά βασισμένα τόσο στον χρήστη όσο και στο περιεχόμενο των tweets, προκειμένου να διαχωρίσουν τα tweets σε spam & non-spam. Χρησιμοποιώντας πάλι τον SVM classifier, πέτυχαν ακρίβεια ίση με 87.6%. Συγκρίνοντας, λοιπόν, τις δύο αυτές τεχνικές, κατέληξαν ότι η χρήση χαρακτηριστικών βασισμένων τόσο στον χρήστη όσο και στο περιεχόμενο των tweets είναι περισσότερο αποδοτική στην ανίχνευση των spam tweets.

Οι Lee et al. [2] χρησιμοποίησαν χαρακτηριστικά βασισμένα τόσο στον χρήστη όσο και στο περιεχόμενο των tweets, προκειμένου να κατηγοριοποιήσουν τους χρήστες σε spammers & non-spammers. Για την ταξινόμηση χρησιμοποίησαν τους ακόλουθους αλγορίθμους επιβλεπόμενης μηχανικής μάθησης: SVM, Decorate, Simple Logistic & Decision Trees. Χρησιμοποίησαν δύο διαφορετικά datasets για την ταξινόμηση, ένα με 10 % spammers και 90 % non-spammers και ένα άλλο με 90 % spammers και 10 % non-spammers. Κατέληξαν ότι οι μετρικές αξιολόγησης είναι εύρωστες στις αλλαγές αυτές. Στο πείραμά τους, ο Decorate classifier πέτυχε την υψηλότερη ακρίβεια, ίση με 88.98 %.

Οι Liu et al. [3] πρότειναν μία μέθοδο βασισμένη αποκλειστικά στο περιεχόμενο των tweets. Συγκεκριμένα, με χρήση LDA βρήκαν την πιθανότητα ο κάθε χρήστης να ενδιαφέρεται για συγκεκριμένο αριθμό topics. Ο κύριος στόχος τους ήταν να πετύχουν κοντινές τιμές στις μετρικές recall & precision, αυξάνοντας το recall στην ανίχνευση των bots. Κάνοντας χρήση του Adaboost classifier, προτείνουν ότι η καλύτερη επίδοση είναι για αριθμό topics στην εκπαίδευση του LDA model ίσου με 200. Πέτυχαν F1 measure ίσο με 76.55 % & 14.69 %, κάνοντας χρήση του Lybia Honeyspot dataset & του Libya Dataset αντίστοιχα.

Οι Tavares et al. [4] πρότειναν μία προσέγγιση βασισμένη στη συχνότητα των δημοσιεύσεων του χρήστη. Με χρήση αλγορίθμων επιβλεπόμενης μάθησης, κατηγοριοποίησαν τους χρήστες σε humans, bots, cyborgs, πετυχαίνοντας F1-measure ίσο με 88 %. Οι Andriotis et al. [5] χρησιμοποίησαν χαρακτηριστικά βασισμένα στο περιεχόμενο των tweets, στον χρήστη, στην ανάλυση συναισθήματος και εξήγαγαν με την εκπαίδευση LDA model τα topics, που ενδιαφέρουν τους χρήστες του Twitter. Ως ταξινομητές χρησιμοποίησαν τους k Nearest Neighbors, Decision Trees, Gaussian Naive Bayes, SVM, Random Forest, Adaboost Classifier. Πέτυχαν το καλύτερο F1-score ίσο με 0.950631 με χρήση του Adaboost και συνδυασμό όλων των features.

Οι Ferrara et al. [6] επιχείρησαν να κατηγοριοποιήσουν τους χρήστες του Twitter σε humans & bots. Αρχικά, χρησιμοποίησαν χαρακτηριστικά βασισμένα στον χρήστη και στα tweets. Ως ταξινομητές χρησιμοποίησαν τους: Logistic Regression, SGD Classifier, Random Forest, AdaBoost, MLP. Για την αντιμετώπιση του προβλήματος των imbalanced datasets χρησιμοποίησαν τις τεχνικές παραγωγής συνθετικών δεδομένων: SMOTE+ENN & SMOTE+TOMEK. Στη συνέχεια, πρότειναν μία νέα προσέγγιση βαθιάς μάθησης. Συγκεκριμένα, χρησιμοποιώντας μόνο τα tweets του χρήστη και κάποια metadata features υλοποίησαν μία αρχιτεκτονική βαθιάς μάθησης. Η αρχιτεκτονική αυτή περιλαμβάνει GloVE embedding layer, LSTM & Dense layers. Πέτυχαν accuracy ίσο με 0.9633 και AUC/ROC ίσο με 0.9643. Ως μετρικές αξιολόγησης στα πειράματά τους χρησιμοποίησαν τις: accuracy, precision, recall, F1-score, AUC/ROC. Χρησιμοποίησαν το dataset των Cresci et al. [7].

Οι Cai et al. [8] πρότειναν ένα μοντέλο (BeDM), που κάνει εφαρμογή των βαθέων νευρωνικών δικτύων, για να ανιχνεύσουν bots στο Twitter. Συγκεκριμένα, με χρήση συνελικτικών νευρωνικών δικτύων (CNN), LSTM και κάνοντας χρήση μόνο των tweets, της συχνότητας και του είδους των δημοσιεύσεων, πέτυχαν F1 score ίσο με 87.32 %. Χρησιμοποίησαν ένα δημόσια διαθέσιμο dataset [3]

Οι Ping et al. [9] πρότειναν το μοντέλο DeBD, που βασίζεται σε νευρωνικά δίκτυα βαθιάς μάθησης,

προκειμένου να ανιχνεύσουν social bots στο Twitter. Αρχικά, το μοντέλο αυτό χρησιμοποιεί συνελικτικά νευρωνικά δίκτυα (CNN), προκειμένου να εξάγει χαρακτηριστικά από το περιεχόμενο των tweets και τη μεταξύ τους σχέση. Στη συνέχεια, χρησιμοποιεί ως είσοδο σε LSTM layers τα tweet metadata, προκειμένου να εξάγει τα χωρικά χαρακτηριστικά. Τα δύο τύπου χαρακτηριστικά αυτά περνάνε από ένα fusing layer και έτσι γίνεται η κατηγοριοποίηση σε human ή bot. Χρησιμοποίησαν το δημόσια διαθέσιμο για έρευνα dataset των Cresci et. al. [7] και πέτυχαν μετρικές περίπου ίσες με 1. Οι Lingam et al. [10] με χρήση ενισχυτικής μάθησης και Deep Q-learning model έκαναν χρήση τριών διαφορετικών dataset: The Fake Project [7], Social Honeypot [11] & User Popularity Band dataset [12]. Χρησιμοποίησαν χαρακτηριστικά βασισμένα στο περιεχόμενο των tweets, στο προφίλ του χρήστη αλλά και σε γράφους. Πέτυχαν την καλύτερη ακρίβεια (precision) ίση με 0.93 με συνδυασμό όλων των χαρακτηριστικών.

Οι Feng Wei & Uyen Trang Nguyen [13] χρησιμοποίησαν μόνο τα tweets του χρήστη και με χρήση word embeddings, τριών επιπέδων Bidirectional LSTM και ενός fully connected softmax layer στην έξοδο πέτυχαν recall ίσο με 0.976. Ως μετρικές αξιολόγησης χρησιμοποίησαν τα: precision, recall, specificity, accuracy, F - Measure, MCC. Χρησιμοποίησαν το dataset των Cresci et. al. [7]. Οι Wu, Liu et. al. [14] εξήγαγαν έντεκα είδη χαρακτηριστικών βασισμένα στον χρήστη, στο περιβάλλον του (followers) καθώς και στο περιεχόμενο των tweets. Προκειμένου να επιτύχουν ίσο αριθμό δεδομένων και στις δύο κλάσεις, χρησιμοποίησαν γεννητικά ανταγωνιστικά δίκτυα (GANs) και δημιούργησαν με αυτόν τον τρόπο "ψεύτικα" χαρακτηριστικά. Στη συνέχεια, εκπαίδευσαν ένα νευρωνικό δίκτυο και αξιολόγησαν την επίδοσή του βάσει των μετρικών αξιολόγησης (accuracy, precision, recall, F - measure) . Χρησιμοποίησαν τα datasets των [15] και [7].

Οι Khaled et al. [16] εξήγαγαν 16 χαρακτηριστικά βασισμένα στη συμπεριφορά του χρήστη, δηλαδή την ύπαρξη ή μη εικόνας προφίλ, τον αριθμό των friends/followers, τον συνολικό αριθμό των tweets & retweets κ.ά. Στη συνέχεια υλοποίησαν τεχνικές εύρεσης των καλύτερων features, που θα αποτελέσουν είσοδο στους αλγορίθμους μηχανικής μάθησης. Αρχικά, υλοποίησαν τη μέθοδο μείωσης διαστάσεων (PCA). Υλοποίησαν, επίσης, τεχνικές επιλογής χαρακτηριστικών (feature selection techniques), και συγκεκριμένα μεθόδους φιλτραρίσματος (filter methods) και περιτυλίγματος (wrapper methods). Έχοντας βρει τον καλύτερο συνδυασμό των features σε κάθε μέθοδο, τα χρησιμοποίησαν ως είσοδο σε Μηχανές Διανυσμάτων Υποστήριξης (Support Vector Machines), νευρωνικό δίκτυο (NN) και στο τέλος συνδύασαν τον αλγόριθμο SVM με το νευρωνικό δίκτυο (SVM - NN). Ως μετρικές αξιολόγησης, χρησιμοποίησαν τα: accuracy, false positive rate, false negative rate. Πέτυχαν το καλύτερο accuracy ίσο με 0.983, χρησιμοποιώντας ως feature subset εκείνο, που προέκυψε από τη συσχέτιση (filter methods), και ως ταξινομητή τον συνδυασμό SVM - NN. Συμπέραναν ότι η μέθοδος PCA δεν πετυχαίνει ικανοποιητικά αποτελέσματα. Χρησιμοποίησαν το dataset των Cresci et. al. [17].

Οι Davis et al. [18] χρησιμοποίησαν πάνω από 1000 χαρακτηριστικά, προκειμένου να κατηγοριοποιήσουν τους χρήστες του Twitter σε real users & bots. Τα χαρακτηριστικά αυτά μπορούν να διακριθούν σε 6 κατηγορίες: (a) network features, (b) user features (γλώσσα, τοποθεσία, χρόνος δημιουργίας λογαριασμού), (c) friends features (αριθμός φίλων, ακολούθων), (d) temporal features (tweet rate), (e) content features (pos tagging), (f) sentiment features. Ως ταξινομητή χρησιμοποίησαν τον αλγόριθμο Τυχαίων Δασών. Πέτυχαν AUC score ίσο με 0.95 υλοποιώντας τη μέθοδο του 10-fold cross validation. Ως δεδομένα χρησιμοποίησαν μία λίστα λογαριασμών [11], που αποτελείται από real users & bots, συνοδευόμενη με τα αντίστοιχα labels και κάνοντας χρήση του Twitter API συνέλλεξαν τα tweets των χρηστών.

Οι Alvari et al. [19] πρότειναν μία μέθοδο βασισμένη σε χαρακτηριστικά (features), προκειμένου να ανιχνεύσουν κακόβουλους χρήστες στο Twitter, γνωστούς και ως pathogenic social media (PSM) accounts. Χρησιμοποίησαν χαρακτηριστικά, που μπορούν να διακριθούν σε τρεις κατηγορίες: (a) user - related information (user activity, profile characteristics), (b) source - related information (information linked via URLs shared by users) και (c) content related information (tweet characteristics). Ως ταξινομητές χρησιμοποίησαν τους: Gradient Boosting Tree, Random Forest, AdaBoost, Logistic Regression, Decision Tree, SVM, Naive Bayes. Επίσης, χρησιμοποίησαν τεχνικές βαθιάς μάθησης (LSTM + Dense), προκειμένου να ανιχνεύσουν αν ένα tweet ανήκει σε κακόβουλο χρήστη. Για την αξιολόγηση των μεθόδων, που υλοποίησαν, χρησιμοποίησαν τη μέθοδο του 10 - fold cross validation και ως μετρικές αξιολόγησης τις F1-macro, F1-score. Χρησιμοποίησαν 3 datasets από τη Λετονία, τη Σουηδία και το Ηνωμένο Βασίλειο.

Οι Kaubiyal & Jain [20] πρότειναν μία μέθοδο κατηγοριοποίησης των χρηστών του Twitter σε real users & bots, βασισμένη στην εξαγωγή χαρακτηριστικών. Συγκεκριμένα, χρησιμοποίησαν χαρακτηριστικά: (a) account based (number of friend/ followers, account age), (b) tweet based (tweet similarity, retweets/tweets, mentions/tweets), (c) ownership detail based features (the amount of

time elapsed since the inititation of the domain and the Twitter account) και (d) URL based (URL length, count of sub-domains). Στη συνέχεια, υλοποίησαν τεχνικές επιλογής χαρακτηριστικών για την εύρεση του βέλτιστου υποσυνόλου των features. Ως ταξινομητές χρησιμοποίησαν τους: Logistic Regression, SVM, Random Forest. Ως μετρικές αξιολόγησης χρησιμοποίησαν τις: accuracy, precision, recall, F-score. Ο αλγόριθμος των Τυχαίων Δασών πέτυχε το καλύτερο accuracy, ίσο με 97.9 %. Για την υλοποίηση των πειραμάτων τους χρησιμοποίησαν το dataset των Cresci et. al. [7].

Οι Koggalahewa et al. [21] προτείνουν μία μέθοδο μη επιβλεπόμενης μάθησης, προκειμένου να κατηγοριοποιήσουν τους χρήστες του Twitter σε spammers & non-spammers. Συγκεκριμένα, βάσει του ενδιαφέροντος, που παρουσιάζουν οι χρήστες του Twitter σε ορισμένα topics, υπολογίζεται η αποδοχή του κάθε χρήστη από τους υπόλοιπους χρήστες. Στη συνέχεια ορίζεται ένα κατώφλι, ίσο με 40 % και εάν η αποδοχή του κάθε χρήστη ξεπερνά το κατώφλι αυτό, τότε ο χρήστης ταξινομείται ως αληθινός χρήστης. Αλλιώς, ταξινομείται ως spammer. Ως μετρικές αξιολόγησης χρησιμοποιούν τις: accuracy, precision, recall. Χρησιμοποιούν τα δημόσια διαθέσιμα datasets: Social Honeypot, HSpam14 [22] & The Fake Project.

Οι Gong et al. [23] πρότειναν μία αρχιτεκτονική βαθιάς μάθησης, προκειμένου να ανιχνεύσουν τους κακόβουλους χρήστες σε ιστοσελίδες κοινωνικής δικτύωσης και συγκεκριμένα το Dianping, μία ιστοσελίδα όπου ο κάθε χρήστης προτείνει τοποθεσίες, εστιατόρια κ.ά. Αρχικά, εξάγουν χαρακτηριστικά, τα οποία εντάσσονται στις παρακάτω κατηγορίες: (α) time series (check ins, reviews per day), (β) spatio-temporal (number of visited cities), (γ) social (friends/followers), (δ) demographic (gender, age), (ε) UCG (average of the ratings, number of uploaded photos). Στη συνέχεια, χρησιμοποιούν τα time - series features ως είσοδο σε νευρωνικό δίκτυο βαθιάς μάθησης, το οποίο αποτελείται από Bi-LSTM layer, Dense Layer και Softmax activation function και λαμβάνουν στην έξοδο την πιθανότητα ο χρήστης να είναι real user ή malicious. Οι δύο αυτές πιθανότητες αποτελούν και τα νέα features, τα οποία χρησιμοποιούν μαζί με τα features (β), (γ), (δ), (ε) ως είσοδο σε αλγορίθμους μηχανικής μάθησης για την τελική αξιολόγηση των πειραμάτων τους. Ως αλγορίθμους μηχανικής μάθησης χρησιμοποίησαν τους: XGBoost, Random Forest, Decision Tree J48, SVM (polynomial & radial basis function kernel). Ως μετρικές αξιολόγησης χρησιμοποίησαν τις: precision, recall, F1-score.

Οι Wald et al. [24] επιχείρησαν να κατηγοριοποιήσουν τους χρήστες του Twitter σε real users & bots προτείνοντας διάφορες τεχνικές επιλογής χαρακτηριστικών και συγκρίνοντας την απόδοσή τους. Αρχικά, εξήγαγαν χαρακτηριστικά βασισμένα στον χρήστη (αριθμός followers/friends, μήκος περιγραφής σε χαρακτήρες) και στα tweets (LIWC, pos tagging, retweets, replies, hashtags). Στη συνέχεια, χρησιμοποίησαν μεθόδους φιλτραρίσματος και περιτυλίγματος για την εύρεση του κατάλληλου υποσυνόλου των χαρακτηριστικών. Ως ταξινομητές χρησιμοποίησαν τους εξής: k-NN, Logistic Regression, Naive Bayes, Random Forest, SVM, MLP. Ως μετρική αξιολόγησης χρησιμοποίησαν το AUC score και ως dataset χρησιμοποίησαν τα δεδομένα από τον οργανισμό Online Privacy Foundation.

Οι Luo et. al. [25] χρησιμοποίησαν μία αρχιτεκτονική βαθιάς μάθησης, προκειμένου να συμπεράνουν αν ένα tweet ανήκει σε real user ή bot. Η αρχιτεκτονική αυτή περιλαμβάνει: Glove Embedding layer, Bi-LSTM, Attention Mechanism, Bi-LSTM, 2 dense layers. Χρησιμοποίησαν το δημόσια διαθέσιμο dataset[1]. Πέτυχαν accuracy ίσο με 79.64 % και ROC ίσο με 87.04 %.

Οι Barbon et al. [26] ανέπτυξαν έναν αλγόριθμο, ο οποίος βασίζεται στον διακριτό μετασχηματισμό κυματιδίων (discrete wavelet transform), προκειμένου να κατηγοριοποιήσουν τους χρήστες των μέσων κοινωνικής δικτύωσης σε humans, legitimate & malicious bots. Αφού εξήγαγαν από το κείμενο τα χαρακτηριστικά, χρησιμοποίησαν τεχνικές επιλογής χαρακτηριστικών (CBFS), έτσι ώστε να βρουν τα βέλτιστα υποσύνολα χαρακτηριστικών. Ως ταξινομητή για τα πειράματά τους χρησιμοποίησαν τον αλγόριθμο των Τυχαίων Δασών. Πέτυχαν accuracy ίσο με 95.05 % & 93.88 % στα $Dataset_S$ & $Dataset_M$ αντίστοιχα αντίστοιχα.

Οι Clark et al. [27] πρότειναν μία προσέγγιση, βασισμένη αποκλειστικά στην εξαγωγή χαρακτηριστικών από το κείμενο, προκειμένου να ανιχνεύσουν δημοσιεύσεις με αυτοματοποιημένο περιεχόμενο. Συγκεκριμένα, εξάγουν τρία διαφορετικά χαρακτηριστικά: Average Pairwise Dissimilarity, Word Introduction Rate Decay Parameter & Average number of URLs per tweet. Η ταξινόμηση των χρηστών γίνεται βάσει της απόστασής τους από το μέσο όρο καθενός από τα τρία ανωτέρω χαρακτηριστικά. Για την αξιολόγηση των πειραμάτων τους χρησιμοποίησαν τη μέθοδο του 10-fold cross validation. Ως μετρική αξιολόγησης χρησιμοποιούν το AUC/ROC.

Οι Zhao et al. [28] προτείνουν μία μέθοδο αντιμετώπισης του προβλήματος των ανομοιογενών συνόλων δεδομένων, βασισμένη στη μάθηση ευαίσθητη στο κόστος (cost sensitive learning). Εξήγα-

[1] https://pan.webis.de/clef19/pan19-web/author-profiling.html

γαν δώδεκα χαρακτηριστικά, εκ των οποίων τα έξι είναι βασισμένα στον χρήστη και τα υπόλοιπα έξι στα tweets του κάθε χρήστη. Πρότειναν μία μέθοδο, η οποία αποτελείται από δύο στάδια. Στο πρώτο στάδιο, εκπαίδευσαν με τα χαρακτηριστικά αυτά έξι αλγορίθμους μηχανικής μάθησης και στη συνέχεια τροφοδότησαν τις εξόδους, τις οποίες χρησιμοποίησαν ως νέα χαρακτηριστικά, σε ένα νευρωνικό δίκτυο βαθιάς μάθησης. Για να αντιμετωπίσουν το πρόβλημα της λανθασμένης ταξινόμησης εξαιτίας του ανομοιογενούς συνόλου δεδομένων, έκαναν χρήση ενός πίνακα κόστους και τροποποίησαν έτσι τη συνάρτηση κόστους του νευρωνικού δικτύου. Χρησιμοποίησαν το δημόσια διαθέσιμο dataset που συλλέχθηκε από τους Chen et al. [29], το οποίο περιέχει 600 εκατομμύρια tweets, από τα οποία τα 6.5 εκατομμύρια είναι tweets με κακόβουλο περιεχόμενο. Ως μετρικές αξιολόγησης χρησιμοποίησαν τις: True Positive Rate, False Positive Rate, Precision, F1-score, G-mean & Kappa.

Οι Dickerson et al. [30] ήταν οι πρώτοι που χρησιμοποίησαν την ανάλυση συναισθήματος ως χαρακτηριστικό, προκειμένου να ανιχνεύσουν bots στο Twitter. Χρησιμοποίησαν χαρακτηριστικά, τα οποία μπορούν να διακριθούν σε τέσσερις επιμέρους κατηγορίες: (a) tweet syntax: avg number of hashtags/mentions/URLs/emoticons , (b) tweet semantics: sentiment analysis, lda topics, (c) user behavior: tweet frequency, geoenabled tweet, (d) user neighborhood: in/out degree . Ως ταξινομητές χρησιμοποίησαν τους: Gaussian Naive Bayes, SVMs, Random Forests, Extremely Randomized Trees, Adaboost & Gradient Boosting Classifier. Πριν το σύνολο των χαρακτηριστικών αποτελέσει είσοδο στους αλγορίθμους μηχανικής μάθησης, υπέστη προεπεξεργασία μέσω τεχνικών μείωσης διαστάσεων (PCA). Ως μετρικές αξιολόγησης χρησιμοποίησαν τις: precision, recall & AUC/ROC. Συμπέραναν ότι η χρήση χαρακτηριστικών σχετικών με την ανάλυση συναισθήματος των tweets βελτιώνει την ακρίβεια των αλγορίθμων μηχανικής μάθησης. Χρησιμοποίησαν το "India Election Dataset" (IEDS), ένα σύνολο δεδομένων, που συλλέχθηκε στο διάστημα 15 Ιουλίου 2013 έως 24 Μαρτίου 2014.

Οι Cresci et al. [31] είναι οι πρώτοι, που προτείνουν τη μέθοδο του Digital DNA, προκειμένου να κατηγοριοποιήσουν τους χρήστες του Twitter σε humans & bots. Εξαιτίας του γεγονότος ότι τα bots στις μέρες μας δεν μπορούν να γίνουν εύκολα αντιληπτά από τις τεχνικές μηχανικής μάθησης, που έχουν αναπτυχθεί έως τώρα, οι Cresci et al. μοντελοποιούν το είδος και το περιεχόμενο των tweets των χρηστών ως μία αλληλουχία χαρακτήρων, που αποτελούν τις βάσεις του DNA. Επομένως, κάθε χρήστης αναπαρίσταται ως μία ακολουθία χαρακτήρων ανάλογα με το είδος και το περιεχόμενο των tweets, που δημοσιεύει. Στη συνέχεια, οι συγγραφείς ορίζουν το μήκος της μεγαλύτερης κοινής συμβολοσειράς (Longest Common Substring - LCS) μεταξύ δύο ακολουθιών. Επεκτείνουν το πρόβλημα αυτό σε M ακολουθίες, όπου ορίζουν το πρόβλημα k- common substring, προκειμένου να βρουν ομοιότητες για τις δύο κατηγορίες χρηστών, humans & bots. Για ένα σύνολο λογαριασμών, που αποτελείται από αληθινούς χρήστες, το μήκος της μεγαλύτερης κοινής υπακολουθίας των DNA ακολουθιών βρέθηκε να είναι πολύ μικρό. Η ιδέα αυτή χρησιμοποιήθηκε, στη συνέχεια, στην ανίχνευση των bots σε ένα σύνολο λογαριασμών, που αποτελείται από αληθινούς χρήστες και bots. Συγκεκριμένα, βρέθηκε η k-common υπακολουθία για όλους τους λογαριασμούς και ορίστηκε ένα κατώφλι, σύμφωνα με το οποίο όλοι εκείνοι οι λογαριασμοί, οι οποίοι μοιράζονται μία μεγάλου μήκους υπακολουθία ταξινομούνται ως bots και οι υπόλοιποι με μικρότεροι μήκους κοινή υπακολουθία ταξινομούνται ως αληθινοί χρήστες. Βασισμένοι σε αυτήν την ιδέα, οι συγγραφείς ανέπτυξαν δύο τεχνικές, μία που στηρίζεται σε επιβλεπόμενη μάθηση και άλλη μία σε μη επιβλεπόμενη, προκειμένου να βρουν σύνολα λογαριασμών που συμπεριφέρονται με παρόμοιο τρόπο. Ως μετρικές αξιολόγησης χρησιμοποίησαν τις: precision, recall, specificity, accuracy, F-Measure & MCC. Χρησιμοποίησαν το dataset των Cresci et al. [7].

Οι Pasricha & Hayes [32] προτείνουν τη μέθοδο του Digital DNA, εμπνευσμένη από τη Βιοπληροφορική, προκειμένου να κατηγοριοποιήσουν τους χρήστες του Twitter σε humans & bots. Αντικατέστησαν κάθε tweet, retweet & reply, κατά χρονολογική σειρά, του χρήστη με τους χαρακτήρες A, C & T, δημιουργώντας έτσι μία αλληλουχία DNA, την οποία και συμπίεσαν. Στη συνέχεια, ως χαρακτηριστικά εξήγαγαν το μέγεθος (σε bytes) της ακολουθίας μετά τη συμπίεση και τον συντελεστή συμπίεσης. Ως ταξινομητή χρησιμοποίησαν τον αλγόριθμο λογιστικής παλινδρόμησης (Logistic Regression). Ως μετρικές αξιολόγησης χρησιμοποίησαν τις: accuracy, precision, recall, F-Measure, MCC & Specificity.

Οι Kosmajac & Keselj [33] βασίστηκαν στην ιδέα μοντελοποίησης της συμπεριφοράς των χρηστών του Twitter με τη μέθοδο του Digital DNA. Αντιστοίχισαν κωδικούς σε κάθε tweet. Συγκεκριμένα, βασιζόμενοι στο είδος του tweet, αντιστοίχισαν τον κωδικό 8 για retweet, 16 για reply και 0 εάν δεν είναι ούτε retweet ούτε reply. Στη συνέχεια, προσέθεσαν στους κωδικούς αυτούς, τους κωδικούς που αντιστοίχισαν για το περιεχόμενο των tweet. Συγκεκριμένα, αντιστοίχισαν τους κωδικούς 1, 2 και 4, εάν το tweet περιέχει hashtags, mentions & urls αντίστοιχα. Με τον τρόπο αυτόν, για κάθε tweet προέκυψε ένας αριθμός. Ο αριθμός αυτός μετατράπηκε σε χαρακτήρα ASCII. Η διαδικασία

αυτή πραγματοποιήθηκε για όλα τα tweets των χρηστών κατά τη χρονολογική σειρά δημοσίευσής τους. Έτσι, δημιουργήθηκε μία αλληλουχία DNA για κάθε χρήστη, από την οποία εξήγαγαν n-grams με n=1,2,3. Στη συνέχεια, όρισαν 5 διαφορετικά στατιστικά μέτρα, προκειμένου να αποτυπωθούν οι διαφορές μεταξύ των δύο κλάσεων και βάσει των οποίων εξήγαγαν τα χαρακτηριστικά, που θα αποτελέσουν είσοδο στους αλγορίθμους μηχανικής μάθησης. Ως αλγορίθμους μηχανικής μάθησης χρησιμοποίησαν τους: Gaussian Naive Bayes, Support Vector Machines, Logistic regression, K Nearest Neighbours, Random Forest & Gradient Booosting. Ως μετρική αξιολόγησης χρησιμοποίησαν το F1-score. Χρησιμοποίησαν τα δημόσια διαθέσιμα datasets, των Cresci et al. [7] και Varol et al. [15].

Οι Martinelli et al. [34] πρότειναν μία μέθοδο βαθιάς μάθησης, προκειμένου να συμπεράνουν εάν ένα tweet ανήκει σε spammer ή non-spammer. Πιο συγκεκριμένα, χρησιμοποίησαν διάφορους MLP ταξινομητές με τον αριθμό των κρυφών επιπέδων να κυμαίνεται μεταξύ του 0 & 4. Ως χαρακτηριστικά (εισόδους στους ταξινομητές) χρησιμοποίησαν προεκπαιδευμένα διανύσματα λέξεων. Αφού αντικατέστησαν κάθε όρο - token του tweet με τη διανυσματική του αναπαράσταση, υπολόγισαν τον μέσο όρο των διανυσματικών αυτών αναπαραστάσεων σε κάθε tweet. Αυτός ο μέσος όρος αποτελεί και το χαρακτηριστικό.

Οι Alom et al. [35] πρότειναν δύο μεθόδους βαθιάς μάθησης, προκειμένου να κατηγοριοποιήσουν τόσο τους χρήστες σε spammers & non-spammers όσο και τα tweets σε spam & non-spam. Αρχικά, πρότειναν μία αρχιτεκτονική βαθιάς μάθησης, η οποία δέχεται ως είσοδο μόνο το tweet και αποτελείται από Embedding & CNN layer, έτσι ώστε να ανιχνεύσουν εάν το tweet δημοσιεύτηκε από spammer ή non-spammer. Στη συνέχεια, πρότειναν μία δεύτερη αρχιτεκτονική βαθιάς μάθησης, η οποία κάνει χρήση της προηγούμενης και ενός νευρωνικού δικτύου, προκειμένου να κατηγοριοποιήσουν τους χρήστες του Twitter σε spammers & non-spammers. Ως εισόδους στο νευρωνικό δίκτυο χρησιμοποίησαν κάποια χαρακτηριστικά (age of the account, followers, friends etc.) του χρήστη, που δημοσίευσε το tweet.

Οι Ashour et al. [36] πρότειναν μία μέθοδο χρησιμοποιώντας n-grams χαρακτήρων, προκειμένου να ανιχνεύσουν spam tweets στο Twitter. Μελέτησαν τις εξής δύο αναπαραστάσεις των n-grams: Term Frequency (tf) & Term frequency-Inverse document frequency (tf-idf). Ως ταξινομητές χρησιμοποίησαν τους: SVM, Random Forest, Logistic Regression. Χρησιμοποίησαν ένα δημόσια διαθέσιμο σύνολο δεδομένων (Social Honeypot Dataset), όπου κράτησαν ένα tweet ανά χρήστη.

Στον παρακάτω πίνακα παρουσιάζεται συνοπτικά επιπλέον συναφής βιβλιογραφία.

Αριθμός Βιβλιογραφικής Παραπομπής	Χαρακτηριστικά που χρησιμοποιήθηκαν	Classifiers	Dataset	Αποτελέσματα - Μετρικές Αξιολόγησης
[37]	**1)** Metadata (retweet ratio, tweet time interval standard deviation), **2)** Content (URL/Mention/ Hashtag ratio), **3)** Interaction (followers, friends), **4)** Community	**1)** Random Forest, **2)** Decision trees, **3)** Bayesian network	"Empirical evaluation and new design for fighting evolving twitter spammers" [37] 10000 benign users 1000 spammers SMOTE techniques	**1)** Recall **2)** false positive rate **3)** F1-Score
[38]	**1)** Bait - oriented features (identification of the techniques used by spammers to grab a victim's attention) **2)** Behavioral - entropy features (identifying patterns in their respective activities) **3)** URL features **4)** Content entropy features	**1)** Decision Tree **2)** Random Forest **3)** Bayes Network **4)** Decorate	**A)** Empirical Evaluation and New Design for Fighting Evolving Twitter Spammers. In Recent Advances in Intrusion Detection (RAID), **B)** Twitter API	**1)** True positive rate **2)** False positive rate
[39]	**1)** Profile similarity indexing between the account owners and his friends/followers **2)** Content based **3)** Timing based	**1)** SVM **2)** Naive Bayes	Twitter API (5000 users with 200 recent tweets)	**1)** Accuracy **2)** Precision **3)** Recall **4)** F - score
[29]	**1)** User based (account age, #followers, #lists) **2)** Tweet based (no_hashtags, no_urls, no_chars)	**1)** Random Forest **2)** C4.5 **3)** Bayes Network **4)** Naive Bayes **5)** kNN **6)** SVM	Twitter API	**1)** True positive rate **2)** False positive rate **3)** F - measure
[40]	**1)** User based (followers, friends, age of the account) **2)** Content based (no_tweets, hashtag/URL/mention ratio, tweet frequency) **3)** Graph based (in/out degree, betweenness)	**1)** J48 **2)** Decorate **3)** Naive Bayes	Dataset provided by Guofey Gu (http://faculty.cs.tamu.edu/guofei/)	**1)** True positive rate **2)** False Positive Rate **3)** Precision
[41]	**1)** Topic features (LDA) **2)** Interest consistency between the description and tweets, **3)** #friends, #followers, **4)** #links, **5)** #unique words, **6)** average "@ username" per tweet	**1)** SVM **2)** J48 **3)** Decision Tree **4)** Random Forest	Honeypot dataset (Uncovering social spammers:social honeypots+ machine learning.)	**1)** Accuracy **2)** Precision **3)** Recall **4)** F1 - score
[42]	**1)** Graph based **2)** User behaviours **3)** Content based	**1)** Naive Bayes **2)** SVM - SMO **3)** MLP **4)** kNN **5)** ADTree **6)** J48 **7)** Random Forest	Twitter API	**1)** Accuracy **2)** Precision **3)** Recall **4)** F - measure

[43]	**1)** User features **2)** Content features **3)** n-gram features **4)** Sentiment features	**1)** Naive bayes **2)** kNN **3)** SVM **4)** Decision Tree **5)** Random Forest	**1)** Social honeypot dataset **2)** 1KS-10KN Dataset	**1)** F1 -measure : 0.94 using Random Forest
[44]	**1)** User based **2)** Content based **3)** Tweeting characteristics (tweet sources)	**1)** Ridge Logistic Regression **2)** SVM (RBF kernel) **3)** XGBoost **4)** SAMME (Stagewise Additive Modeling using Multiclass exponential loss function) **5)** Ensemble	Twitter API	**1)** Precision **2)** Recall **3)** Specificity
[45]	**1)** Profile features **2)** Syntactic & stylistic features **3)** Lexical features **4)** Network features **5)** Sentiment features **6)** Image features	**1)** Log linear models **2)** LSTM	Twitter API	F1-score
[46]	**1)** User profile features **2)** Account information features **3)** Pairwise engagement features: **A)** Engage with features **B)** Engaged by features	**1)** Maximum Entropy **2)** Random Forest **3)** Exremely Randomized Trees (ExtraTrees) **4)** SVC **5)** Gradient Boosting **6)** MLP	**1)** Honeypot **2)** SPDautomated **3)** SPDmanual	**1)** F - score **2)** Precision **3)** Recall **4)** Accuracy **5)** ROC/ AUC
[47]	**1)** Account based **2)** Content based	**1)** Logistic Regression **2)** SVM **3)** Random Forests **4)** MLP **5)** AdaBoost	MIB Dataset (Cresci)	**1)** Accuracy **2)** Precision **3)** Recall **4)** F1 - score **5)** AUC/ ROC
[48]	**1)** User based (friends, followers, profile image, description, screen_name) **2)** Tweet based (levenshtein distance between user's tweets)	SVM	MIB dataset (Cresci) NBC News Russian Bots	**1)** Accuracy **2)** True positive rate **3)** Misclassification rate
[49]	**1)** Dirichlet distribution has been used by statistical framework for identifying spammer in Twitter **2)** Full name length **3)** #media posted **4)** Avg #posts per week **5)** Account lifetime **6)** Idle time in days **7)**Friends, followers **8)** #mentions	**1)** AdaBoost, **2)** Bagging, **3)** Decorate, **4)** LogitBoost, **5)** MultiBoost **6)** ADTree, **7)** Random Forest **8)** RBF Networks and **9)** SVM	Real data of Twitter and Instagram	**1)** Accuracy **2)** Correct detection rate **3)** False alarm rate **4)** F-measure

[50]	User based features only	**1)** Naive Bayes **2)** SVM	450 users 50 % spammers	**1)** Accuracy **2)** Precision **3)** Recall SVM performs better with accuracy: 89.6 %
[51]	**1)** Graph based features **2)** Content based features	**1)** Bayesian Classifier **2)** Neural network **3)** SVM **4)** Decision Trees	500 Twitter users with 20 recent tweets for each 3 % spam accounts	**1)** Precision **2)** Recall **3)** F-Measure
[52]	**1)** User based **2)** Content based	**1)** Random Forest **2)** SMO **3)** Naive bayes **4)** Ibk (kNN equivalent)	1000 users Evaluated using the 20, 50, 100 recent tweet/user	**1)** Precision **2)** Recall **3)** F- measure

Πίνακας 2.1: Συγκεντρωτικός πίνακας βιβλιογραφικής επισκόπησης εργασιών

Κεφάλαιο 3

Μηχανική Μάθηση

Στο Κεφάλαιο αυτό θα παρουσιαστούν κάποιες βασικές αρχές της Μηχανικής Μάθησης, τις οποίες θα χρησιμοποιήσουμε κατά τη διαδικασία εξαγωγής των πειραμάτων μας στην παρούσα εργασία. Συγκεκριμένα, η ενότητα 3.1 πραγματεύεται τους κλάδους, στους οποίους κατηγοριοποιείται η Μηχανική Μάθηση. Στην ενότητα 3.2 αναλύονται οι συνηθέστεροι "παραδοσιακοί" αλγόριθμοι μηχανικής μάθησης. Στην ενότητα 3.3 γίνεται μία εισαγωγή στα νευρωνικά δίκτυα, στις συναρτήσεις ενεργοποίησης και στη συνάρτηση κόστους. Συγχρόνως, δίνεται ο ορισμός των βαθέων νευρωνικών δικτύων (deep neural networks), παρουσιάζονται οι μέθοδοι ανανέωσης των βαρών σε ένα νευρωνικό δίκτυο και στη συνέχεια αναλύεται ο τρόπος λειτουργίας των αναδρομικών νευρωνικών δικτύων. Στην ενότητα 3.4 παρουσιάζονται οι μέθοδοι εύρεσης των βέλτιστων υπερπαραμέτρων σε έναν αλγόριθμο Μηχανικής Μάθησης. Η ενότητα 3.5 πραγματεύεται μεθόδους αξιολόγησης του μοντέλου. Στην ενότητα 3.6 αναφέρονται μέθοδοι υπερδειγματοληψίας και υποδειγματοληψίας για τη δημιουργία ομοιόμορφων συνόλων δεδομένων. Στην ενότητα 3.7 δίνεται μία εισαγωγή σε τεχνικές μείωσης διαστάσεων. Τέλος, το κεφάλαιο ολοκληρώνεται με την ενότητα 3.8, στην οποία αναλύονται οι τεχνικές επιλογής χαρακτηριστικών, που θα συμβάλλουν στη βέλτιστη απόδοση του μοντέλου.

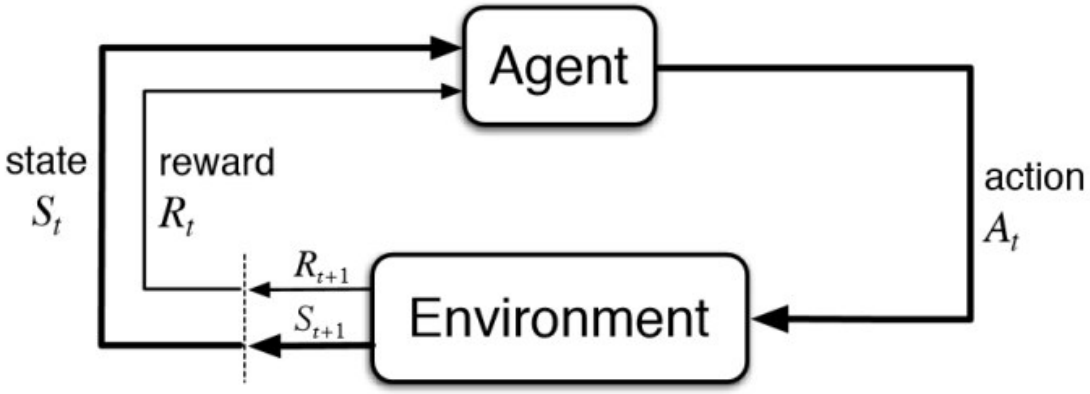
3.1 Κλάδοι Μηχανικής Μάθησης

Η Μηχανική Μάθηση (Machine Learning) είναι ένα πεδίο της Τεχνητής Νοημοσύνης, που παρέχει τη δυνατότητα στα συστήματα να μαθαίνουν από δεδομένα και να κάνουν προβλέψεις με βάση αυτά, χωρίς να έχουν προγραμματιστεί ρητά για τον σκοπό αυτό. Η διαδικασία της μάθησης ξεκινά με παρατήρηση των δεδομένων, προκειμένου να βρεθούν κάποια μοτίβα στα δεδομένα αυτά, κάτι που θα συμβάλλει στη βελτίωση της επίδοσης των συστημάτων. Οι αλγόριθμοι μηχανικής μάθησης χρησιμοποιούνται ευρέως σε πολλές εφαρμογές, όπως στο φιλτράρισμα των email και την όραση υπολογιστών. Διακρίνονται σε 3 μεγάλες κατηγορίες:

- **Επιβλεπόμενη Μάθηση (Supervised Learning):** Σε αυτήν την κατηγορία μηχανικής μάθησης τα δεδομένα εκπαίδευσης αποτελούνται από εισόδους (features), που συνοδεύονται με τις αντίστοιχες τιμές εξόδου (labelled data). Οι αλγόριθμοι επιβλεπόμενης μάθησης διακρίνονται σε αλγορίθμους:
  - **Ταξινόμησης (Classification):** Στην περίπτωση αυτή, η τιμή της εξόδου (target value) παίρνει διακριτές τιμές (discrete values). Κατά τη διάρκεια της εκπαίδευσης, ο αλγόριθμος θα ψάξει για patterns στα δεδομένα εισόδου που έχουν υψηλή συσχέτιση με τις επιθυμητές εξόδους. Αφού εκπαιδευτεί ο αλγόριθμος, θα του δοθούν στη συνέχεια ως είσοδοι, χαρακτηριστικά (features), που δεν έχει δει και θα προβλέψει την τιμή εξόδου των δεδομένων αυτών. Στόχος των αλγορίθμων επιβλεπόμενης μάθησης είναι να προβλέψει τη σωστή κλάση, στην οποία ανήκουν τα δεδομένα. Την επίδοση του αλγορίθμου την αξιολογούμε με μετρικές, όπως θα δούμε στην επόμενη ενότητα. Παραδείγματα classification είναι:
    - κατηγοριοποίηση των χρηστών των μέσων κοινωνικής δικτύωσης σε κακόβουλους ή μη
    - χαρακτηρισμός εάν ένας όγκος είναι καλοήθης ή κακοήθης
    - κατηγοριοποίηση των email σε spam ή non-spam
    - πρόβλεψη φρούτου από τα χαρακτηριστικά του, αν είναι για παράδειγμα μήλο, μπανάνα, ανανάς (multi-label classification)
  - **Παλινδρόμησης (Regression):** Στην περίπτωση αυτή η τιμή της εξόδου (target value) παίρνει συνεχείς τιμές (continuous values). Παραδείγματα αποτελούν η πρόβλεψη της τιμής ενός σπιτιού, η εκτίμηση της τηλεθέασης ενός προγράμματος, η πιθανότητα ο πελάτης να αγοράσει ένα προϊόν.
- **Μη Επιβλεπόμενη Μάθηση (Unsupervised Learning):** Σε αυτήν την κατηγορία μηχανικής μάθησης, τα δεδομένα εκπαίδευσης αποτελούνται από εισόδους (features), που δεν

συνοδεύονται από τις αντίστοιχες τιμές εξόδου (unlabelled data). Κύριος στόχος των αλγορίθμων μη επιβλεπόμενης μάθησης είναι να βρει ομάδες χαρακτηριστικών, που ακολουθούν παρόμοιο μοτίβο (clustering).

- **Ενισχυτική Μάθηση (Reinforcement Learning):** Αφορά έναν πράκτορα, που βρίσκεται σε διαρκή αλληλεπίδραση με το περιβάλλον και προσπαθεί να πάρει αποφάσεις, έτσι ώστε να μεγιστοποιήσει την ανταμοιβή σε μία συγκεκριμένη κατάσταση. Ενώ στην επιβλεπόμενη μάθηση το μοντέλο εκπαιδεύεται γνωρίζοντας τις τιμές εξόδου, στην ενισχυτική μάθηση ο πράκτορας αποφασίζει τι θα κάνει βασιζόμενος στην εμπειρία του. Η ενισχυτική μάθηση βρίσκει εφαρμογές σε συστήματα ελέγχου, πολυπρακτορικά συστήματα κ.α. Η παρακάτω εικόνα δείχνει την αλληλεπίδραση του πράκτορα με το περιβάλλον:



Εικόνα 3.1: Ενισχυτική Μάθηση (Αλληλεπίδραση του πράκτορα με το περιβάλλον

3.2 Αλγόριθμοι Μηχανικής Μάθησης

3.2.1 Λογιστική Παλινδρόμηση (Logistic Regression)

Πρόκειται για έναν αλγόριθμο, που χρησιμοποιείται σε προβλήματα ταξινόμησης. Η απλούστερη μορφή του αλγορίθμου αυτού είναι η δυαδική ταξινόμηση, όπου η έξοδος y παίρνει δύο διακριτές τιμές (συνηθέστερα 0 & 1). Τότε, η υπόθεση $h(x)$ μπορεί να εκφραστεί από την ακόλουθη σιγμοειδή συνάρτηση:

$$h(x) = \frac{1}{1 + e^{-\theta^T x}} \tag{3.1}$$

Η έξοδος της υπόθεσης $h(x)$ ισούται με την πιθανότητα ένα δείγμα του συνόλου δεδομένων να ανήκει σε μία συγκεκριμένη κλάση. Δίνεται, επίσης, ένα κατώφλι, προκειμένου να επιλεχθεί η κλάση στην οποία ανήκει το κάθε δείγμα. Πιο συγκεκριμένα:

$$h'(x) = \begin{cases} 1, & h(x) \geq 0.5 \Rightarrow \theta^T x \geq 0 \\ 0, & h(x) < 0.5 \Rightarrow \theta^T x < 0 \end{cases} \tag{3.2}$$

Οι παράμετροι θ επιλέγονται από τη συνάρτηση κόστους:

$$cost(\theta) = -\sum_x ylog(h(x)) + (1-y)log(1-h(x))) \tag{3.3}$$

3.2.1.1 Τεχνικές Εξομάλυνσης (Regularization Techniques)

Συχνά στο γενικό γραμμικό μοντέλο είναι πιθανόν μία ή περισσότερες ανεξάρτητες μεταβλητές X_j να είναι γραμμικά συσχετισμένες. Η παρουσία του φαινομένου αυτού οδηγεί σε αυξημένα τυπικά σφάλματα και συνεπώς δυσκολεύει την επίδραση της εκτίμησης κάθε επεξηγηματικής μεταβλητής στην εξαρτημένη μεταβλητή Y. Η τεχνική της εξομάλυνσης (regularization) συμβάλλει στην αποφυγή της υπερπροσαρμογής (overfitting). Overfitting έχουμε όταν το μοντέλο είναι πάρα πολύ σύνθετο και ικανό να ταιριάξει τέλεια στα δεδομένα εκπαίδευσης. Το γεγονός αυτό έχει ως αποτέλεσμα το σφάλμα κατά τη φάση της εκπαίδευσης να είναι πολύ μικρό, ενώ κατά τη φάση της ταξινόμησης νέων δεδομένων, που δεν έχει δει πριν, το σφάλμα είναι μεγαλύτερο. Ένα μοντέλο λογιστικής παλινδρόμησης, που χρησιμοποιεί L1 regularization ονομάζεται μοντέλο Lasso Regression, ενώ το μοντέλο, που χρησιμοποιεί L2 regularization, ονομάζεται μοντέλο Ridge Regression.

- **L2 Regularization:** Κατά τη μέθοδο αυτή προστίθεται στη συνάρτηση κόστους ένας επιπλέον όρος, ο όρος ποινής (penalty term), ο οποίος είναι ίσος με:

  $$\alpha \sum_{j=1}^{p} w_j^2 \tag{3.4}$$

  Κατά τη μέθοδο αυτή, τα βάρη w_j τείνουν στο 0, χωρίς να γίνουν απαραίτητα 0. Μειώνεται, δηλαδή, η επίδραση των άσχετων χαρακτηριστικών στο μοντέλο εκπαίδευσης.

- **L1 Regularization:** Κατά τη μέθοδο αυτή προστίθεται στη συνάρτηση κόστους ένας επιπλέον όρος, ο όρος ποινής (penalty term), ο οποίος είναι ίσος με:

  $$\alpha \sum_{j=1}^{p} |w_j| \tag{3.5}$$

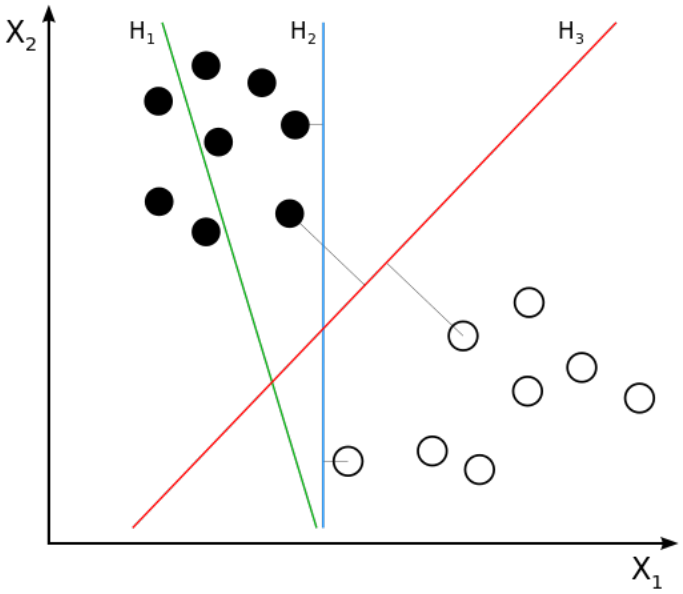
  Κατά τη μέθοδο αυτή, τα περισσότερα βάρη w_j είναι ίσα 0, δημιουργώντας έτσι έναν αραιό (sparse) πίνακα βαρών. Επομένως, με την τεχνική αυτή αμελούνται τα λιγότερο σημαντικά features.

- Μία άλλη μέθοδος για την αποφυγή του overfitting είναι η παλινδρόμηση **Elastic Net**, η οποία συνδυάζει γραμμικά τις ποινές που επιβάλλουν οι δύο προηγούμενες μέθοδοι.

3.2.2 Μηχανές Διανυσμάτων Υποστήριξης (Support Vector Machines)

Οι μηχανές διανυσμάτων υποστήριξης (SVM) [53] ανήκουν στους αλγορίθμους επιβλεπόμενης μάθησης και στηρίζονται στη γραφική αναπαράσταση των διαφόρων στοιχείων και τον διαχωρισμό αυτών ανάλογα με την κλάση, στην οποία ανήκουν. Η λειτουργία του SVM βασίζεται στην κατασκευή ενός υπερεπιπέδου διαχωρισμού, το οποίο βρίσκεται στη μέγιστη δυνατή απόσταση από τα κοντινότερα σημεία εκπαίδευσης και των δύο κλάσεων.

Για παράδειγμα, στο παρακάτω σχήμα παρατηρούμε ότι η ευθεία H_1 δεν είναι δυνατόν να διαχωρίσει τα δεδομένα των δύο κλάσεων. Αντίθετα, ενώ οι ευθείες H_2 & H_3 καταφέρνουν να διαχωρίσουν τα δεδομένα, η H_3 αποδεικνύεται καλύτερη, αφού μεγιστοποιεί την απόσταση από τα δεδομένα διαφορετικών κλάσεων.



Εικόνα 3.2: Διαχωρισμός κλάσεων

Επιλέγουμε, λοιπόν, ως βέλτιστο, το υπερεπίπεδο εκείνο, που μεγιστοποιεί την απόσταση από τα κοντινότερα προς αυτό δεδομένα διαφορετικών κλάσεων. Το υπερεπίπεδο αυτό είναι γνωστό ως υπερεπίπεδο μέγιστου περιθωρίου (maximum-margin hyperplane). Τα κοντινότερα δεδομένα των δύο κλάσεων ως προς το υπερεπίπεδο καλούνται διανύσματα υποστήριξης (support vectors).

3.2.2.1 Γραμμική Μηχανή Διανυσμάτων Υποστήριξης

Θεωρούμε δείγμα εκπαίδευσης $(\overrightarrow{x_1}, y_1), ...(\overrightarrow{x_n}, y_n)$, όπου το y_i δηλώνει την κλάση, που ανήκει το δεδομένο και παίρνει τιμές 1 & -1. Αναζητούμε το υπερεπίπεδο μέγιστου περιθωρίου, έτσι ώστε η απόσταση του κοντινότερου δείγματος εκπαίδευσης κάθε κλάσης από το υπερεπίπεδο να μεγιστοποιείται. Γράφουμε, λοιπόν, το υπερεπίπεδο στη μορφή:

$$\vec{w} \cdot \vec{x} - b = 0 \tag{3.6}$$

όπου $\vec{w}$ το κανονικό διάνυσμα υπερεπιπέδου και b η πόλωση. Στη συνέχεια, επιλέγουμε τα δύο παράλληλα υπερεπίπεδα, που διαχωρίζουν τα δεδομένα των δύο κλάσεων με τη μέγιστη απόσταση μεταξύ τους, όπως ορίζονται στις παρακάτω εξισώσεις:

$$\vec{w} \cdot \vec{x} - b = 1 \quad \text{if} \quad x \in Class1(y_i = 1) \tag{3.7}$$

$$\vec{w} \cdot \vec{x} - b = -1 \quad \text{if} \quad x \in Class2(y_i = -1) \tag{3.8}$$

Η απόσταση των δύο αυτών υπερεπιπέδων καλείται περιθώριο (margin) και είναι ίση με $\frac{2}{\|\vec{w}\|}$. Προκειμένου, λοιπόν, να μεγιστοποιηθεί το περιθώριο διαχωρισμού, απαιτείται η ελαχιστοποίηση της Ευκλείδειας νόρμας του διανύσματος βαρών $\vec{w}$.

Ορίζουμε, επίσης, τις παρακάτω δύο εξισώσεις, προκειμένου τα δεδομένα των δύο κλάσεων να βρίσκονται εκτός του περιθωρίου διαχωρισμού.

$$\vec{w} \cdot \vec{x} - b \geq 1 \quad \text{if} \quad y_i = 1 \tag{3.9}$$

$$\vec{w} \cdot \vec{x} - b \leq -1 \quad \text{if} \quad y_i = -1 \tag{3.10}$$

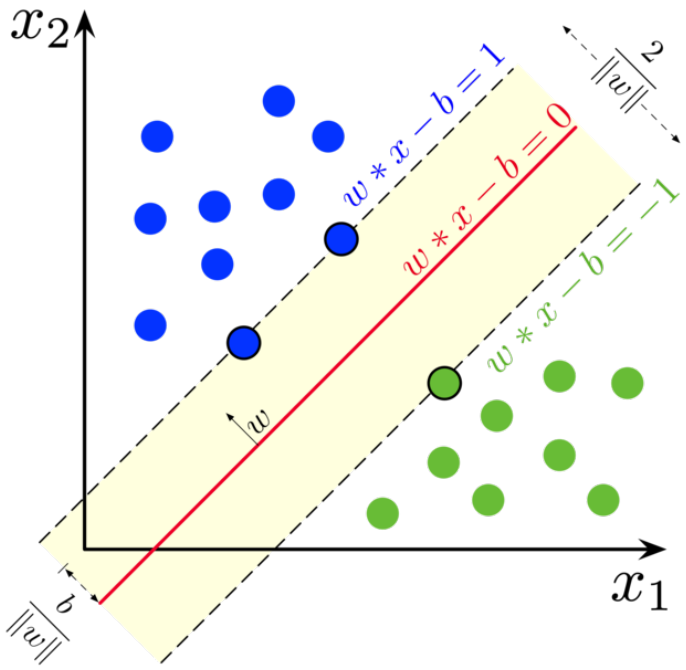
Οι παραπάνω εξισώσεις μπορούν να γραφούν ως εξής:

$$y_i(\vec{w} \cdot \vec{x} - b) \geq 1 \quad \text{for} \quad i = 1, 2, ..., n \tag{3.11}$$

Καταλήγουμε, επομένως στο πρόβλημα βελτιστοποίησης του αλγορίθμου ταξινόμησης SVM.

$$\text{minimize} \quad \|\vec{w}\| \quad \text{subject to} \quad y_i(\vec{w} \cdot \vec{x} - b) \geq 1 \quad \text{for} \quad i = 1, 2, ..., n \tag{3.12}$$

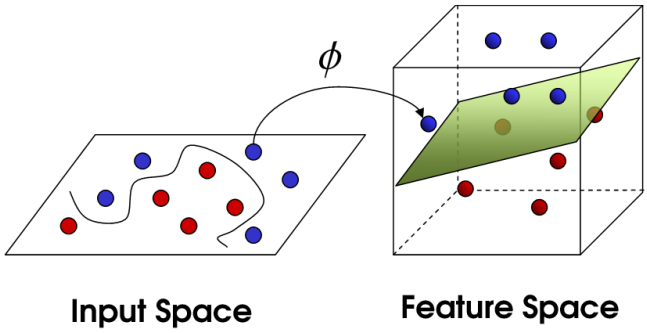
Στο παρακάτω σχήμα φαίνονται τα διανύσματα υποστήριξης, το περιθώριο διαχωρισμού, τα υπερεπίπεδα που ικανοποιούν τις εξισώσεις (3.8) & (3.9), το υπερεπίπεδο μέγιστου περιθωρίου καθώς και το διάνυσμα $\vec{w}$, που είναι κάθετο σ' αυτό.



Εικόνα 3.3: Μηχανή Διανυσμάτων Υποστήριξης

3.2.2.2 Μη Γραμμικά Διαχωρίσιμες Κλάσεις

Σε περιπτώσεις, όπου τα δεδομένα εκπαίδευσης δεν είναι δυνατόν να διαχωριστούν γραμμικά, υφίστανται κάποιους μετασχηματισμούς , οι οποίοι αποκαλούνται πυρήνες, όπως φαίνεται και στην παρακάτω εικόνα. Συγκεκριμένα, την εικόνα παρατηρούμε ότι τα γραμμικά μη διαχωρίσιμα στις 2 διαστάσεις δεδομένα μπορεί να είναι γραμμικά διαχωρίσιμα σε περισσότερες διαστάσεις (3 στην εικόνα), με χρήση των kernel tricks.



Εικόνα 3.4: Kernel trick

Ορισμένοι από τους πιο γνωστούς πυρήνες είναι οι:

- Ο πολυωνυμικός ομοιογενής
- Ο πολυωνυμικός ανομοιογενής
- Ο πυρήνας Gauss ακτινικής βάσης
- Ο πυρήνας υπερβολικής εφαπτομένης

3.2.3 Naive Bayes

Στο πεδίο της μηχανικής μάθησης, οι ταξινομητές Naive Bayes είναι βασισμένοι στο θεώρημα του Bayes, υποθέτοντας ανεξαρτησία μεταξύ όλων των χαρακτηριστικών (naive). Έστω διάνυσμα χαρακτηριστικών (features) $X = (x_1, x_2, .., x_n)$. Η πιθανότητα το διάνυσμα X να ανήκει σε μία από τις κλάσεις C_i, όπου $1 \leq i \leq k$, είναι ίση με:

$$p(C_i|X) = \frac{p(C_i)p(X|C_i)}{p(X)} \tag{3.13}$$

Γνωρίζουμε, όμως, ότι:

$$p(C_i)p(X|C_i) = p(C_i, x_1, ..., x_n) \tag{3.14}$$

Άρα, μπορούμε να γράψουμε:

$$\begin{aligned} p(C_i, x_1, ..., x_n) &= p(x_1, ..., x_n, C_i) = p(x_1|x_2, ..., x_n, C_i)p(x_2, ..., x_n, C_k) \\ &= p(x_1|x_2, ..., x_n, C_i)p(x_2|x_3, ..., x_n, C_i)p(x_3, ..., x_n, C_k) \\ &= ... \\ &= p(x_1|x_2, ..., x_n, C_i)p(x_2|x_3, ..., x_n, C_i)...p(x_{n-1}|x_n, C_i)p(x_n|C_i)p(C_i) \end{aligned} \tag{3.15}$$

Θεωρώντας ότι όλα τα features είναι ανεξάρτητα μεταξύ τους, έχουμε ότι:

$$p(x_i|x_{i+1}, ..., x_n, C_i) = p(x_i|C_i) \tag{3.16}$$

Επομένως:

$$p(C_i|x_1, \ldots, x_n) = p(C_i)p(x_1|C_i)p(x_2|C_i)p(x_3|C_i)\ldots = p(C_i)\prod_{j=1}^{n} p(x_j|C_i) \tag{3.17}$$

Συνεπώς, η απόφαση του Naive Bayes για το άγνωστο δείγμα X είναι:

$$y = \underset{i \in 1,2,...,k}{\operatorname{argmax}}\, p(C_i)\prod_{j=1}^{n} p(x_j|C_i) \tag{3.18}$$

Γενικά, ο ταξινομητής Naive Bayes υπολογίζει για το διάνυσμα των χαρακτηριστικών X τις πιθανότητες να ανήκει σε καθεμιά από τις κλάσεις και το ταξινομεί τελικά στην κλάση εκείνη για την οποία η πιθανότητα είναι μεγαλύτερη.

3.2.3.1 Gaussian Naive Bayes

Στην περίπτωση αυτή, θεωρούμε ότι τα συνεχή δεδομένα ακολουθούν κανονική κατανομή. Για κάθε χαρακτηριστικό x (με μέση τιμή μ_k & διακύμανση σ_k^2) και κλάση C_k, η συνάρτηση πυκνότητας πιθανότητας θεωρείται Gaussian, δηλαδή:

$$p(x = v|C_k) = \frac{1}{\sqrt{2\pi\sigma_k^2}} e^{-\frac{(v-\mu_k)^2}{2\sigma_k^2}} \tag{3.19}$$

3.2.3.2 Multinomial Naive Bayes

Οι multinomial ταξινομητές Bayes είναι ιδιαίτερα δημοφιλείς σε προβλήματα ταξινόμησης κειμένου. Στην προσέγγιση αυτή, τα διανύσματα $x = [x_1, x_2, \ldots, x_n]$ είναι ιστογράμματα, όπου κάθε χαρακτηριστικό εκφράζει τη συχνότητα εμφάνισης ενός ενδεχομένου (Bag of Words). Η πιθανότητα παρατήρησης ενός ιστογράμματος x δίνεται από την ακόλουθη σχέση:

$$p(x|C_k) = \frac{(\sum_i x_i)!}{\prod_i x_i!} \prod_i p_{ki}^{x_i} \tag{3.20}$$

3.2.3.3 Bernoulli Naive Bayes

Όπως η προηγούμενη κατηγορία ταξινομητών, έτσι και οι Bernoulli είναι δημοφιλείς για προβλήματα ταξινόμησης κειμένων. Κάθε χαρακτηριστικό x_i μπορεί να πάρει τιμή 0, εάν ο i^th όρος απουσιάζει από το λεξικό, ή να πάρει την τιμή 1, εάν ο i^{th} όρος βρίσκεται στο λεξικό. Τότε, η συνάρτηση πυκνότητας πιθανότητας είναι η εξής:

$$p(x|C_k) = \prod_{i=1}^{n} p_{ki}^{x_i}(1 - p_{ki})^{1-x_i} \tag{3.21}$$

όπου p_{ki} είναι η πιθανότητα η κλάση C_k να παράξει το ενδεχόμενο i.

3.2.4 Ο Αλγόριθμος k-Nearest Neighbours

Ανήκει στους αλγορίθμους επιβλεπόμενης μάθησης και μπορεί να χρησιμοποιηθεί τόσο σε προβλήματα ταξινόμησης όσο και σε προβλήματα παλινδρόμησης. Ένα σημείο του χώρου των χαρακτηριστικών ταξινομείται στην κλάση, που είναι η πιο κοινή μεταξύ των k πλησιέστερων γειτόνων, οι οποίοι μετρώνται και βρίσκονται από μία συνάρτηση απόστασης. Βρίσκει εφαρμογή σε προβλήματα αναγνώρισης προτύπων και εξόρυξης δεδομένων.

Η παράμετρος k καθορίζεται από τον χρήστη και επιλέγεται μέσω πειραμάτων, καθώς η καλύτερη επιλογή του k εξαρτάται από τα δεδομένα. Γενικότερα, μεγάλες τιμές του k τείνουν να μειώσουν τον θόρυβο στην ταξινόμηση, αλλά κάνουν τα περιθώρια μεταξύ των κλάσεων λιγότερο διακριτά. Σε δυαδικά προβλήματα ταξινόμησης, όπου έχουμε να προβλέψουμε δύο κλάσεις, είναι προτιμότερο να επιλέγουμε το k να είναι ένας μονός αριθμός, προκειμένου να αποφευχθεί η ισοπαλία στην ψήφο της κλάσης.

Σε περίπτωση, όμως, που υπάρχει κλάση με μεγάλο αριθμό δεδομένων, αυτό θα επηρεάσει την κλάση του νέου δεδομένου που πρόκειται να ταξινομηθεί. Το πρόβλημα αυτό μπορεί να επιλυθεί με την απόδοση βαρών στους k πλησιέστερους γείτονες, λαμβάνοντας υπόψη την απόσταση μεταξύ του νέου δεδομένου ως προς καθέναν από τους k πλησιέστερους γείτονες. Το βάρος αυτό είναι ανάλογο του $\frac{1}{d}$, όπου d η απόσταση.

Η συνηθέστερη μετρική απόστασης για συνεχείς μεταβλητές είναι η Ευκλείδια απόσταση (Euclidean distance), που ορίζεται από τον παρακάτω τύπο:

$$d(x_i, y_i) = \sqrt{\sum_{i=1}^{k}(x_i - y_i)^2} \tag{3.22}$$

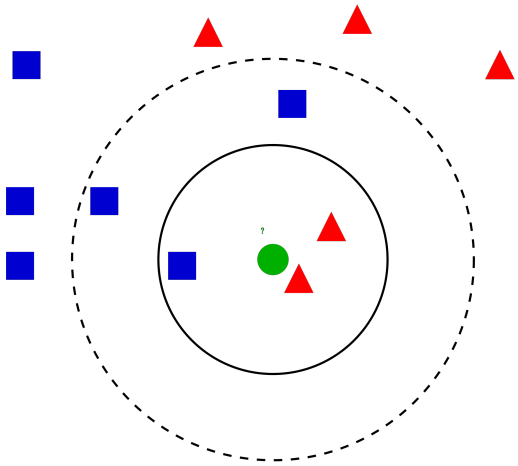
Αντίθετα, για διακριτές τιμές, όπως σε περιπτώσεις ταξινόμησης κειμένου, χρησιμοποιείται η απόσταση Hamming. Υπολογίζεται από την ακόλουθη εξίσωση:

$$D_H(x_i, y_i) = \sum_{i=1}^{k} |x_i - y_i| \tag{3.23}$$

όπου:

$$\begin{cases} x_i = y_i \Rightarrow D_H = 0 \\ x_i \neq y_i \Rightarrow D_H = 1 \end{cases} \tag{3.24}$$

Στην παρακάτω εικόνα δίνεται ένα παράδειγμα προσδιορισμού κατηγορίας με βάση τους 3 & 5 κοντινότερους γείτονες. Εάν k=3, η νέα περίπτωση, που παριστάνεται από το πράσινο κυκλάκι, ταξινομείται στα κόκκινα τρίγωνα, ενώ αν k=5, ταξινομείται στα μπλε τετράγωνα.



Εικόνα 3.5: Παράδειγμα προσδιορισμού κατηγορίας με βάση τους 3 & 5 κοντινότερους γείτονες

3.2.5 Boosting

Πρόκειται για μία μέθοδο μετατροπής ενός συνόλου αδύναμων αλγορίθμων μάθησης σε ισχυρούς. Η βασική ιδέα του Boosting είναι ο συνδυασμός των αποτελεσμάτων πολλών «αδύναμων» αλγορίθμων μάθησης. Δηλαδή ο συνδυασμός αλγορίθμων των οποίων το σφάλμα (error rate) είναι ελαφρώς καλύτερο από την τυχαία επιλογή. Με την τεχνική του boosting μειώνεται τόσο η διακύμανση(variance) όσο και η μεροληψία(bias). Στη συνέχεια, παρουσιάζονται τρία διαφορετικά είδη αλγορίθμων με τη μέθοδο Boosting.

- **AdaBoost Classifier:** Τροποποιεί κάθε φορά τα βάρη των δειγμάτων έτσι ώστε κάθε νέο αδύναμο μοντέλο που εκπαιδεύεται να λαμβάνει σοβαρά υπόψη τα λάθη των προηγούμενων. Τελικά, συνδυάζονται οι αποφάσεις των επιμέρους μοντέλων ανάλογα με τις επιδόσεις τους. Χρησιμοποιεί κατά κανόνα decision stumps. Είναι, ωστόσο, αρκετά ευαίσθητος σε θόρυβο και outliers.

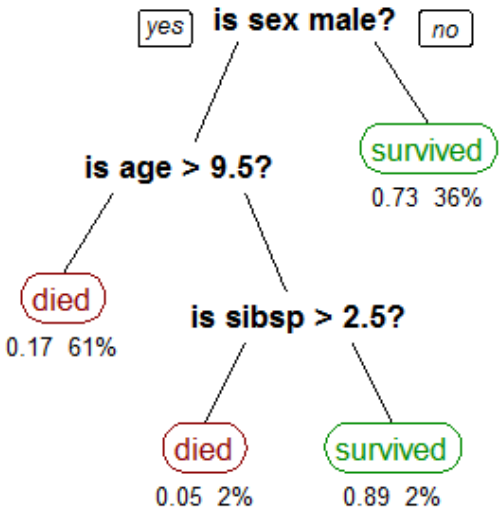
- **Gradient Boosting:** Στη μέθοδο αυτή, αντιμετωπίζουμε το πρόβλημα του boosting ως ένα πρόβλημα ελαχιστοποίησης. Θεωρούμε, αρχικά, έναν αδύναμο αλγόριθμο μάθησης και σε κάθε βήμα προσθέτουμε έναν επιπλέον αδύναμο αλγόριθμο, προκειμένου να αυξήσουμε την επίδοση και να κατασκευάσουμε έτσι έναν ισχυρό αλγόριθμο. Με επαναληπτικό τρόπο, προσθέτουμε ένα μοντέλο κάθε φορά, με σκοπό τη μείωση των σφαλμάτων που έγιναν από τα υπάρχοντα μοντέλα. Τα μοντέλα προστίθενται διαδοχικά μέχρις ότου δεν μπορούν να γίνουν περαιτέρω βελτιώσεις. Ονομάζεται gradient boosting, επειδή χρησιμοποιεί έναν αλγόριθμο gradient descent, για να ελαχιστοποιήσει το κόστος κατά την πρόσθεση νέων μοντέλων.

- **Xgboost Classifier [54]:** Αποτελεί την state-of-the-art τροποποίηση του gradient boosting. Είναι κατάλληλο για μεγάλα datasets σε καλές ταχύτητες.

3.2.6 Δέντρα Απόφασης (Decision Trees)

Τα δέντρα απόφασης έχουν χρησιμοποιηθεί ευρέως στο πεδίο της Μηχανικής Μάθησης τόσο σε προβλήματα ταξινόμησης (classification) όσο και σε προβλήματα παλινδρόμησης (regression).

Τα μοντέλα δέντρων, όπου η μεταβλητή-στόχος μπορεί να πάρει ένα πεπερασμένο σύνολο τιμών ονομάζονται δέντρα ταξινόμησης. Ο αλγόριθμος των Δέντρων Απόφασης οδηγεί στη δημιουργία μιας δενδροειδούς μορφής, όπου κάθε κόμβος αναπαριστά την απόφαση, που πρέπει να παρθεί και κάθε φύλλο αναπαριστά την ετικέτα των κλάσεων (label). Έχοντας σχηματίσει το δέντρο, μπορούμε να προβλέψουμε την έξοδο του δείγματος, εάν το διασχίσουμε και φτάσουμε στα φύλλα του. Για παράδειγμα, το παρακάτω μοντέλο χρησιμοποιεί τρία χαρακτηριστικά (features), προκειμένου να προβλέψει, εάν ο επιβάτης επιβίωσε ή όχι.



Εικόνα 3.6: Δέντρο Απόφασης

Σημειώνεται ότι η σημασία των χαρακτηριστικών (feature importance) είναι ένας αριθμός, που κυμαίνεται μεταξύ 0 & 1. Αν η σημασία των χαρακτηριστικών είναι 0, τότε το χαρακτηριστικό αυτό, δεν χρησιμοποιήθηκε καθόλου στην πρόβλεψη. Αντίθετα, αν η σημασία του είναι ίση με 1, τότε το χαρακτηριστικό αυτό προβλέπει την κλάση ακριβώς. Το άθροισμα της σημασίας όλων των χαρακτηριστικών είναι ίσο με 1. Όσο πιο σημαντικό είναι ένα χαρακτηριστικό στην πρόβλεψη της κλάσης, τόσο πιο πάνω βρίσκεται στο δέντρο.

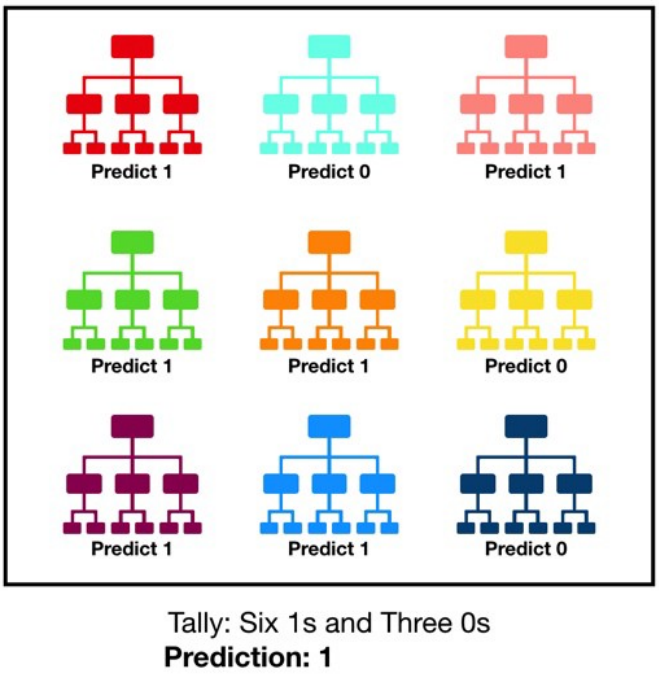
Συνοπτικά, τα Δέντρα Αποφάσεων μπορούν να παράξουν κατανοητούς κανόνες, να υποδείξουν τα πιο σημαντικά features στην πρόβλεψη της κλάσης αλλά και να χειριστούν τόσο αριθμητικά όσο και categorical δεδομένα. Επίσης, απαιτούν ελάχιστη προεπεξεργασία των δεδομένων από τους χρήστες. Ωστόσο, είναι δυνατόν να δημιουργηθούν πολύπλοκα δέντρα, οδηγώντας στο φαινόμενο της υπερπροσαρμογής (overfitting). Επίσης, είναι ιδιαίτερα ασταθή, καθώς μικρές αλλαγές στα δεδομένα ενδέχεται να οδηγήσουν στην παραγωγή ενός εντελώς διαφορετικού δέντρου. Το πρόβλημα αυτό αντιμετωπίζεται με μεθόδους, όπως η τεχνική bagging που θα αναλυθεί στη συνέχεια. Τέλος, οι αλγόριθμοι Δέντρων Απόφασης χρειάζονται έναν αρκετά μεγάλο όγκο δεδομένων για την εκπαίδευσή τους, με αποτέλεσμα να καταναλώνουν αρκετούς υπολογιστικούς πόρους, έχοντας μεγάλο κόστος σε χώρο και χρόνο.Μερικοί από τους πιο γνωστούς αλγορίθμους δέντρων απόφασης είναι οι παρακάτω:

- CART (Classification And Regression Tree - Δέντρο Ταξινόμησης και Παρεμβολής)
- ID3 (Iterative Dichotomiser 3 - Επαναληπτική Διχοτόμηση)
- MARS
- C4.5

3.2.7 Τυχαία Δάση (Random Forests)

Στη στατιστική και τη μηχανική μάθηση προτείνεται η χρήση πολλαπλών αλγορίθμων μάθησης και ο συνδυασμός τους στο τέλος, προκειμένου να επιτευχθεί καλύτερη επίδοση στην πρόβλεψη (Ensemble Methods).

Τα Τυχαία Δάση (Random Forests) [55] αποτελούνται από έναν μεγάλο αριθμό Δέντρων Απόφασης και ανήκουν στην οικογένεια των ensemble methods. Σε προβλήματα ταξινόμησης κάθε Δέντρο Απόφασης, ξεχωριστά, προβλέπει την κλάση στην οποία ανήκει το συγκεκριμένο δεδομένο εκπαίδευσης και στο τέλος η κλάση, που προβλέφθηκε τις περισσότερες φορές, αποτελεί την κλάση της τελικής μας πρόβλεψης. Αυτό φαίνεται χαρακτηριστικά στο παρακάτω σχήμα:



Εικόνα 3.7: Πρόβλεψη της κλάσης με τον αλγόριθμο των Τυχαίων Δασών

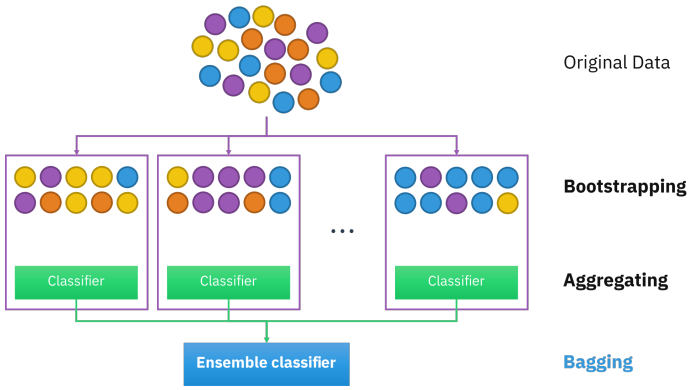
Σε προβλήματα παλινδρόμησης, ως έξοδος δίνεται η τάξη με τη μέση πρόβλεψη των μεμονωμένων δέντρων.

Προκειμένου ο αλγόριθμος των Τυχαίων Δασών να αποδώσει καλά, απαιτείται οι προβλέψεις και κατά συνέπεια τα σφάλματα των μεμονωμένων Δέντρων Απόφασης να έχουν χαμηλή συσχέτιση μεταξύ τους.

Για να διασφαλιστεί ότι η συμπεριφορά του κάθε Δέντρου Απόφασης είναι ασυσχέτιστη ως προς τα υπόλοιπα, τα Τυχαία Δάση χρησιμοποιούν την τεχνική bagging (Bootstrap Aggregating).

3.2.7.1 Bagging (Bootstrap Aggregating)

Η τεχνική Bagging συμβάλλει στην αποφυγή του overfitting και στη μείωση της διακύμανσης. Συγκεκριμένα, θεωρώντας ένα σύνολο εκπαίδευσης D μεγέθους n, η τεχνική αυτή παράγει με χρήση ομοιόμορφης δειγματοληψίας με αντικατάσταση m καινούρια σύνολα εκπαίδευσης D_i μεγέθους n'. Επιλέγοντας δεδομένα με αντικατάσταση, καταλήγουμε με επαναλαμβανόμενα δεδομένα για κάθε νέο σύνολο εκπαίδευσης D_i. Επομένως, κάθε δέντρο απόφασης εκπαιδεύεται σε διαφορετικά σύνολα δεδομένων. Στο παρακάτω σχήμα παρουσιάζεται η ιδέα της τεχνικής Bagging.



Εικόνα 3.8: Bagging

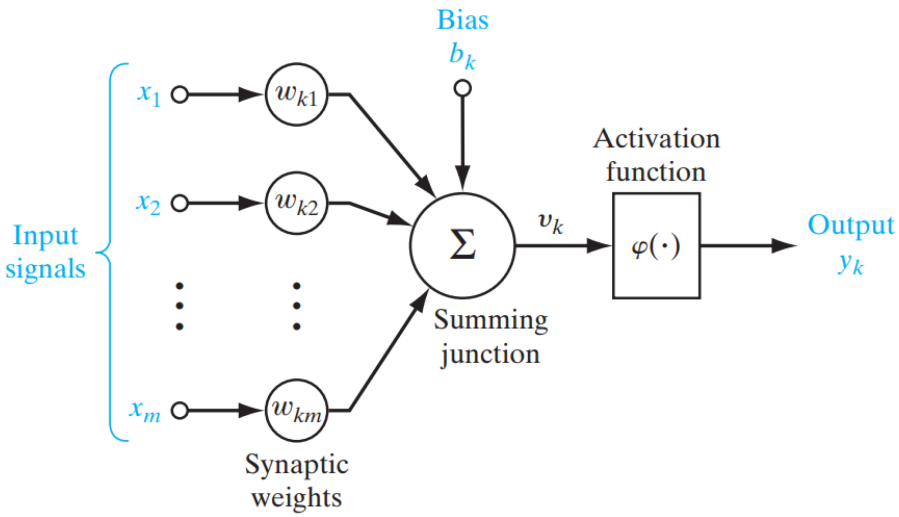
3.3 Τεχνητά Νευρωνικά Δίκτυα (Artificial Neural Networks)

Τα τεχνητά νευρωνικά δίκτυα είναι υπολογιστικά συστήματα εμπνευσμένα από τα βιολογικά νευρωνικά δίκτυα, που αποτελούν τους ανθρώπινους εγκεφάλους. Έχουν χρησιμοποιηθεί με επιτυχία σε διάφορους τομείς, όπως στην αναγνώριση της ομιλίας, στην επεξεργασία εικόνας, σε ιατρικές εφαρμογές, στον προσαρμοστικό έλεγχο κ.ά.

3.3.1 Single Layer Perceptron (Το Perceptron του Rosenblatt)

Στο παρακάτω σχήμα φαίνεται το μοντέλο ενός τεχνητού νευρώνα [56], που τροφοδοτείται από μία είσοδο X. Κάθε επιμέρους χαρακτηριστικό της εισόδου πολλαπλασιάζεται με ένα αριθμητικό βάρος W_{ki} ,$1 \leq i \leq m$. Το βάρος W_{ki} δηλώνει πόσο σημαντικό είναι το feature x_i. Αφού πολλαπλασιαστούν οι είσοδοι με τα αντίστοιχα βάρη, πηγαίνουν σε έναν αθροιστή, όπου και αθροίζονται μαζί με μία αριθμητική τιμή πόλωσης b_k. Το αποτέλεσμα v_k περνά από μία συνάρτηση ενεργοποίησης φ, η έξοδος της οποίας αποτελεί την τελική έξοδο. Συνοπτικά μπορούμε να γράψουμε:

$$y_k = \varphi(\sum_{i=1}^{m} x_i w_{ki} \ + b_k) \tag{3.25}$$



Εικόνα 3.9: Single Layer Perceptron

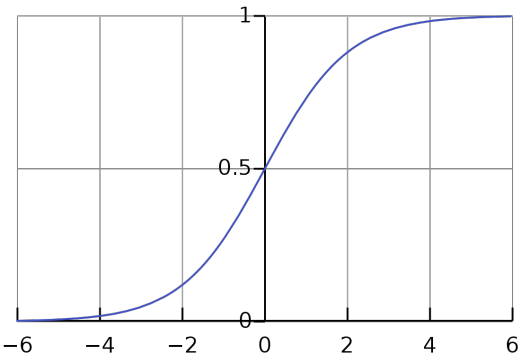
3.3.2 Συναρτήσεις Ενεργοποίησης (Activation Functions)

Μερικές από τις πιο γνωστές συναρτήσεις ενεργοποίησης είναι οι εξής:

- **Σιγμοειδής (sigmoid):** Αντιστοιχίζει όλες τις τιμές στο διάστημα (0,1). Δίνεται από τον τύπο:

$$f(x) = \sigma(x) = \frac{1}{1 + e^{-x}} \tag{3.26}$$

Η γραφική παράσταση της συνάρτησης δίνεται στο σχήμα που ακολουθεί:

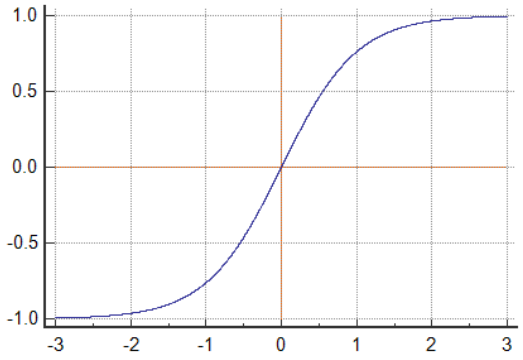


Εικόνα 3.10: Σιγμοειδής Συνάρτηση Ενεργοποίησης

- **Συνάρτηση Υπερβολικής Εφαπτομένης (tanh):** Αντιστοιχίζει όλες τις τιμές στο διάστημα (-1,1). Δίνεται από τον τύπο:

$$f(x) = tanh = \frac{(e^{x} - e^{-x})}{e^{x} + e^{-x}} \tag{3.27}$$

Η γραφική παράσταση της συνάρτησης δίνεται στο σχήμα, που ακολουθεί:

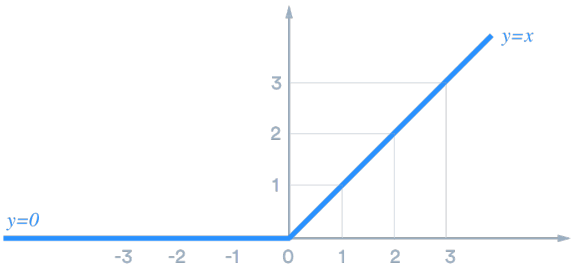


Εικόνα 3.11: Υπερβολική Εφαπτομένη Συνάρτηση Ενεργοποίησης

- **Rectified Linear Unit (ReLU):** Αποτελεί την πιο διαδεδομένη συνάρτηση ενεργοποίησης. Χρησιμοποιείται σε αλγορίθμους βαθιάς μάθησης (deep learning) και σε συνελικτικά νευρωνικά δίκτυα. Αντιστοιχίζει όλες τιμές στο διάστημα $[0, \infty)$. Δίνεται από τον παρακάτω τύπο:

$$f(x) = \begin{cases} 0, & \text{if} \quad x \leq 0 \\ x, & \text{if} \quad x > 0 \end{cases} \tag{3.28}$$

Η γραφική παράσταση της συνάρτησης δίνεται από το ακόλουθο σχήμα.



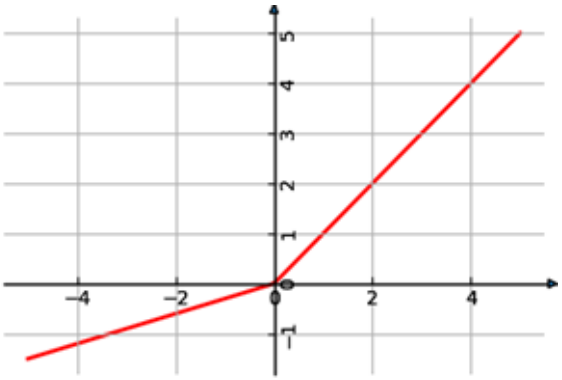
Εικόνα 3.12: ReLU Συνάρτηση Ενεργοποίησης

Επομένως, κάθε αρνητική τιμή που περνά από μία ReLU συνάρτηση ενεργοποίησης μετατρέπεται σε μηδέν. Έτσι, εάν η είσοδος είναι πάντα μικρότερη του μηδενός, τότε θα έχουμε ότι $f(x) = 0$, και επειδή η παράγωγος θα είναι επίσης μηδέν, τα βάρη δεν θα ανανεώνονται, προκαλώντας με αυτόν τον τρόπο τη θανάτωση του νευρώνα (dying ReLU problem).

- **Leaky ReLU:** Αποτελεί μία προσπάθεια να επιλυθεί το πρόβλημα του dying ReLU. Αντιστοιχίζει όλες τις τιμές στο διάστημα $(-\infty, +\infty)$. Οι αρνητικές τιμές πολλαπλασιάζονται με μία μικρή σταθερά c=0.01, αντί να μηδενίζονται, όπως φαίνεται στον παρακάτω τύπο.

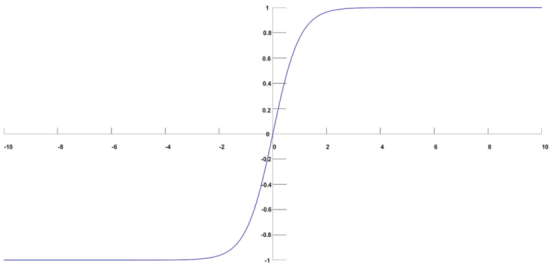
$$f(x) = \begin{cases} 0.01x, & \text{if} \quad x < 0 \\ x, & \text{if} \quad x \geq 0 \end{cases} \tag{3.29}$$

Η γραφική παράσταση της συνάρτησης δίνεται από το ακόλουθο σχήμα:



Εικόνα 3.13: Leaky ReLU Συνάρτηση Ενεργοποίησης

- **Softmax:** Χρησιμοποιείται σε προβλήματα ταξινόμησης τόσο με δύο όσο και με περισσότερες κλάσεις. Δέχεται στην είσοδο αριθμούς, τους οποίους μετατρέπει σε πιθανότητες, το άθροισμα των οποίων είναι ίσο με τη μονάδα.



Εικόνα 3.14: Softmax activation function

$$Softmax(z_j) = \frac{e^{z_j}}{\sum_{j=0}^{N} e^{z_j}} \tag{3.30}$$

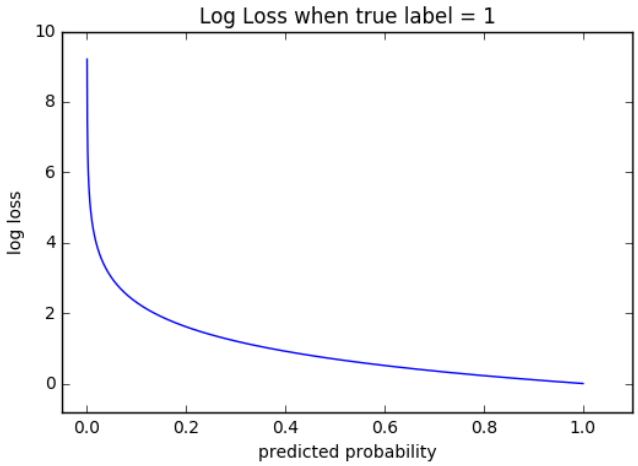
3.3.3 Συνάρτηση Κόστους - Αντικειμενική Συνάρτηση (Loss Function)

Η συνάρτηση κόστους συμβάλλει στη βελτιστοποίηση των παραμέτρων των νευρωνικών δικτύων. Σκοπός μας είναι να ελαχιστοποιήσουμε το κόστος σε ένα νευρωνικό δίκτυο, βελτιστοποιώντας με αυτόν τον τρόπο τα βάρη του. Το κόστος υπολογίζεται χρησιμοποιώντας μία συνάρτηση κόστους (Loss Function), η οποία συγκρίνει την πραγματική (real) με την προβλεπόμενη (predicted) τιμή. Όπως θα δούμε στη συνέχεια, χρησιμοποιούμε τη μέθοδο gradient descent, προκειμένου να ανανεώσουμε τα βάρη στο νευρωνικό δίκτυο και κατ' επέκταση να ελαχιστοποιήσουμε το κόστος. Μερικές από τις πιο γνωστές συναρτήσεις κόστους είναι οι ακόλουθες:

- **Μέσο Τετραγωνικό Σφάλμα (Mean Squared Error):** Σε προβλήματα παλινδρόμησης (regression), χρησιμοποιείται ως συνάρτηση κόστους κυρίως το μέσο τετραγωνικό σφάλμα. Υπολογίζεται παίρνοντας τον μέσο όρο των τετραγώνων των διαφορών της πραγματικής από την προβλεπόμενη τιμή. Δίνεται από τον ακόλουθο τύπο:

$$J = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{3.31}$$

- **Cross Entropy loss & Logistic Regression:** Υπολογίζει την επίδοση ενός μοντέλου ταξινόμησης (classification model), του οποίου η έξοδος είναι μία τιμή πιθανότητας μεταξύ του 0 και 1. Το cross - entropy loss αυξάνεται, καθώς η πιθανότητα της πρόβλεψης αποκλίνει από την πραγματική τιμή. Ένα ιδανικό μοντέλο ταξινόμησης θα έχει cross - entropy loss ίσο με 0. Αυτό φαίνεται στο παρακάτω σχήμα.



Εικόνα 3.15: Cross entropy loss

Παρατηρούμε ότι όταν η πραγματική τιμή είναι 1 και η προβλεπόμενη πιθανότητα είναι 1, τότε το κόστος είναι ίσο με το 0. Αντίθετα, όσο περισσότερο αποκλίνει η πιθανότητα της πρόβλεψης από την πραγματική τιμή, τόσο μεγαλύτερο γίνεται το κόστος.

Η συνάρτηση κόστους δίνεται από τον ακόλουθο τύπο:

$$Loss = \frac{1}{m}[\sum_{i=1}^{m} -y^{(i)}log(h_\theta(x^{(i)})) + (1 - y^{(i)})log(1 - h_\theta(x^{(i)}))] \tag{3.32}$$

όπου m είναι ο αριθμός των δεδομένων.

Ορίζεται και ως εξής:

$$Cost(h_\theta(x), y) = \begin{cases} -log(h_\theta(x)), & y = 1 \\ -log(1 - h_\theta(x)), & y = 0 \end{cases} \tag{3.33}$$

Ενδεικτικά, παραθέτουμε και κάποιες άλλες συναρτήσεις κόστους:

- **Mean Absolute Error**

$$MAE = \frac{\sum_{i=1}^{n} |y_i - \hat{y}_i|}{n} \tag{3.34}$$

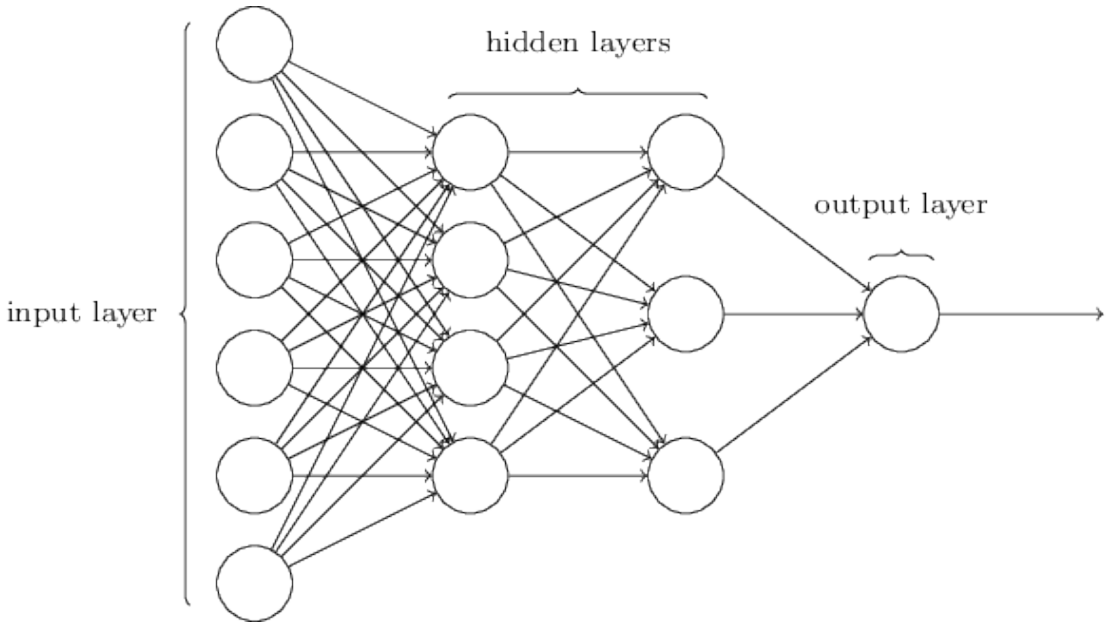
- **Hinge Loss/SVM Loss**

$$HingeLoss = \sum_{j \neq y_i} \max(0, s_j - s_{yi} + 1) \tag{3.35}$$

- **KL-Divergence**
- **Huber Loss**

3.3.4 Το Perceptron Πολλών Επιπέδων (Multilayer Perceptron - MLP)

Το Perceptron πολλών επιπέδων έχει τουλάχιστον ένα κρυφό στρώμα (hidden layer) ανάμεσα στο στρώμα εισόδου και το στρώμα εξόδου. Αυτό που τα ξεχωρίζει από το επίπεδο εξόδου, είναι ότι η έξοδος των νευρώνων των κρυφών επιπέδων αποτελεί πάντα είσοδο για τους νευρώνες των επόμενων επιπέδων (κρυφά ή επίπεδο εξόδου). Κάθε επίπεδο αποτελείται από πολλούς νευρώνες που συνδέονται με τους νευρώνες του επόμενου επιπέδου μέσω των βαρών, όπως είδαμε στην ενότητα 3.3.1.Ο αριθμός των νευρώνων στο στρώμα εξόδου (output layer) είναι ίσος με τον αριθμό των κλάσεων (labels), που επιθυμούμε να προβλέψουμε. Στο παρακάτω σχήμα, φαίνεται ένα Perceptron πολλών επιπέδων. Συγκεκριμένα, έχουμε τέσσερα επίπεδα (στρώμα εισόδου, δύο κρυφά στρώματα, στρώμα εξόδου). Οι αλγόριθμοι βαθιάς μάθησης (deep learning) χρησιμοποιούν περισσότερα του ενός κρυφά επίπεδα.



Εικόνα 3.16: Multilayer Perceptron (MLP)

3.3.5 Ο Αλγόριθμος Backpropagation

Ο αλγόριθμος της οπίσθιας διάδοσης σφαλμάτων (Backpropagation) αποτελεί μία μέθοδο ελαχιστοποίησης της συνάρτησης κόστους ανανεώνοντας τα βάρη του νευρωνικού δικτύου.

Στα πολυεπίπεδα νευρωνικά δίκτυα, όπως και στην περίπτωση του απλού perceptron, η διαδικασία της εκπαίδευσης βασίζεται στην κατάλληλη προσαρμογή των βαρών του δικτύου, προκειμένου η τελική έξοδος να προσεγγίζει όσο τον δυνατόν περισσότερο την αναμενόμενη έξοδο.

Η πιο γνωστή και σε ευρεία χρήση μέθοδος ανανέωσης των βαρών του δικτύου και κατ' επέκταση εύρεσης του ελαχίστου μιας συνάρτησης σφάλματος, είναι η μέθοδος κατάβασης δυναμικού (gradient descent).

$$\theta = \theta_{nom} - \alpha \frac{\partial J}{\partial \theta_{nom}} \tag{3.36}$$

όπου α είναι το learning rate, θ_{nom} η τρέχουσα τιμή του βάρους και ϑ η νέα τιμή του βάρους. Πρόκειται για μία επαναληπτική μέθοδο, με την έννοια ότι ανανεώνουμε τα βάρη του δικτύου πολλαπλές φορές, προκειμένου να ελαχιστοποιήσουμε τη συνάρτηση κόστους.

Η μέθοδος κατάβασης δυναμικού (gradient descent) διακρίνεται στις εξής κατηγορίες [57]: (Σημείωση: Ορίζουμε ως εποχή (epoch) το μέγεθος εκείνο, κατά το οποίο ολόκληρο το dataset επεξεργάζεται από το νευρωνικό δίκτυο μία φορά)

- **Batch Gradient Descent:** Πρόκειται για μία περίπτωση της μεθόδου κατάβασης δυναμικού (gradient descent), κατά την οποία επεξεργάζονται όλα τα δεδομένα στο σύνολο εκπαίδευσης σε κάθε επανάληψη (iteration). Για παράδειγμα, έστω ότι έχουμε μία συνάρτηση γραμμικής παλινδρόμησης (linear regression) και ορίζουμε τη συνάρτηση κόστους ως εξής: $J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(\hat{y}^i - y^i)^2$, όπου m ο αριθμός των δεδομένων στο σύνολο εκπαίδευσης. Τότε:

$$\frac{\partial J(\theta)}{\partial \theta_j} = \frac{1}{m}\sum_{i=1}^{m}(\hat{y}^i - y^i)x_j^i \tag{3.37}$$

  Ανανεώνουμε τα βάρη του νευρωνικού δικτύου ως εξής:

  Repeat(

$$\theta_j := \theta_j - \alpha\frac{1}{m}\sum_{i=1}^{m}(\hat{y}^i - y^i)x_j^i \tag{3.38}$$

  For every j=0,...,n όπου n είναι ο αριθμός των features
)

  Ωστόσο, εάν ο αριθμός των δεδομένων στο σύνολο εκπαίδευσης είναι εξαιρετικά μεγάλος, τότε η μέθοδος αυτή είναι υπολογιστικά πολύ ακριβή και δεν προτιμάται. Έτσι, επιλέγουμε να χρησιμοποιούμε μία από τις ακόλουθες δύο μεθόδους.

- **Stochastic Gradient Descent:** Αυτή η μέθοδος επεξεργάζεται ένα δεδομένο του συνόλου εκπαίδευσης σε κάθε επανάληψη. Έτσι, οι παράμετροι ανανεώνονται μετά από κάθε επανάληψη, κατά την οποία μόνο ένα δεδομένο του συνόλου εκπαίδευσης έχει επεξεργαστεί. Ο αλγόριθμος της μεθόδου αυτής είναι ο ακόλουθος:

  Repeat (

  for i in range(m) (

$$\theta_j := \theta_j - \alpha(\hat{y^i} - y^i)x_j^i \tag{3.39}$$

  For every j=0,...,n όπου n είναι ο αριθμός των features
)
)
  Ωστόσο, εάν ο αριθμός των δεδομένων στο σύνολο εκπαίδευσης είναι μεγάλος, αυτό συνεπάγεται ότι και ο αριθμός των επαναλήψεων θα είναι μεγάλος.

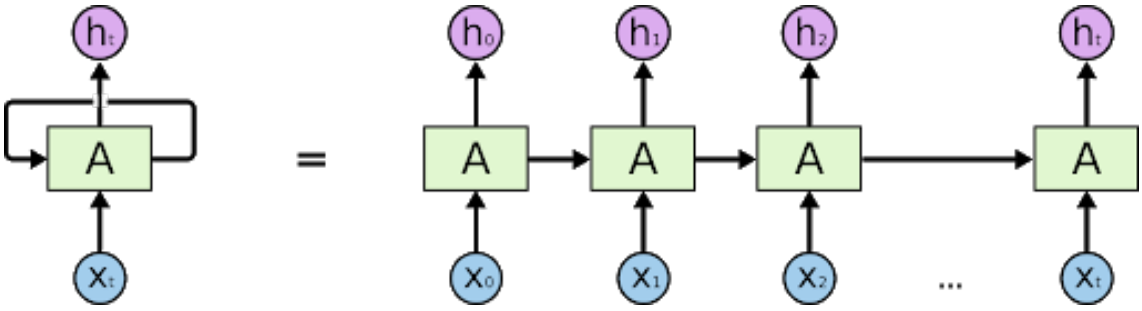
- **Mini Batch Gradient Descent:** Αυτή η μέθοδος είναι γρηγορότερη από τις προηγούμενες δύο και προτιμάται εάν ο αριθμός των δεδομένων είναι πολύ μεγάλος (ιδίως σε προβλήματα βαθιάς μάθησης). Κατά τη μέθοδο αυτή χωρίζουμε τον συνολικό αριθμό των δεδομένων του συνόλου εκπαίδευσης (έστω m training examples) σε n batches, μεγέθους $\frac{m}{n}$ το καθένα. Στην περίπτωση αυτή, ο αριθμός των επαναλήψεων είναι ίσος με τον συνολικό αριθμό των batches. Με τη μέθοδο αυτή, σε κάθε επανάληψη επεξεργαζόμαστε n training examples.

Εκτός από τις μεθόδους αυτές, τα τελευταία χρόνια χρησιμοποιούνται σε όλο και μεγαλύτερο βαθμό τεχνικές αυτόματης βελτιστοποίησης, στις οποίες γίνεται αυτόματη ρύθμιση του ρυθμού μάθησης, όπως Adagrad, Adadelta και Adam [58], η οποία είναι αυτή τη στιγμή η πιο δημοφιλής τεχνική.

3.3.6 Αναδρομικά Νευρωνικά Δίκτυα (Recurrent Neural Networks-RNN)

Τα αναδρομικά νευρωνικά δίκτυα είναι μία ειδική κατηγορία νευρωνικών δικτύων που επεξεργάζονται αποτελεσματικά κάθε είδους ακολουθιακά δεδομένα. Παραδείγματα, αποτελούν η φωνή, η γραφή καθώς και η οπτική πληροφορία, που προκύπτει από μία κίνηση.

Στα αναδρομικά νευρωνικά δίκτυα, η έξοδος του προηγούμενου βήματος τροφοδοτείται ως είσοδος στο τρέχον βήμα. Στα παραδοσιακά νευρωνικά δίκτυα, όλα τα ζευγάρια εισόδων και εξόδων είναι ανεξάρτητα μεταξύ τους. Ωστόσο, σε περιπτώσεις όπου μας ζητείται να προβλέψουμε την επόμενη λέξη σε μία πρόταση, οι προηγούμενες λέξεις απαιτούνται και επομένως είναι απαραίτητο το μοντέλο μας να θυμάται τις λέξεις αυτές. Τα αναδρομικά νευρωνικά δίκτυα καταφέρνουν να λύσουν το πρόβλημα αυτό με τη βοήθεια της εσωτερικής τους κρυφής κατάστασης (hidden state), που κρατάει πληροφορία για την ακολουθία.



Εικόνα 3.17: Recurrent Neural Network (RNN)

Στην αριστερή εικόνα φαίνεται η αναδρομική λειτουργία του RNN. Το RNN δέχεται το ένα μετά το άλλο τα στοιχεία της ακολουθίας και ενημερώνει την εσωτερική ή κρυφή του κατάσταση.

Ένας άλλος τρόπος αναπαράστασης είναι με το "ξετύλιγμα" του δικτύου στο χρόνο, όπως φαίνεται στη δεξιά εικόνα. Στην περίπτωση αυτή, αρχικά λαμβάνεται ως είσοδος το x_0, παράγεται η έξοδος h_0, η οποία με τη σειρά της δίνεται ως είσοδος μαζί με το x_1 στο επόμενο χρονικό βήμα (timestep). Ομοίως, η έξοδος h_1 τροφοδοτείται ως είσοδος με το x_2 στο επόμενο timestep. Σημειώνεται ότι η έξοδος h_t συχνά αναφέρεται και ως κατάσταση του δικτύου. Με αυτόν τον τρόπο, το νευρωνικό δίκτυο θυμάται το περιεχόμενο της ακολουθίας.

Η γενική μορφή των εξισώσεων ενός αναδρομικού δικτύου είναι η εξής:

$$h_t = \varphi(Wx_t + Uh_{t-1} + b) \tag{3.40}$$

όπου:

- h_t η κρυφή αναπαράσταση τη χρονική στιγμή t
- x_t το διάνυσμα του στοιχείου της ακολουθίας τη χρονική στιγμή t
- φ μία μη γραμμική συνάρτηση ενεργοποίησης
- W ο πίνακας παραμέτρων, που επιδρούν πάνω στην είσοδο x_t
- U ο πίνακας παραμέτρων, που επιδρούν πάνω στην έξοδο του δικτύου την προηγούμενη χρονική στιγμή
- b ένα διάνυσμα πόλωσης

Θεωρητικά, τα αναδρομικά νευρωνικά δίκτυα (RNNs) κάνουν χρήση της πληροφορίας σε μεγάλου μήκους ακολουθίες. Ωστόσο, στην πράξη είναι περιορισμένα μόνο σε μικρού μήκους λόγω του vanishing gradient ή exploding gradient προβλήματος. Το μειονέκτημα αυτό καλούνται να επιλύσουν τα Long Short Term Memory networks.

3.3.7 Long short - term memory (LSTM)

Πρόκειται για μία παραλλαγή του RNN, που προτάθηκε για πρώτη φορά το 1997 από τους Sepp Hochreiter & Jürgen Schmidhuber. Τα δίκτυα αυτά καταφέρνουν να ξεπερνούν τον πρόβλημα των μακρινών εξαρτήσεων. Είναι σχεδιασμένα με τέτοιο τρόπο, ώστε να διατηρούν ένα πιο σταθερό σφάλμα κατά την ανάστροφη μετάδοσή του μέσα στον χρόνο, μαθαίνοντας έτσι ακόμα και μέσα από πολύ μεγάλες ακολουθίες χρονικών βημάτων. Αυτό, που τα ξεχωρίζει από τα απλά αναδρομικά νευρωνικά δίκτυα, είναι η αρχιτεκτονική του κρυφού τους επιπέδου, το οποίο συνηθίζεται και να αποκαλείται ως κύτταρο LSTM [59]. Ένα κύτταρο LSTM διαθέτει τρεις θύρες: τη θύρα λήθης, εισόδου και εξόδου.

- **Θύρα λήθης (forget gate):** Το πρώτο βήμα είναι να αποφασίσουμε ποια πληροφορία θα διαγράψουμε από τη μνήμη. Η απόφαση αυτή λαμβάνεται από τη θύρα λήθης. Δέχεται ως είσοδο την τρέχουσα είσοδο x_t και την έξοδο του προηγούμενου timestep h_{t-1} και παράγει στην έξοδο έναν αριθμό μεταξύ του 0 και 1 (σιγμοειδής συνάρτηση ενεργοποίησης). Η έξοδος αυτή πολλαπλασιάζεται με κάθε αριθμό του διανύσματος C_{t-1} της προηγούμενης κατάστασης, ρυθμίζοντας έτσι το ποια πληροφορία θα «ξεχαστεί».

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \tag{3.41}$$

- **Θύρα εισόδου (input gate):** Το επόμενο βήμα είναι να αποφασίσουμε ποια νέα πληροφορία πρόκειται να αποθηκεύσουμε στη μνήμη. Αρχικά, μέσω της θύρας εισόδου αποφασίζεται ποιες τιμές θα ενημερωθούν. Η θύρα εισόδου δέχεται ως είσοδο την τρέχουσα είσοδο x_t και την έξοδο του προηγούμενου timestep h_{t-1}. Στη συνέχεια, η τρέχουσα είσοδος x_t και η έξοδος του προηγούμενου timestep h_{t-1} περνάνε μέσα από ένα μονο-επίπεδο νευρωνικό δίκτυο με συνάρτηση ενεργοποίησης την υπερβολική εφαπτομένη, το οποίο παράγει τις νέες υποψήφιες τιμές $\tilde{c}_t$, που πρόκειται να προστεθούν στη μνήμη.

$$\begin{aligned} i_t &= \sigma(W_i x_t + U_i h_{t-1} + b_i) \quad \text{θύρα εισόδου} \\ \tilde{c}_t &= \tanh\left(W_c x_t + U_c h_{t-1} + b_c\right) \quad \text{υποψήφια μνήμη (candidate cell state)} \end{aligned} \tag{3.42}$$

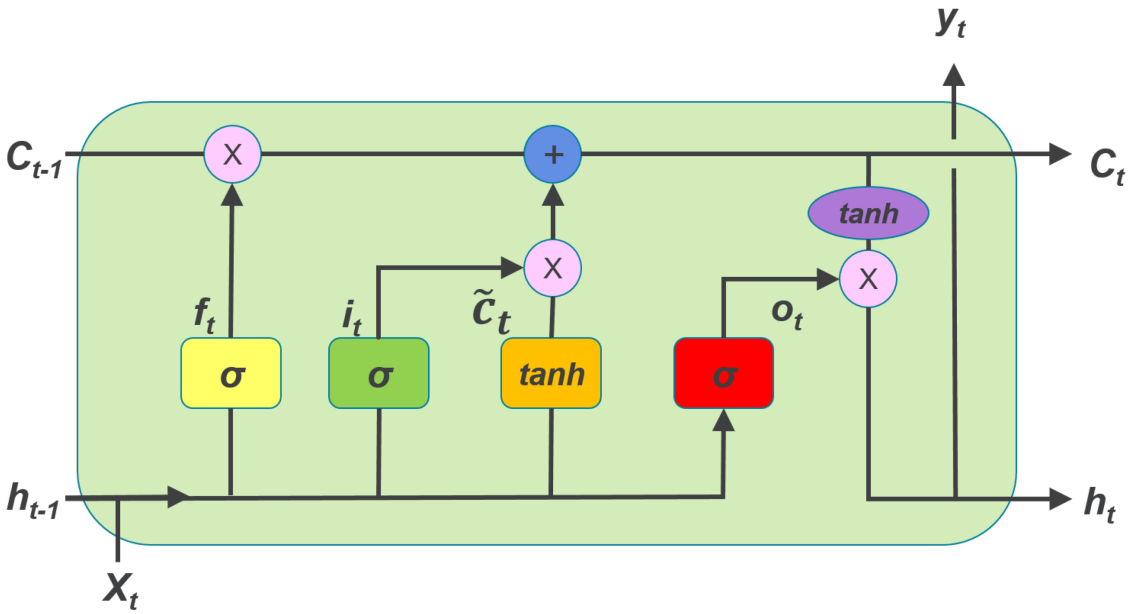
- **Μνήμη (cell state):** Στο βήμα αυτό, ανανεώνουμε την παλιά μνήμη c_{t-1} στην καινούρια μνήμη c_t. Συγκεκριμένα, πολλαπλασιάζουμε τη θύρα λήθης με τις τιμές της παλιάς μνήμης c_{t-1}, ξεχνώντας όρους, που αποφασίστηκε να ξεχαστούν σε προηγούμενο βήμα. Προσθέτουμε, επίσης, τον όρο $i_t \odot \tilde{c}_t$. Πρόκειται για νέες υποψήφιες τιμές, κλιμακωμένες κατά το πόσο αποφασίστηκε να ενημερώσουμε την τρέχουσα κατάσταση.

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \tag{3.43}$$

  Συμβολίζουμε ως $\odot$ το γινόμενο Hadamard (πολλαπλασιασμός στοιχείο προς στοιχείο).

- **Θύρα εξόδου (output gate):** Η έξοδος βασίζεται σε μία φιλτραρισμένη εκδοχή της κατάστασης της μνήμης. Αρχικά, η τρέχουσα είσοδος x_t και η έξοδος του προηγούμενου timestep h_{t-1} περνάνε μέσα από ένα μονο-επίπεδο νευρωνικό δίκτυο με συνάρτηση ενεργοποίησης τη σιγμοειδή, προκειμένου να αποφασιστεί ποια μέρη της κατάστασης μνήμης πρόκειται να συμμετέχουν στην τελική έξοδο. Έπειτα, η κατάσταση της μνήμης, αφού συμπιεστεί μέσω της συνάρτησης tanh, πολλαπλασιάζεται με τη θύρα εξόδου, αποφασίζοντας επομένως ποια μέρη της κατάστασης αυτής θα συμμετέχουν στην τελική έξοδο.

$$\begin{aligned} o_t &= \sigma(W_o x_t + U_o h_{t-1} + b_o) \quad \text{θύρα εισόδου} \\ h_t &= o_t \odot \tanh c_t \quad \text{έξοδος (output)} \end{aligned} \tag{3.44}$$



Εικόνα 3.18: Κύτταρο LSTM

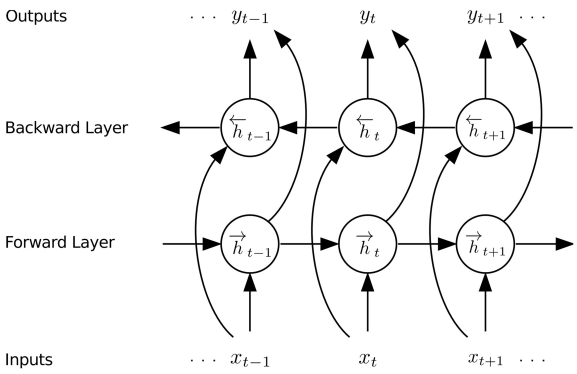
3.3.8 Αμφίδρομο LSTM

Τα αμφίδρομα LSTM (Bidirectional LSTM ή BiLSTM) αποτελούν μία επέκταση των κλασσικών LSTMs και μπορούν να βελτιώσουν την απόδοση του μοντέλου σε ακολουθιακά προβλήματα ταξινόμησης. Συγκεκριμένα, ένα αμφίδρομο LSTM αποτελείται από τον συνδυασμό δύο διαφορετικών LSTMs, όπου το καθένα επεξεργάζεται την ακολουθία με διαφορετική φορά. Βασίζεται στην ιδέα ότι η έξοδος κάθε χρονική στιγμή εξαρτάται τόσο από προηγούμενα όσο και από επόμενα στοιχεία στην ακολουθία. Για παράδειγμα, προκειμένου να προβλέψουμε μία λέξη που λείπει σε μία πρόταση - ακολουθία, πρέπει να δούμε τις λέξεις τόσο πριν όσο και μετά τη λέξη που ψάχνουμε, έτσι ώστε να κατανοήσουμε το περιεχόμενο της πρότασης.

Στην παρακάτω εικόνα, ένα LSTM επεξεργάζεται την ακολουθία από αριστερά προς τα δεξιά, ενώ το δεύτερο LSTM την επεξεργάζεται από δεξιά προς αριστερά. Σε κάθε χρονική στιγμή t, ένα κρυφό δεξιόστροφο LSTM με κρυφή κατάσταση $\vec{h}$ δέχεται στην είσοδο την προηγούμενη κρυφή κατάσταση $\overrightarrow{h_{t-1}}$ και την είσοδο x_t στην τωρινή χρονική στιγμή t. Επιπρόσθετα, ένα κρυφό αριστερόστροφο LSTM με κρυφή κατάσταση $\overleftarrow{h}$ δέχεται ως είσοδο την είσοδο x_t στην τωρινή χρονική στιγμή t, αλλά και τη μελλοντική κρυφή κατάσταση $\overrightarrow{h_{t+1}}$. Επομένως, για κάθε χρονική στιγμή t έχουμε:

$$h_i = [\vec{h_i^T}, \overleftarrow{h_i^T}]^T \tag{3.45}$$

όπου το h_i αποτελεί το concatenation των κρυφών καταστάσεων του αριστερόστροφου και του δεξιόστροφου LSTM.

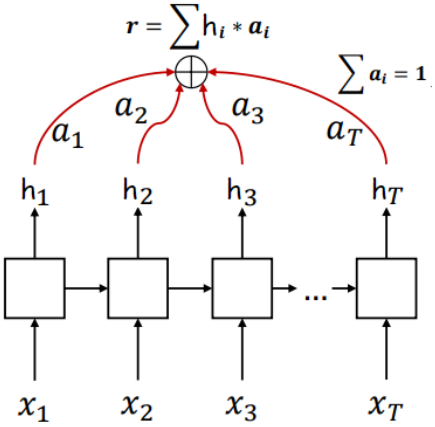


Εικόνα 3.19: Bidirectional LSTM

3.3.9 Μηχανισμός Προσοχής (Attention Mechanism)

Οι Bahdanau et al. [60] ήταν αυτοί, που χρησιμοποίησαν πρώτοι τον μηχανισμό προσοχής σε μηχανική μετάφραση στον τομέα της επεξεργασίας φυσικής γλώσσας.

Συγκεκριμένα, ο μηχανισμός προσοχής εφαρμόζεται στο ανώτερο στρώμα ενός RNN και εστιάζει στα σημαντικότερα τμήματα της ακολουθίας εισόδου, αγνοώντας μέρη της ακολουθίας, τα οποία δεν επιδρούν στην ταξινόμηση. Τα στοιχεία του διανύσματος α, που προκύπτει από τον μηχανισμό προσοχής αποτελούν τους συντελεστές επιρροής της κάθε κατάστασης στην τελική έξοδο και λόγω της χρήσης της συνάρτησης ενεργοποίησης Softmax, αθροίζουν στη μονάδα. Αναλυτικότερα, ο μηχανισμός αυτός προσδίδει βάρη στις κρυφές καταστάσεις h_i.



Εικόνα 3.20: Attention Mechanism

- **Attention Vector**

$$e_i = \tanh\left(W_h h_i + b_h\right), \quad e_i \in [-1, 1] \tag{3.46}$$

- **Attention weights**

$$a_i = softmax(e_i) = \frac{exp(e_i)}{\sum_{t=1}^{T} exp(e_t)}, \quad \sum_{i=1}^{T} a_i = 1 \tag{3.47}$$

- **Context Vector**

$$r = \sum_{i=1}^{T} a_i h_i, \quad r \in R^{2L} \tag{3.48}$$

3.4 Επιλογή Παραμέτρων (Hyperparameter Tuning)

Ορισμένες από τις πιο γνωστές μεθόδους που χρησιμοποιούνται για τη βελτιστοποίηση των υπερ-παραμέτρων αλγορίθμων μάθησης είναι οι grid search, random search [61] και Μπεϋζιανή Βελτιστοποίηση [62].

3.4.1 Μέθοδος Grid Search

Η μέθοδος grid search αποτελεί μία κλασσική μέθοδο που χρησιμοποιείται για τη βελτιστοποίηση των υπερπαραμέτρων, η οποία αντιστοιχεί σε εξαντλητική αναζήτηση των καλύτερων υπερπαραμέτρων ενός αλγορίθμου μάθησης σε ένα συγκεκριμένο υποσύνολο του χώρου, στον οποίο ορίζονται. Το υποσύνολο, στο οποίο γίνεται η αναζήτηση των υπερ-παραμέτρων, ορίζεται από το χρήστη. Οι τιμές των υπερπαραμέτρων μπορεί να είναι διακριτές ή συνεχείς, οπότε θα πρέπει να γίνεται προσεκτική επιλογή του υποσυνόλου, στο οποίο θα εφαρμοστεί ο αλγόριθμος grid search. Ένας αλγόριθμος grid search αξιολογείται μέσω cross-validation και καθοδηγείται από κάποια μετρική απόδοσης, όπως για παράδειγμα από το accuracy για προβλήματα ταξινόμησης ή RMSE για προβλήματα παλινδρόμησης.

3.4.2 Μέθοδος Random Search

Η μέθοδος random search χρησιμοποιεί τυχαίους συνδυασμούς υπερπαραμέτρων, έτσι ώστε να καταλήξει στο συνδυασμό που επιφέρει την καλύτερη απόδοση του μοντέλου. Καθώς σε κάθε επανάληψη του μοντέλου επιλέγονται τυχαίες τιμές, είναι αρκετά πιθανό να βρεθούν οι βέλτιστες υπερπαράμετροι σε ελάχιστο χρόνο, σε αντιδιαστολή με τη μέθοδο grid search, η οποία είναι μία υπολογιστικά ακριβή διαδικασία, όπως προαναφέρθηκε. Η μέθοδος random search υπερέχει έναντι της grid search σε περιπτώσεις, όπου δεν είναι εξίσου σημαντικές όλες οι υπερπαράμετροι του αλγορίθμου. Στην περίπτωση αυτή, η πιθανότητα να βρεθεί ο βέλτιστος συνδυασμός υπερπαραμέτρων είναι μεγαλύτερη, καθώς το μοντέλο μπορεί να έχει εκπαιδευτεί, τελικά, με τις βέλτιστες υπερπαραμέτρους. Η καλύτερη απόδοση της μεθόδου random search εντοπίζεται κυρίως σε δεδομένα λίγων διαστάσεων, καθώς απαιτείται λιγότερος αριθμός επαναλήψεων για να βρεθεί το βέλτιστο σετ υπερπαραμέτρων.

3.4.3 Μπεϋζιανή Βελτιστοποίηση

Η τεχνική αυτή χτίζει ένα πιθανοτικό μοντέλο για την αντικειμενική συνάρτηση. Τα αποτελέσματα κάθε δοκιμής, χρησιμοποιούνται ως δεδομένα για την ενημέρωση των πεποιθήσεων του μοντέλου. Με αυτό τον τρόπο, αν για παράδειγμα μικρές τιμές μίας παραμέτρου οδηγούν σε κακά αποτελέσματα, τότε θα είναι πιο δύσκολο να επιλεχθεί μία μικρή τιμή για την παράμετρο αυτή σε επόμενες δοκιμές.

3.5 Αξιολόγηση του Αλγορίθμου Μηχανικής Μάθησης

3.5.1 Μετρικές Αξιολόγησης

3.5.1.1 Πίνακας Σύγχυσης (Confusion Matrix)

Στο πεδίο της μηχανικής μάθησης και συγκεκριμένα σε προβλήματα στατιστικής ταξινόμησης, ο πίνακας σύγχυσης (confusion matrix) ορίζεται ως ένας ειδικός πίνακας, που επιτρέπει την απεικόνιση της απόδοσης ενός αλγορίθμου επιβλεπόμενης μάθησης. Στη μη επιβλεπόμενη μάθηση, ο πίνακας αυτός ονομάζεται πίνακας ταιριάσματος (matching matrix). Κάθε γραμμή του πίνακα αναπαριστά τις περιπτώσεις σε μία προβλεπόμενη κλάση, ενώ κάθε στήλη του πίνακα αναπαριστά τις περιπτώσεις σε μία πραγματική κλάση.

		True Condition	
		Condition Positive	**Condition Negative**
Predicted Condition	**Predicted condition positive**	**True Positive (TP)**	**False Positive (FP)**
	Predicted condition negative	**False Negative (FN)**	**True Negative (TN)**

Πίνακας 3.1: Πίνακας Σύγχυσης

Ορίζουμε τις μεταβλητές του πίνακα σύγχυσης ως εξής:

- **True Positive (TP):** Πρόκειται για περιπτώσεις, στις οποίες έχουμε προβλέψει θετικά και η πρόβλεψη επιβεβαιώνεται.
- **True Negative (TN):** Πρόκειται για περιπτώσεις, στις οποίες έχουμε προβλέψει αρνητικά και η πρόβλεψη επιβεβαιώνεται.
- **False Positive (FP):** Πρόκειται για περιπτώσεις, στις οποίες έχουμε προβλέψει θετικά και η πρόβλεψη δεν επιβεβαιώνεται.
- **False Negative (FN):** Πρόκειται για περιπτώσεις, στις οποίες έχουμε προβλέψει αρνητικά και η πρόβλεψη δεν επιβεβαιώνεται.

Με βάση αυτές τις μεταβλητές ορίζονται, επιπλέον οι εξής παράμετροι:

- **Ορθότητα (accuracy):** Ορίζεται ως το ποσοστό των σωστών προβλέψεων του ταξινομητή επί του συνόλου των προβλέψεων που έγιναν. Δίνεται από τον παρακάτω τύπο:

$$accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions} = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.49}$$

  Ωστόσο, δεν αρκεί να εξετάσουμε μόνο την ορθότητα του μοντέλου μας, ώστε να αξιολογήσουμε την απόδοσή του. Σε περιπτώσεις μεγάλης ανομοιομορφίας στην κατανομή των δειγμάτων των κλάσεων, ο ταξινομητής θα προβλέπει την κλάση με τα περισσότερα δεδομένα, με αποτέλεσμα να επιτυγχάνεται πολύ υψηλό accuracy. Ωστόσο, δεν θα μπορεί να προβλέψει σωστά την κλάση με τα λιγότερα δεδομένα. Για τον λόγο αυτό, ορίζουμε τις παρακάτω μετρικές.

- **Ανάκληση (Recall/ True Positive Rate/ Sensitivity):** Ορίζεται ως το ποσοστό των θετικών δειγμάτων, που προβλέφθηκαν σωστά από τον ταξινομητή. Δίνεται από τον ακόλουθο τύπο:

$$recall = \frac{TP}{TP + FN} \tag{3.50}$$

- **Ακρίβεια (precision):** Ορίζεται ως το ποσοστό των σωστών θετικών προβλέψεων του ταξινομητή. Δίνεται από τον παρακάτω τύπο:

$$precision = \frac{TP}{TP + FP} \tag{3.51}$$

- **Μετρική F1-score:** Αποτελεί το μέτρο, το οποίο συνδυάζει ακρίβεια και ανάκληση. Συγκεκριμένα, είναι ο αρμονικός μέσος της ακρίβειας και της ανάκλησης και μαθηματικά εκφράζεται από τον παρακάτω τύπο:

$$F1 - score = 2 \cdot \frac{precision \cdot recall}{precision + recall} = 2\frac{TP}{2 \cdot TP + FP + FN} \tag{3.52}$$

- **False Positive Rate:** Ορίζεται ως το ποσοστό των αρνητικών δειγμάτων που ο ταξινομητής λανθασμένα προέβλεψε ως θετικά.

$$FPR = \frac{FP}{TN + FP} \tag{3.53}$$

- **Specificity (True Negative Rate):** Ορίζεται ως το ποσοστό των αρνητικών δειγμάτων, που προβλέφθηκαν σωστά από τον ταξινομητή. Δίνεται από τον ακόλουθο τύπο:

$$Specificity = \frac{TN}{TN + FP} \tag{3.54}$$

- **Matthews Correlation Coefficient (MCC):** Ορίζεται ως η εκτίμηση της συσχέτισης μεταξύ της κλάσης, που προβλέπεται και της κλάσης, στην οποία ανήκουν πραγματικά οι χρήστες.

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FN) \cdot (TP + FP) \cdot (TN + FP) \cdot (TN + FN)}} \tag{3.55}$$

- **ROC Curve and AUC (Area under Curve):** Είναι ένα γράφημα, που δείχνει την επίδοση ενός μοντέλου ταξινόμησης σε όλα τα κατώφλια ταξινόμησης. Η καμπύλη αυτή αναπαριστά δύο παραμέτρους, το TPR & FPR.

3.5.2 Cross Validation - Αξιολόγηση & Βελτιστοποίηση Μοντέλων

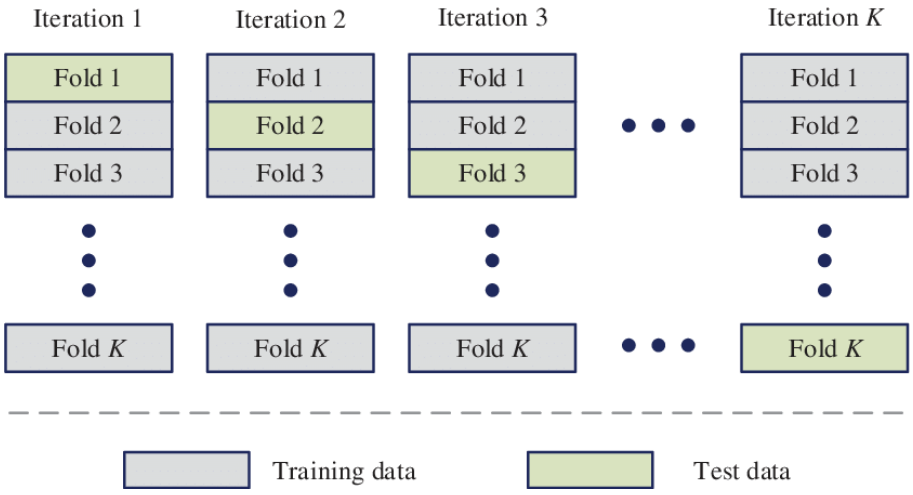
Μία μέθοδος αξιολόγησης το μοντέλου είναι το `train_test_split`, δηλαδή ο διαχωρισμός των δεδομένων σε training & test set ορίζοντας το test size. Το μοντέλο εκπαιδεύεται στο training set και στη συνέχεια αξιολογείται στο test set. Ωστόσο, η μέθοδος αυτή, σε περίπτωση που έχουμε μικρό αριθμό δεδομένων, ενδέχεται να οδηγήσει σε biased αποτελέσματα. Επίσης, δεν μπορεί να μας εξασφαλίσει ότι το μοντέλο δεν είναι overfitted στο training set ή το test set, που επιλέχθηκε, δεν έτυχε να είναι αρκετά εύκολο στις προβλέψεις του μοντέλου. Τα προβλήματα αυτά καλείται να λύσει η τεχνική k- fold cross validation, η οποία αναλύεται στη συνέχεια.

1. Επίλεξε μία τιμή για το k και χώρισε το dataset σε k τμήματα.

2. Εκπαίδευσε το μοντέλο χρησιμοποιώντας ως training set τα k-1 τμήματα και αξιολόγησε το μοντέλο χρησιμοποιώντας ως test set το k-οστό τμήμα (αυτό που έχει απομείνει). Κράτησε το επιθυμητό αποτέλεσμα / μετρική αξιολόγησης του μοντέλου (π.χ accuracy, precision, F1 - score κ.ά).

3. Επανάλαβε τα βήματα 1 & 2, μέχρι κάθε k-τμήμα να έχει χρησιμοποιηθεί ως test set. Στη συνέχεια, υπολόγισε τον μέσο όρο των επιμέρους μετρικών αξιολόγησης (υπολογίστηκαν στο βήμα 2). Αυτή είναι και η τελική μετρική αξιολόγησης του μοντέλου.

Η διαδικασία του k-fold cross validation, όπως περιγράφηκε, φαίνεται στην παρακάτω εικόνα.



Εικόνα 3.21: k-fold cross validation

3.5.3 Nested Cross-Validation

Σε περιπτώσεις όπου το cross validation χρησιμοποιείται ταυτόχρονα για την επιλογή των βέλτιστων υπερπαραμέτρων και την αξιολόγηση του μοντέλου, τότε απαιτείται να χρησιμοποιηθεί το nested cross validation [63], η διαδικασία του οποίου αναλύεται παρακάτω. Αρχικά, ένα εσωτερικό cross validation χρησιμοποιείται, για να επιλεγούν οι κατάλληλες υπερπαράμετροι και κατά συνέπεια το καλύτερο μοντέλο. Δεύτερον, ένα εξωτερικό cross validation χρησιμοποιείται, προκειμένου να αξιολογήσει το μοντέλο, που επιλέχθηκε από το εσωτερικό cross validation.

1. Επίλεξε μία τιμή για το k και χώρισε το dataset σε k τμήματα.

2. Για κάθε τμήμα k=1,2,...,K: εξωτερικός βρόχος για τελική αξιολόγηση του μοντέλου με τις επιλεγμένες υπερπαραμέτρους.

   2.1 Όρισε το τμήμα k ως το test set.

   2.2 Χρησιμοποίησε ως training set τα υπόλοιπα τμήματα, εκτός του k.

   2.3 Επίλεξε μία τιμή για το L και χώρισε το training set του βήματος 2.2 σε L τμήματα.

   2.4 Για κάθε τμήμα l=1,2,...,L: εσωτερικός βρόχος για επιλογή βέλτιστων υπερπαραμέτρων.

      2.4.1 Όρισε το τμήμα l ως το validation set.

      2.4.2 Χρησιμοποίησε ως training set τα υπόλοιπα τμήματα, εκτός του l.

      2.4.3 Εκπαίδευσε το μοντέλο με κάθε υπερπαράμετρο στο training set του βήματος 2.4.2 και αξιολόγησέ το στο validation set του βήματος 2.4.1. Κράτησε το επιθυμητό αποτέλεσμα / μετρική αξιολόγησης του μοντέλου (π.χ accuracy, precision, F1 - score κ.ά).

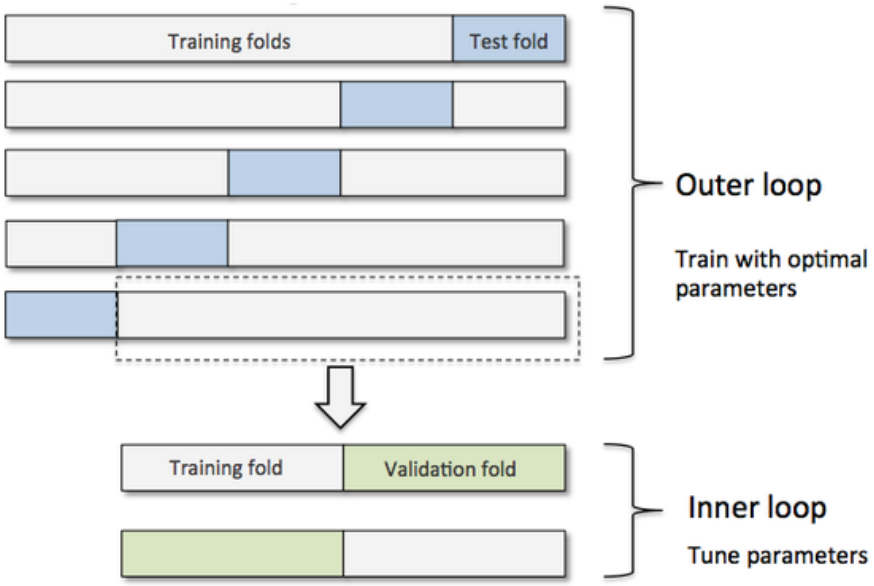
   2.5 Για κάθε σύνολο υπερπαραμέτρων υπολόγισε τον μέσο όρο των επιμέρους μετρικών αξιολόγησης και βάσει αυτού το μέσου όρου επίλεξε το καλύτερο σύνολο υπερπαραμέτρων.

2.6 Εκπαίδευσε το μοντέλο με τις καλύτερες υπερπαραμέτρους στο training set του βήματος 2.2 και αξιολόγησε την απόδοσή του στο test set του βήματος 2.1. Κράτησε το επιθυμητό αποτέλεσμα / μετρική αξιολόγησης του μοντέλου (π.χ accuracy, precision, F1 - score κ.ά).

3. Υπολόγισε τον μέσο όρο των επιμέρους μετρικών αξιολόγησης (υπολογίστηκαν στο βήμα 2). Αυτή είναι και η τελική μετρική αξιολόγησης του μοντέλου.

Σημείωση: Επανάλαβε τα βήματα 2.1 & 2.2, έτσι ώστε όλα τα τμήματα k να έχουν συμπεριφερθεί ως test set. Ομοίως, επανάλαβε τα βήματα 2.4.1 & 2.4.2, έτσι ώστε όλα τα τμήματα l για κάθε σύνολο υπερπαραμέτρων να έχουν συμπεριφερθεί ως validation set.

Η διαδικασία του nested cross validation, όπως περιγράφηκε, φαίνεται στην παρακάτω εικόνα.



Εικόνα 3.22: Nested Cross Validation

3.6 Μη Ισορροπημένη Μάθηση (Imbalanced Learning)

Ένα κοινό πρόβλημα κατά την ταξινόμηση στον κλάδο της μηχανικής μάθησης είναι η ύπαρξη ανομοιογενών συνόλων δεδομένων (imbalanced datasets), όπου κάποιες κλάσεις μεροληπτούν σημαντικά υπέρ των άλλων. Αυτό έχει ως αποτέλεσμα κάθε νέο δεδομένο, που πρόκειται να ταξινομηθεί, να ταξινομείται στην κλάση με τα περισσότερα δεδομένα, καθώς το μοντέλο μηχανικής μάθησης θα έχει δει την κλάση αυτή τις περισσότερες φορές. Προκειμένου να αντιμετωπιστεί αυτό το πρόβλημα, έχουν αναπτυχθεί τεχνικές [64], μερικές από τις οποίες αναλύονται στη συνέχεια.

3.6.1 Μέθοδοι Δειγματοληψίας (Sampling Methods)

Random Oversampling: Οι μηχανισμοί της τυχαίας υπερδειγματοληψίας λειτουργούν προσθέτοντας ένα σύνολο που προκύπτει από την τάξη της μειοψηφίας. Συγκεκριμένα, για ένα σύνολο τυχαία επιλεγμένων παραδειγμάτων μειοψηφίας αυξάνεται το αρχικό σύνολο αναπαράγοντας τα επιλεγμένα παραδείγματα και προσθέτοντάς τα στο αρχικό σύνολο. Το μειονέκτημα της μεθόδου αυτής είναι ότι μπορεί να οδηγήσει σε overfitting λόγω των πολλαπλών ίδιων δεδομένων στο σύνολο εκπαίδευσης.

Random Undersampling: Κατά την τεχνική αυτή επιλέγεται τυχαία ένα σύνολο δεδομένων από την τάξη με τα περισσότερα δεδομένα και αφαιρείται από το αρχικό σύνολο δεδομένων, έτσι ώστε να προκύψει ένα ομοιογενές σύνολο δεδομένων (balanced dataset). Το μειονέκτημα της μεθόδου αυτής είναι η πιθανή απώλεια πληροφορίας λόγω της αφαίρεσης ενός συνόλου δεδομένων. Για την αντιμετώπιση του προβλήματος αυτού έχουν αναπτυχθεί οι αλγόριθμοι EasyEnsemble & BalanceCascade.

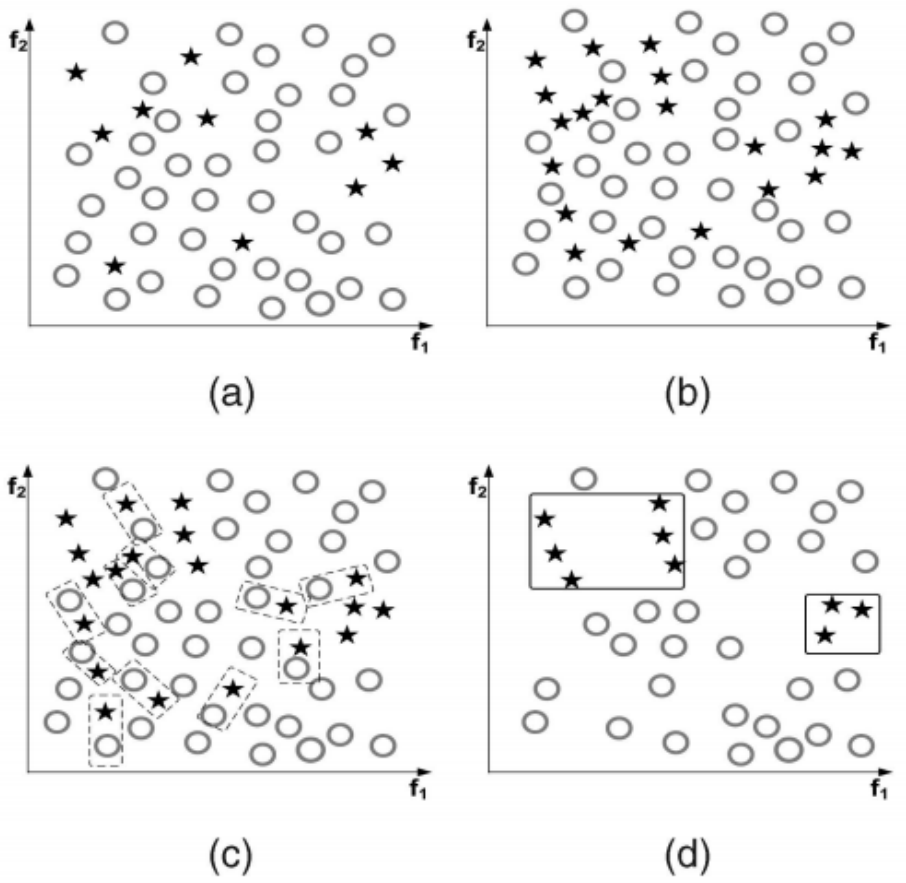
Στη συνέχεια, αναφέρουμε ορισμένες τεχνικές παραγωγής συνθετικών δεδομένων.

SMOTE algorithm [65]: Ο αλγόριθμος αυτός δημιουργεί τεχνητά δεδομένα, τα οποία βασίζονται στις ομοιότητες μεταξύ των παραδειγμάτων της κλάσης, που μειοψηφεί. Συγκεκριμένα, για ένα

υποσύνολο $S_{min} \in S$ θεωρούμε τους K κοντινότερους γείτονες του $x_i \in S_{min}$, όπου x_i το στιγμιότυπο μειοψηφίας. Ορίζουμε τους K κοντινότερους γείτονες ως τα K στοιχεία του υποσυνόλου S_{min}, των οποίων η ευκλείδεια απόσταση με το στιγμιότυπο μειοψηφίας x_i είναι η μικρότερη δυνατή. Για τη δημιουργία ενός συνθετικού δείγματος, επιλέγεται τυχαία ένας από τους K κοντινότερους γείτονες, ο οποίος συμβολίζεται ως $\hat{x_i}$ και στη συνέχεια η διαφορά των διανυσμάτων $\hat{x_i}$ & x_i πολλαπλασιάζεται με έναν τυχαίο αριθμό δ μεταξύ [0,1]. Τελικά, το διάνυσμα αυτό προστίθεται στο x_i .Καταλήγουμε, δηλαδή, στην παρακάτω εξίσωση:

$$x_{new} = x_i + (\hat{x_i} - x_i) \times \delta \tag{3.56}$$

Για την απομάκρυνση της επικάλυψης, που παρουσιάζεται από τις μεθόδους δειγματοληψίας, έχουν αναπτυχθεί διάφορες τεχνικές εκκαθάρισης δεδομένων (data cleaning). Η μέθοδος συνδέσμων Tomek μπορεί να οριστεί ως ένα ζεύγος πλησιέστερων γειτόνων αντίθετων τάξεων με ελαχιστοποιημένη απόσταση. Οι σύνδεσμοι Tomek μπορούν να χρησιμοποιηθούν, για να εκκαθαρισθούν ανεπιθύμητες επικαλύψεις μεταξύ τάξεων μετά από συνθετική δειγματοληψία, όπου όλοι οι σύνδεσμοι Tomek απομακρύνονται, έως ότου όλα τα ζεύγη πλησιέστερων γειτόνων με ελαχιστοποιημένη απόσταση να είναι της ίδιας τάξης. Με την απομάκρυνση παραδειγμάτων επικάλυψης είναι δυνατό να επιτευχθούν συμπλέγματα με καλά ορισμένες τάξεις στο σύνολο δοκιμών, το οποίο στη συνέχεια οδηγεί σε καλά ορισμένους κανόνες ταξινόμησης με βελτιωμένη απόδοση ταξινόμησης. Κάποιες ενδεικτικές εργασίες σε αυτήν την περιοχή αποτελούν οι ενσωματώσεις του SMOTE με ENN(SMOTE+ENN) ή του SMOTE με τις συνδέσεις Tomek (SMOTE+Tomek), όπου ENN ορίζεται ως ο κανόνας επεξεργασίας του πλησιέστερου γείτονα. Στην παρακάτω εικόνα, φαίνεται η κατανομή ενός συνόλου δεδομένων μετά την εφαρμογή του SMOTE αλγορίθμου ενσωματωμένου με τις συνδέσεις Tomek.



Εικόνα 3.23: *(a)* Original dataset distribution, *(b)* Post-SMOTE dataset, *(c)* The identified Tomek links, *(d)* The dataset after removing Tomek links

Τέλος, για την απομάκρυνση της επικάλυψης, που παρουσιάζεται από τις μεθόδους δειγματοληψίας, έχουν αναπτυχθεί διάφορες προσαρμοστικές μέθοδοι δειγματοληψίας, όπως οι
Borderline-SMOTE και Adaptive Synthetic Sampling (ADASYN).

3.6.2 Μάθηση Ευαίσθητη στο Κόστος (Cost Sensitive Learning)

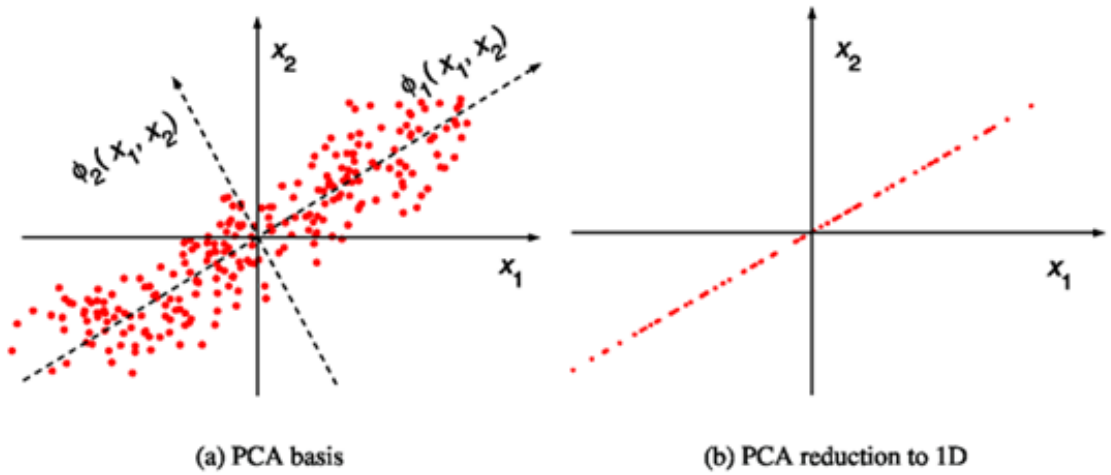
Ενώ οι μέθοδοι δειγματοληψίας προσπαθούν να ισορροπήσουν τα δεδομένα στις διαφορετικές κλάσεις, οι μέθοδοι της μάθησης ευαίσθητης στο κόστος λαμβάνουν υπόψη τα κόστη, που σχετίζονται με δεδομένα, που έχουν ταξινομηθεί λανθασμένα. Η μάθηση ευαίσθητη στο κόστος στοχεύει στο πρόβλημα της μη ισορροπημένης μάθησης χρησιμοποιώντας διαφορετικούς πίνακες κόστους, που περιγράφουν τα κόστη της λανθασμένης ταξινόμησης κάθε συγκεκριμένου δείγματος δεδομένων.

3.6.3 Ενεργητική Μάθηση (Active Learning)

Οι μέθοδοι ενεργητικής μάθησης χρησιμοποιούνται, για να λύσουν προβλήματα, που σχετίζονται με μη επισημασμένα δεδομένα (unlabeled data). Αυτές οι μέθοδοι δεν ψάχνουν όλο τον χώρο των δεδομένων εκπαίδευσης, αλλά επιλέγουν με αποτελεσματικό τρόπο ενημερωτικά δείγματα από ένα τυχαία σύνολο πληθυσμών εκπαίδευσης, μειώνοντας , έτσι, σημαντικά το υπολογιστικό κόστος σε μη ισορροπημένα σύνολα δεδομένων.

3.7 Μείωση Διαστάσεων (Dimensionality Reduction)

Η τεχνική μείωσης των διαστάσεων αναφέρεται στη διαδικασία μετατροπής ενός συνόλου δεδομένων με πολλές διαστάσεις (στήλες) σε ένα σύνολο δεδομένων μικρότερων διαστάσεων (λιγότερες στήλες), διασφαλίζοντας όμως ότι περιέχει παρόμοια πληροφορία με το αρχικό. Συνήθεις τεχνικές είναι οι Singular Value Decomposition (SVD), Principal Component Analysis (PCA) και Latent Discriminant Analysis (LDA). Οι τεχνικές αυτές "συμπιέζουν" διαστάσεις με μικρή διακύμανση, ενώ διατηρούν αυτές με μεγάλη διακύμανση, δηλαδή με μεγάλη πληροφορία. Στην παρακάτω εικόνα, βλέπουμε πώς μετασχηματίζεται ένα σύνολο δεδομένων από τον χώρο R^2 στον χώρο R^1 με χρήση της τεχνικής PCA, την οποία και θα χρησιμοποιήσουμε στο πείραμά μας.



Εικόνα 3.24: Principal Component Analysis (PCA)

3.8 Επιλογή Χαρακτηριστικών (Feature Selection)

Συχνά, ερχόμαστε αντιμέτωποι με datasets, τα οποία περιέχουν έναν αρκετά μεγάλο αριθμό χαρακτηριστικών (features). Τα datasets αυτά, που είναι γνωστά και ως datasets πολλών διαστάσεων (high dimensionality datasets), περιλαμβάνουν κάποια εντελώς ασήμαντα features. Έχει δειχθεί ότι τα features αυτά συνεισφέρουν ελάχιστα ή ακόμα και καθόλου στη διαδικασία πρόβλεψης. Συγκεκριμένα, τα χαρακτηριστικά ίσως είναι μη αποδοτικά στην κατασκευή ενός μοντέλου ταξινόμησης για δύο λόγους [66] ***(α)*** δεν έχουν καμία σχέση με την κλάση (label), οπότε χαρακτηρίζονται ως irrelevant και ***(β)*** συνεισφέρουν σημαντικά στην πρόβλεψη της κλάσης, αλλά ταυτόχρονα έχουν και υψηλή συσχέτιση με άλλα χαρακτηριστικά, οπότε χαρακτηρίζονται ως redundant. Δημιουργούν, επομένως, ένα πλήθος προβλημάτων, τα οποία με τη σειρά τους κάνουν ολοένα και δυσκολότερη τη διαδικασία της πρόβλεψης σε θέματα μηχανικής μάθησης. Συγκεκριμένα, συμπεριφέρονται ως θόρυβος, με αποτέλεσμα το

μοντέλο μηχανικής μάθησης να έχει χαμηλές επιδόσεις σε μετρικές αξιολόγησης. Επιπλέον, τα χαρακτηριστικά αυτά τείνουν να αυξήσουν τον χρόνο εκπαίδευσης του μοντέλου μας, με αποτέλεσμα να το κάνουν ολοένα και πιο πολύπλοκο, οδηγώντας έτσι στο φαινόμενο της υπερπροσαρμογής (overfitting).

Για τους λόγους αυτούς, καταφεύγουμε στη μέθοδο του feature selection [67]. Πρόκειται για μία διαδικασία, κατά την οποία επιλέγονται τα σημαντικότερα χαρακτηριστικά ενός συγκεκριμένου dataset. Η μέθοδος του feature selection είναι σημαντική για τους παρακάτω λόγους:

- Καθιστά ικανούς τους αλγορίθμους μηχανικής μάθησης να εκπαιδεύονται γρηγορότερα.
- Μειώνει την πολυπλοκότητα του μοντέλου.
- Βελτιώνει τις μετρικές αξιολόγησης (ακρίβεια, ανάκληση κ.ά), εφόσον επιλεγεί το κατάλληλο υποσύνολο.
- Μειώνει την υπερπροσαρμογή.

Παρακάτω, παρουσιάζονται οι μέθοδοι του feature selection:

3.8.1 Μέθοδοι φιλτραρίσματος (Filter Methods)

Οι μέθοδοι αυτές [68] επιλέγουν υποσύνολα των χαρακτηριστικών ως στάδιο προεπεξεργασίας, χωρίς να χρησιμοποιούν κάποιον αλγόριθμο μηχανικής μάθησης. Χρησιμοποιούνται για τη μείωση της υπερπροσαρμογής (overfitting). Συνήθως, εφαρμόζουν ένα στατιστικό μέτρο, για να αναθέσουν μια βαθμολογία-score για κάθε χαρακτηριστικό. Τα χαρακτηριστικά κατατάσσονται σύμφωνα με τη βαθμολογία τους και, έπειτα, επιλέγονται είτε να παραμείνουν είτε να αφαιρεθούν από το σύνολο δεδομένων.

Εικόνα 3.25: Filter Methods

- **Αμοιβαία Πληροφορία (Mutual Information):** Η αμοιβαία πληροφορία μεταξύ δύο τυχαίων μεταβλητών είναι μία μη αρνητική τιμή, η οποία μετρά την εξάρτηση μεταξύ των μεταβλητών. Είναι ίση με το 0, αν και μόνο αν οι δύο τυχαίες μεταβλητές είναι ανεξάρτητες, ενώ υψηλότερες τιμές ισοδυναμούν με υψηλότερη εξάρτηση μεταξύ των μεταβλητών αυτών. Η τεχνική αυτή μετρά πόση πληροφορία συνεισφέρει η παρουσία ενός χαρακτηριστικού στην σωστή ταξινόμηση. Είναι ισοδύναμη με τα KL - divergence & Information Gain (IG). Ορίζεται ως εξής:

$$MI(X_i, Y) = -\sum_{j=1}^{h}\sum_{k=1}^{l} Pr(X_i = x_{ik}, Y = y_j) log_2 \frac{Pr(X = x_{ik}, Y = y_j)}{Pr(Y = y_j)Pr(X = x_{ik})} \tag{3.57}$$

- **Chi - squared test [69]:** Πρόκειται για ένα στατιστικό τεστ, που μετρά την εξάρτηση ανάμεσα σε στοχαστικές μεταβλητές. Χρησιμοποιώντας το ως μέθοδο, εξαλείφει features, τα οποία είναι ανεξάρτητα της κλάσης και επομένως περιττά και ασήμαντα για την ταξινόμηση.

$$\chi^2 = \sum_{i=1}^{r}\sum_{j=1}^{c} \frac{(O_{ij} - E_{ij})^2}{E_{ij}} \tag{3.58}$$

- **ANOVA F-value**
- **Correlation - based:** Pearson/Spearman Coefficient, signal to noise ratio
- **Relief / ReliefF algorithm**

  Relief [70]: Επιλέγεται τυχαία ένα δεδομένο εκπαίδευσης (training example) και ο αλγόριθμος Relief ψάχνει να βρει τους δύο κοντινότερους γείτονες: έναν της ίδιας κλάσης με το δεδομένο εκπαίδευσης, που ονομάζεται nearest hit H, και ένα από την άλλη κλάση, που ονομάζεται nearest miss M. Στη συνέχεια ανανεώνει τα βάρη W[A] όλων των χαρακτηριστικών A. Εάν τα

δεδομένα R_i & Η του συνόλου εκπαίδευσης έχουν διαφορετικές τιμές στο χαρακτηριστικό Α, τότε το χαρακτηριστικό Α διαχωρίζει τα δύο αυτά δεδομένα σε διαφορετικές κλάσεις, κάτι που δεν είναι επιθυμητό. Για τον λόγο αυτό, αφαιρούμε τον όρο $\frac{(diff(A,R_i,H))}{m}$. Αντίθετα, εάν τα δεδομένα R_i & Μ του συνόλου εκπαίδευσης έχουν διαφορετικές τιμές στο χαρακτηριστικό Α, τότε το χαρακτηριστικό Α διαχωρίζει τα δύο αυτά δεδομένα σε διαφορετικές κλάσεις, κάτι που είναι επιθυμητό. Για τον λόγο αυτό, προσθέτουμε τον όρο $\frac{(diff(A,R_i,M))}{m}$. Η διαδικασία αυτή επαναλαμβάνεται m φορές, όπου η παράμετρος m ορίζεται από τον χρήστη.

1. set all weights W[A]:=0.0;
2. for i:=1 to m do begin
3. randomly select an instance R_i;
4. find nearest hit H and nearest miss M;
5. for A:=1 to α do
6. $W[A] := W[A] - \frac{diff(A,R_i,H))}{m} + \frac{diff(A,R_i,M)}{m}$;
7. end

Η συνάρτηση diff(A,I_1, I_2) υπολογίζει τη διαφορά των τιμών του χαρακτηριστικού Α για τα δεδομένα εκπαίδευσης I_1 & I_2. Για διακριτά (nominal) χαρακτηριστικά ορίζεται ως εξής:

$$diff(A, I_1, I_2) = \begin{cases} 0, & value(A, I_1) = value(A, I_2) \\ 1, & otherwise \end{cases} \tag{3.59}$$

Για αριθμητικά χαρακτηριστικά ορίζεται ως εξής:

$$diff(a, I_1, I_2) = \frac{|value(A, I_1) - value(A, I_2)|}{\max(A) - \min(A)} \tag{3.60}$$

Η συνάρτηση αυτή χρησιμοποιείται, επίσης, για τον υπολογισμό της απόστασης μεταξύ των δεδομένων εκπαίδευσης, προκειμένου να βρεθούν οι κοντινότεροι γείτονες. Η συνολική απόσταση ορίζεται ως το άθροισμα των αποστάσεων όλων των χαρακτηριστικών (Manhattan distance).

ReliefF: Ο αλγόριθμος αυτός αποτελεί μία επέκταση του Relief αλγορίθμου, με την έννοια ότι δεν περιορίζεται σε προβλήματα δύο κλάσεων, είναι περισσότερο εύρωστος και εφαρμόζεται σε προβλήματα με ανεπαρκή δεδομένα. Ομοίως με τον αλγόριθμο Relief, επιλέγεται τυχαία ένα δεδομένο R_i. Ωστόσο, στη συνέχεια αναζητούνται οι k κοντινότεροι γείτονες της ίδιας κλάσης, που ονομάζονται nearest hits H_j και οι k κοντινότεροι γείτονες για καθεμία από τις υπόλοιπες διαφορετικές κλάσεις, που ονομάζονται nearest misses M_j (C). Στη συνέχεια ανανεώνει τα βάρη W[A] όλων των χαρακτηριστικών Α. Παρακάτω παρουσιάζεται αναλυτικά ο αλγόριθμος ReliefF.

Algorithm ReliefF [71]
Input: for each training instance a vector of attribute values and the class value Output: the vector W of estimations of the qualities of attributes

1. set all weights W[A]:=0.0;
2. for i:=1 to m do begin
3. randomly select an instance R_i;
4. find k nearest hits H_j;
5. for each class $C \notin class(R_i)$ do
6. from class C find k nearest misses $M_j(C)$
7. for A:=1 to α do
8.

$$W[A] := W[A] - \sum_{j=1}^{k} \frac{diff(A, R_i, H_j)}{m \cdot k} + \sum_{C \notin class(R_i)} [\frac{P(C)}{1 - P(class(R_i))} \sum_{j=1}^{k} \frac{diff(A, R_i, M_j(C))}{m \cdot k}]$$

9. end

Εάν κάποιο δεδομένο (π.χ. I_1) στο σύνολο εκπαίδευσης έχει άγνωστη τιμή, τότε:

$$diff(A, I_1, I_2) = 1 - P(value(A, I_2)|class(I_1)) \tag{3.61}$$

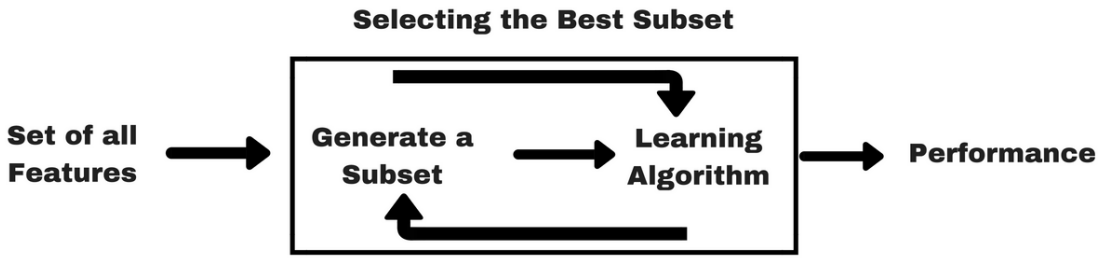
Εάν και τα δύο δεδομένα στο σύνολο εκπαίδευσης έχουν άγνωστη τιμή, τότε:

$$diff(A, I_1, I_2) = 1 - \sum_{V}^{\neq values(A)} (P(V|class(I_1)) \times P(V|class(I_2))) \tag{3.62}$$

Επεκτάσεις του αλγορίθμου Relief αποτελούν οι αλγόριθμοι SURF [72], SURF* [73], MultiSURF, MultiSURF* [74] & TURF [75].

3.8.2 Μέθοδοι περιτυλίγματος (Wrapper Methods)

Χρησιμοποιεί κάποιον αλγόριθμο μηχανικής μάθησης, προκειμένου να βρει το υποσύνολο των χαρακτηριστικών, που θα έχει την καλύτερη απόδοση στον αλγόριθμο αυτό.



Εικόνα 3.26: Wrapper Methods

- **Recursive feature elimination:** Στην τεχνική αυτή χρησιμοποιείται ένα σύνολο αλγορίθμων, όπου τα βάρη και η σημασία των χαρακτηριστικών υποδηλώνουν τη σπουδαιότητα των χαρακτηριστικών αυτών στην τελική πρόβλεψη. Σε κάθε επανάληψη, ένας καθορισμένος αριθμός χαρακτηριστικών αφαιρείται βάσει της κατάταξης των βαρών ή της σημασίας τους, έως ότου να μείνει ο απαιτούμενος αριθμός χαρακτηριστικών. Οι Guyon & Weston et al. [76] ήταν εκείνοι, που πρότειναν πρώτοι τον συνδυασμό της τεχνικής RFE με Μηχανές Διανυσμάτων Υποστήριξης (SVMs), προκειμένου να βρουν το βέλτιστο υποσύνολο χαρακτηριστικών. Συνέκριναν τα αποτελέσματα της έρευνάς τους με εκείνα των Golub et al. [77], και συμπέραναν ότι τα χαρακτηριστικά που επιλέχθηκαν με τη μέθοδο RFE-SVM είναι καλύτερα από εκείνα των Golub et al. ασχέτως των ταξινομητών, που χρησιμοποιήθηκαν. Η διαδικασία της μεθόδου RFE περιγράφεται στα ακόλουθα βήματα:
  - Train the classifier (optimize the weights w_i with respect to J).
  - Compute the ranking criterion for all features (DJ(i) or $(w_i)^2$).
  - Remove the feature with smallest ranking criterion.
- **Sequential Forward Selection (SFS):** Στην περίπτωση αυτή, ξεκινάμε με κενό σύνολο χαρακτηριστικών, προσθέτουμε ένα χαρακτηριστικό κάθε φορά, το οποίο μεγιστοποιεί την αντικειμενική συνάρτηση $J(Y_k + x^+)$ όταν συνδυαστεί με τα χαρακτηριστικά Y_k, που έχουν ήδη επιλεχθεί. Κρατάμε τα χαρακτηριστικά εκείνα με τη μικρότερη p-value. Η διαδικασία αυτή περιγράφεται στα ακόλουθα βήματα:
  1. Start with the empty set $Y_0 = \%0$
  2. Start the next feature $x^+ = \text{argmax}_{x \notin Y_k} J(Y_k + x)$
  3. Update $Y_{k+1} = Y_k + x^+ \quad ; k = k + 1$
  4. Go to 2

Το κύριο μειονέκτημα της μεθόδου αυτής είναι το γεγονός ότι δεν είναι εφικτή η αφαίρεση χαρακτηριστικών, που δεν είναι τόσο σημαντικά σε συνδυασμό με την προσθήκη άλλων χαρακτηριστικών.

- **Sequential Backward Selection (SBS):** Στην περίπτωση αυτή, το μοντέλο μας ξεκινά να εκπαιδεύεται με όλα τα χαρακτηριστικά και σε κάθε επανάληψη αφαιρείται το χαρακτηριστικό εκείνο που συνεισφέρει λιγότερο στην τελική πρόβλεψη, δηλαδή αυτό με τη μεγαλύτερη p-value. Η διαδικασία αυτή περιγράφεται στα ακόλουθα βήματα:

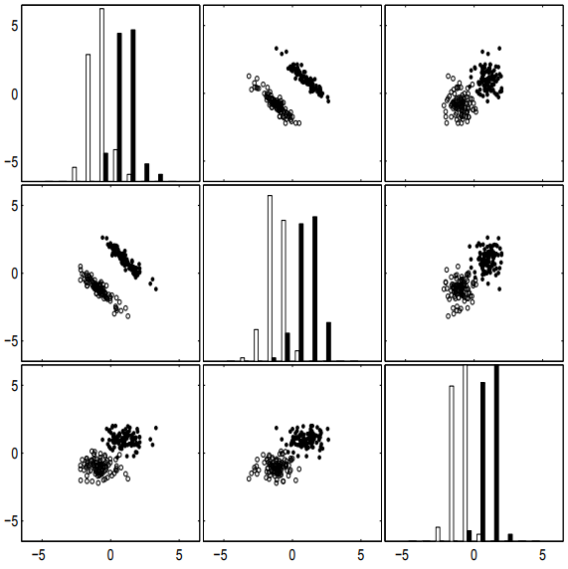
  1. Start with the full set $Y_0 = X$
  2. Remove the worst feature $x^- = \text{argmax}_{x \in Y_k} J(Y_k - x)$
  3. Update $Y_{k+1} = Y_k - x^-$; $k = k + 1$
  4. Go to 2

  Το κύριο μειονέκτημα της μεθόδου αυτής είναι η αδυναμία να επαναξιολογήσει τη χρησιμότητα ενός χαρακτηριστικού, όταν αυτό έχει αφαιρεθεί.

  Άλλες μέθοδοι, που στηρίζονται στις SFS & SBS, είναι οι εξής:

  – **Plus-L minus-R selection (LRS): Αποτελεί γενίκευση των SFS & SBS.**
  – **Bidirectional Search (BDS): Πρόκειται για μία παράλληλη εφαρμογή των SFS & SBS.**
  – **Sequential floating selection: Αποτελεί επέκταση της μεθόδου LRS.** Περιλαμβάνει:
    * **Sequential floating forward selection (SFFS)**
    * **Sequential floating backward selection (SFBS)**

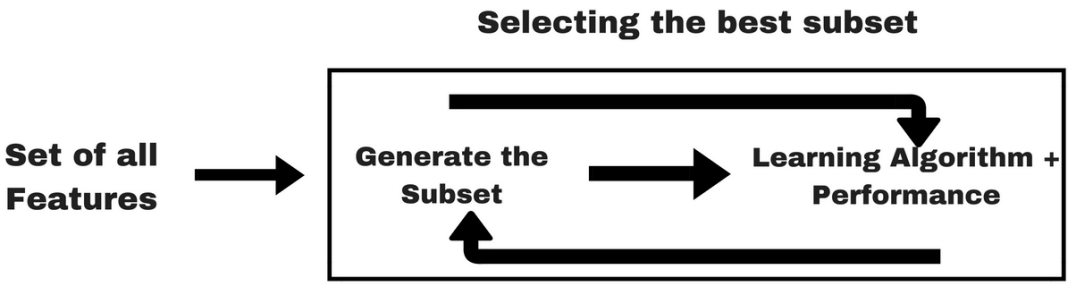
- **Forward vs. Backward Selection:** Η μέθοδος του forward selection είναι υπολογιστικά περισσότερο αποδοτική από αυτή του backward elimination. Ωστόσο, κατά τη μέθοδο του forward selection παρουσιάζεται συχνά το φαινόμενο να μη βρίσκονται τα κατάλληλα υποσύνολα χαρακτηριστικών, καθώς η σημασία των χαρακτηριστικών δεν αξιολογείται με το περιεχόμενο των υπόλοιπων χαρακτηριστικών, που δεν έχουν συμπεριληφθεί ακόμα. Παράδειγμα αποτελεί η παρακάτω εικόνα, στην οποία έχουμε τρία χαρακτηριστικά. Όπως βλέπουμε στο κάτω δεξιά ιστόγραμμα, το τρίτο χαρακτηριστικό διαχωρίζει τις δύο κλάσεις καλύτερα από τα άλλα δύο χαρακτηριστικά, με αποτέλεσμα να προτιμηθεί στη μέθοδο του forward selection. Ωστόσο, εάν συνδυαστεί με ένα από τα άλλα δύο χαρακτηριστικά, δεν διαχωρίζει ικανοποιητικά τις δύο κλάσεις. Αντίθετα, αν συνδυαστούν τα πρώτα δύο χαρακτηριστικά μεταξύ τους, τα οποία όμως έχουν απορριφθεί κατά το forward selection, τότε πετυχαίνουν πολύ καλό διαχωρισμό των δύο κλάσεων. Στην περίπτωση αυτή, η μέθοδος του backward selection θα κρατούσε τα πρώτα δύο χαρακτηριστικά. Ωστόσο, εάν για κάποιο λόγο θέλαμε να κρατήσουμε μόνο ένα χαρακτηριστικό, τότε η μέθοδος του backward selection δεν θα ήταν καλή επιλογή, καθώς θα είχε απορρίψει το χαρακτηριστικό αυτό, που διαχωρίζει καλά τις δύο κλάσεις μόνο του, αλλά δεν πετυχαίνει την ίδια απόδοση, αν συνδυαστεί με τα υπόλοιπα χαρακτηριστικά.



Εικόνα 3.27: Forward vs. Backward selection

3.8.3 Ενσωματωμένες Μέθοδοι (Embedded Methods)

Αποτελεί συνδυασμό των δύο προηγούμενων μεθόδων. Υλοποιείται με αλγορίθμους, που έχουν ενσωματωμένες τις μεθόδους της επιλογής των χαρακτηριστικών ως μέρος της διαδικασίας εκπαίδευσης τους. Συμπεραίνουν τη σημασία των χαρακτηριστικών, δηλαδή το πόσο συνεισέφεραν στην τελική πρόβλεψη. Συγκλίνουν γρηγορότερα στη λύση συγκριτικά με άλλες τεχνικές επιλογής χαρακτηριστικών. Παραδείγματα αποτελούν οι Lasso και Ridge Regression, που χρησιμοποιούν την L1 και L2 εξομάλυνση αντίστοιχα, προθέτοντας όρους ποινής στη συνάρτηση κόστους, όπως είδαμε στην ενότητα . Άλλο παράδειγμα αποτελούν τα Τυχαία Δάση και τα Δέντρα Απόφασης, όπως τα CART [78].



Εικόνα 3.28: Embedded Methods

Κεφάλαιο 4

Επεξεργασία Φυσικής Γλώσσας (Natural Language Processing - NLP)

Όπως είδαμε στο Κεφάλαιο 3, η είσοδος σε ένα νευρωνικό δίκτυο γίνεται με τη μορφή διανυσμάτων. Επομένως, το κείμενο απαιτείται να υποστεί κάποια επεξεργασία, προκειμένου να μετατραπεί σε μία μορφή κατανοητή από το νευρωνικό δίκτυο. Έτσι, το Κεφάλαιο αυτό πραγματεύεται κάποιες βασικές μεθόδους τεχνικές επεξεργασίας και μετατροπής κειμένου, που εμπίπτουν στον τομέα της Επεξεργασίας Φυσικής Γλώσσας. Συγχρόνως, αναλύονται τρόποι εύρεσης της ομοιότητας μεταξύ των λέξεων - φράσεων. Το Κεφάλαιο ολοκληρώνεται με την ανάλυση μεθόδων εύρεσης των κυριότερων topics σε ένα κείμενο.

4.1 Βασικές Αρχές της Επεξεργασίας Φυσικής Γλώσσας

Η επεξεργασία φυσικής γλώσσας αποτελεί έναν υποκλάδο της επιστήμης υπολογιστών, της γλωσσολογίας και της τεχνητής νοημοσύνης. Αφορά την αλληλεπίδραση μεταξύ των υπολογιστικών συστημάτων και των φυσικών γλωσσών, προγραμματίζοντάς τα, έτσι ώστε να επεξεργάζονται και να αναλύουν μεγάλο όγκο δεδομένων φυσικής γλώσσας. Η επεξεργασία φυσικής γλώσσας εμπεριέχει την αναγνώριση ομιλίας, την κατανόηση και παραγωγή φυσικής γλώσσας.

Στη συνέχεια, θα παρουσιαστούν ορισμένες τεχνικές επεξεργασίας και μετατροπής κειμένου, που χρησιμοποιήθηκαν και στα πειράματά μας.

4.1.1 Bag of words

Στην τεχνική αυτή, κάθε κείμενο αναπαρίσταται ως το σύνολο των λέξεων, που το αποτελούν, αγνοώντας τη γραμματική και τη σειρά των λέξεων. Ουσιαστικά, δημιουργείται ένα dictionary, που ως key έχει την κάθε λέξη και ως value, το πλήθος των φορών, που η λέξη αυτή εμφανίζεται στο κείμενο.

Έστω ότι δίνονται τα παρακάτω κείμενα και ζητείται η BoW αναπαράστασή τους:

1. John likes to watch movies. Mary too likes movies.
2. Mary also likes to watch football games.

Για κάθε κείμενο κατασκευάζουμε μία λίστα με τις μοναδικές λέξεις, οπότε προκύπτουν οι εξής λίστες:

1. BoW1 = "John": 1, "likes": 2, "to": 1, "watch": 1, "movies": 2, "Mary": 1, "to": 1
2. BoW2 = "also": 1, "Mary": 1, "likes": 1, "watch": 1, "to": 1,"football": 1, "games": 1

Επομένως, προκύπτουν οι παρακάτω BoW αναπαραστάσεις, στις οποίες όπως φαίνεται η σειρά των στοιχείων (keys) είναι ελεύθερη:

1. BoW1 = "John": 1, "likes": 2, "to": 1, "watch": 1, "movies": 2, "Mary": 1, "to": 1
2. BoW2 = "also": 1, "Mary": 1, "likes": 1, "watch": 1, "to": 1,"football": 1, "games": 1

4.1.2 Term Frequency - Inverse Document Frequency (TF - IDF)

Στην τεχνική αυτή, δηλώνεται πόσο σημαντική είναι μία λέξη σε ένα κείμενο ή σε ένα σύνολο κειμένων. Έτσι, λέξεις που εμφανίζονται συχνά, όπως the,a,to etc. έχουν μικρότερη βαρύτητα. Η μέθοδος tf-idf είναι ίση με το γινόμενο της συχνότητας εμφάνισης ενός όρου στο κείμενο (term frequency) επί την αντίστροφη συχνότητα εμφάνισης του όρου αυτού (inverse document frequency). Συγκεκριμένα:

$$tfidf(t, d, D) = tf(t, d) \cdot idf(t, D) \tag{4.1}$$

$$tf(t,d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}} \tag{4.2}$$

$$idf(t,D) = log \frac{N}{|d \in D : t \in d|} \tag{4.3}$$

όπου:

- $f_{t,d}$: ο αριθμός των φορών, που εμφανίζεται ο όρος t στο κείμενο d
- $\sum_{t' \in d} f_{t',d}$: το συνολικό πλήθος των λέξεων στο κείμενο d
- N:πλήθος κειμένων
- $|d \in D : t \in d|$: αριθμός κειμένων,όπου εμφανίζεται ο όρος t. Αν ο όρος t δεν υπάρχει στα κείμενα, τότε στον παρονομαστή προστίθεται το 1, προκειμένου να αποφευχθεί η διαίρεση με το 0.

Έστω ότι μάς δίνονται τα εξής δύο κείμενα:

Document 1		Document 2	
Term	Term Count	Term	Term Count
this	1	this	1
is	1	is	1
a	2	another	2
sample	1	example	3

Πίνακας 4.1: Παράδειγμα Tf-idf

- tf("this",d_1)=$\frac{1}{5} = 0.2$
- tf("this",d_2)=$\frac{1}{7} \approx 0.14$
- tf("this",D)=$log(\frac{2}{2}) = 0$

Οπότε:

- tfidf("this",d_1, D)=$0.2 \times 0 = 0$
- tfidf("this",d_2, D)=$0.14 \times 0 = 0$

Με παρόμοιο τρόπο υπολογίζουμε το tf-idf και για τις υπόλοιπες λέξεις.

4.1.3 N-grams

Η έννοια των n-grams χρησιμοποιείται ευρέως κατά την εξαγωγή χαρακτηριστικών κειμένου. Αποτελεί γενίκευση των δύο τεχνικών (BoW & Tf-idf), που αναφέρθηκαν προηγουμένως, με την έννοια ότι μας επιτρέπεται να χρησιμοποιήσουμε ακολουθίες δύο, τριών ή και παραπάνω λέξεων ή χαρακτήρων σαν όρους. Συγκεκριμένα, στην περίπτωση που εξάγουμε n χαρακτήρες, τα n-grams ονομάζονται n-grams χαρακτήρων (character n-grams), ενώ στην περίπτωση που εξάγουμε n λέξεις, ονομάζονται n-grams λέξεων (word n-grams). Η ακολουθία δύο λέξεων/χαρακτήρων bigram, η ακολουθία τριών λέξεων/χαρακτήρων trigram κ.ο.κ. Επίσης, ένας απλός χαρακτήρας ή μία απλή λέξη καλείται unigram (n=1). Παρακάτω δίνονται δύο παραδείγματα εξαγωγής n-grams (n=2), όπου το πρώτο παράδειγμα αντιστοιχεί σε εξαγωγή χαρακτήρων χωρίς να υπάρχει η δυνατότητα εξαγωγής λέξεων, ενώ το δεύτερο παράδειγμα αντιστοιχεί σε εξαγωγή λέξεων, όπου υπάρχει και η δυνατότητα εξαγωγής χαρακτήρων.

Παράδειγμα 1: Έστω ότι δίνεται η ακολουθία 'abcdef'.

Κατά την εξαγωγή 2-grams από την παραπάνω ακολουθία προκύπτουν οι όροι: *'ab','bc','cd','de','ef'*

Παράδειγμα 2: Έστω ότι δίνεται η ακολουθία 'This is a great book'.

Κατά την εξαγωγή 2-grams από την παραπάνω ακολουθία προκύπτουν οι όροι: *'This is', 'is a', 'a great','great book'*

4.1.4 Λεκτική Ανάλυση (Tokenization)

Στην διαδικασία αυτή ένα κείμενο, μετατρέπεται από μία ακολουθία χαρακτήρων, σε μία ακολουθία από λεκτικές μονάδες ή όρους (tokens ή terms), όπως λέξεις, σημεία στίξης, αριθμούς κλπ. Αυτό το βήμα είναι απαραίτητο για την εκτέλεση των υπολοίπων, καθώς ενεργούν πάνω στους όρους του κειμένου.

4.1.5 Αποκατάληξη (Stemming)

Η αποκατάληξη (stemming) αποτελεί μία διαδικασία, κατά την οποία αφαιρούνται από τις λέξεις οι καταλήξεις, ώστε να εντοπιστεί η ρίζα της καθεμίας με στόχο τη μείωση της πολυπλοκότητας της ανάλυσης χωρίς απώλεια σημαντικής πληροφορίας και κατ' επέκταση την ικανότητα των συστημάτων μηχανικής μάθησης να είναι λιγότερο επιρρεπή στις μορφολογικές διαφορές των λέξεων. Πρέπει να τονιστεί ότι το αποτέλεσμα της αποκατάληξης μίας λέξης δεν είναι πάντα κανονική λέξη. Στον πίνακα, που ακολουθεί, παρουσιάζονται παραδείγματα αποκατάληξης.

Λέξεις	**Αποκατάληξη (Stem)**
Consult Consultant Consulting Consultantative Consultants Consulting	consult

Πίνακας 4.2: Παράδειγμα αποκατάληξης (stemming) λέξεων

4.1.6 Λημματοποίηση (Lemmatization)

Κατά τη λημματοποίηση (lemmatization), οι διάφορες μορφές μίας λέξης (κλίση, παράγωγα) αντιστοιχούνται στη ρίζα τους. Σε αντίθεση με τη διαδικασία του stemming, η διαδικασία του lemmatization απαιτεί γνώση του τι μέρος του λόγου είναι η λέξη. Επίσης, το λήμμα μίας λέξης είναι πάντα μία κανονική λέξη. Στον πίνακα, που ακολουθεί, γίνεται σύγκριση μεταξύ των stemming & lemmatization.

Λέξεις	**Αποκατάληξη (Stem)**	**Λημματοποίηση (Lemmatization)**
was	wa	be
studies	studi	study
studying	study	study
having	hav	have

Πίνακας 4.3: Σύκριση stemming & lemmatization

4.1.7 Διανύσματα Λέξεων (Word Embeddings)

Τα word embeddings αποτελούν μία τεχνική, που εμπίπτει στο πεδίο της Επεξεργασίας Φυσικής Γλώσσας (NLP), όπου λέξεις αντιστοιχίζονται σε πίνακες πραγματικών αριθμών διαστάσεων (25-250). Σκοπός τους είναι παρόμοιες σημασιολογικά λέξεις να αντιστοιχίζονται σε κοντινά σημεία στον ίδιο διανυσματικό χώρο [79]. Γνωστές μέθοδοι Διανυσμάτων Λέξεων αποτελούν το Word2Vec, GloVE και BERT. Στη συνέχεια, αναλύουμε τη μέθοδο GloVE, την οποία και χρησιμοποιήσαμε στην εργασία μας.

4.1.7.1 GloVE (Global Vectors for Word Represantation)

Η μέθοδος GloVE [80] πρόκειται για μία συλλογή από προεκπαιδευμένες αναπαραστάσεις λέξεων, η οποία εκπαιδεύτηκε σε ένα corpus λέξεων. Είναι ένας αλγόριθμος μη επιβλεπόμενης μάθησης, που χρησιμοποιείται για τη δημιουργία διανυσματικών αναπαραστάσεων των λέξεων. Το μοντέλο αυτό μαθαίνει διανύσματα λέξεων εξετάζοντας λέξεις, που συνεμφανίζονται μέσα στο σώμα ενός κειμένου. Οι αναπαραστάσεις, που προκύπτουν εκδηλώνουν χρήσιμες ιδιότητες των λέξεων, όπως σχέσεις αναλογίας ή σχέσεις σημασιολογικής συγγένειας.

Αρχικά, ορίζεται ο πίνακας X, του οποίου τα στοιχεία X_{ij} δηλώνουν τον αριθμό των φορών, που η λέξη j εμφανίζεται σε μία γειτονιά της λέξης i. Ορίζεται, επίσης, ο πίνακας $X_i = \sum_k X_{ik}$ ως ο αριθμός των φορών, που κάθε λέξη εμφανίζεται σε μία γειτονιά της λέξης i. Τέλος, ορίζεται ως $P_{ij} = P(i|j) = \frac{X_{ij}}{X_i}$ η πιθανότητα η λέξη j να εμφανίζεται σε μία γειτονιά της λέξης i.

Ας υποθέσουμε τις λέξεις i=ice & j=steam. Θέλουμε να κατασκευάσουμε διανυσματικές αναπαραστάσεις για τις δύο αυτές λέξεις, οι οποίες ναι μεν να είναι κοντά, αλλά ταυτόχρονα να εμπεριέχουν τη σχέση που έχουν οι δύο λέξεις μεταξύ τους. Προκειμένου, λοιπόν, να εξεταστεί η σχέση των δύο λέξεων, προτείνεται η μελέτη του λόγου $\frac{P_{ik}}{P_{jk}}$ για διάφορες λέξεις k στο σώμα του κειμένου. Εάν οι λέξεις k σχετίζονται με τη λέξη ice αλλά όχι με τη λέξη steam, αναμένουμε ο λόγος $\frac{P_{ik}}{P_{jk}}$ να είναι μεγάλος. Για λέξεις k, που σχετίζονται και με τις δύο λέξεις i,j αναμένουμε ο λόγος να είναι περίπου ίσος με 1. Αυτό γίνεται προφανές με τη βοήθεια του πίνακα, που ακολουθεί.

Probability and Ratio	K=solid	K=gas	K = water	K = fashion
P(k|ice)	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
P(k|steam)	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$\frac{P(k\|ice)}{P(k\|steam)}$	8.9	8.5×10^{-2}	1.36	0.96

Πίνακας 4.4: GloVE

Το γενικό μαθηματικό μοντέλο παίρνει την ακόλουθη μορφή:

$$F(w_i, w_j, \tilde{w_k}) = \frac{P_{ik}}{P_{jk}} \tag{4.4}$$

Σημειώνεται ότι ο λόγος $\frac{P_{ik}}{P_{jk}}$ εξαρτάται από τρεις λέξεις i,j,k.

Για την ενσωμάτωση της πληροφορίας $\frac{P_{ik}}{P_{jk}}$ στον χώρο των διανυσματικών αναπαραστάσεων των λέξεων, μία λύση αποτελεί η διαφορά των διανυσμάτων. Έτσι, το επόμενο βήμα είναι η αναζήτηση μίας συνάρτησης, έτσι ώστε:

$$F(w_i - w_j, \tilde{w_k}) = \frac{P_{ik}}{P_{jk}} \tag{4.5}$$

Επειδή τα ορίσματα της F είναι διανύσματα (πίνακες), ενώ το δεξί μέλος της εξίσωσης είναι βαθμωτό, καταλήγουμε στη σχέση:

$$F((w_i - w_j)^T \tilde{w_k}) = \frac{P_{ik}}{P_{jk}} \tag{4.6}$$

Για πίνακες συνεμφάνισης λέξεων, η διάκριση μεταξύ μίας λέξης και της γειτονικής της είναι αυθαίρετη, οπότε χρησιμοποιούνται οι εναλλαγές . Προτείνεται, έτσι, η παρακάτω συνάρτηση σαν ομομορφισμός:

$$F((w_i - w_j)^T \tilde{w_k}) = \frac{F(w_i^T \tilde{w_k})}{F(w_j^T \tilde{w_k})} \tag{4.7}$$

Η λύση της εξίσωσης είναι:

$$F = exp \rightarrow w_i^T \tilde{w_k} = log(P_{ik}) = log(X_{ik}) - log(X_i) \tag{4.8}$$

Ο παράγοντας $log(X_i)$ μπορεί να ενσωματωθεί σε μία πόλωση b_i για τη λέξη i, καθώς είναι ανεξάρτητος του k. Προστίθεται, ακόμα, και η πόλωση $\tilde{b_k}$ για τη λέξη k για λόγους συμμετρίας, οπότε καταλήγουμε στη σχέση:

$$w_i^T \tilde{w_k} + b_i + \tilde{b_k} = log(X_{ik}) \tag{4.9}$$

Ελαχιστοποιούμε, λοιπόν, το κριτήριο J, το οποίο υπολογίζει το άθροισμα των τετραγώνων των σφαλμάτων βασισμένο στην παραπάνω σχέση.

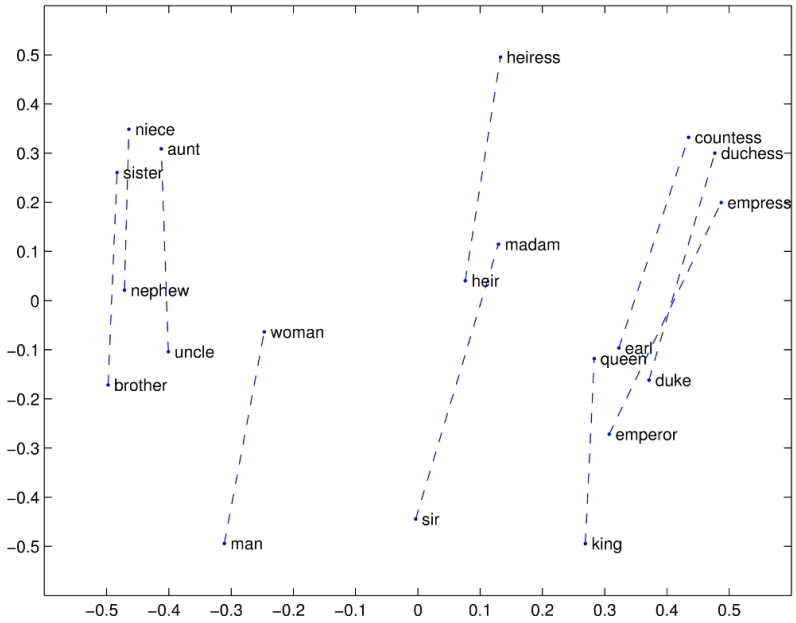
$$J = \sum_{i}^{V} \sum_{j}^{V} f(X_{ij})(w_i^T \tilde{w}_j + b_i + \tilde{b_j} - log(X_{ij})) \tag{4.10}$$

Η συνάρτηση f λειτουργεί ως βάρος στην αντικειμενική συνάρτηση, έχοντας ως ρόλο να προστατέψει την αλλοίωσή της από ζευγάρια λέξεων με μεγάλη συχνότητα συνεμφάνισης.

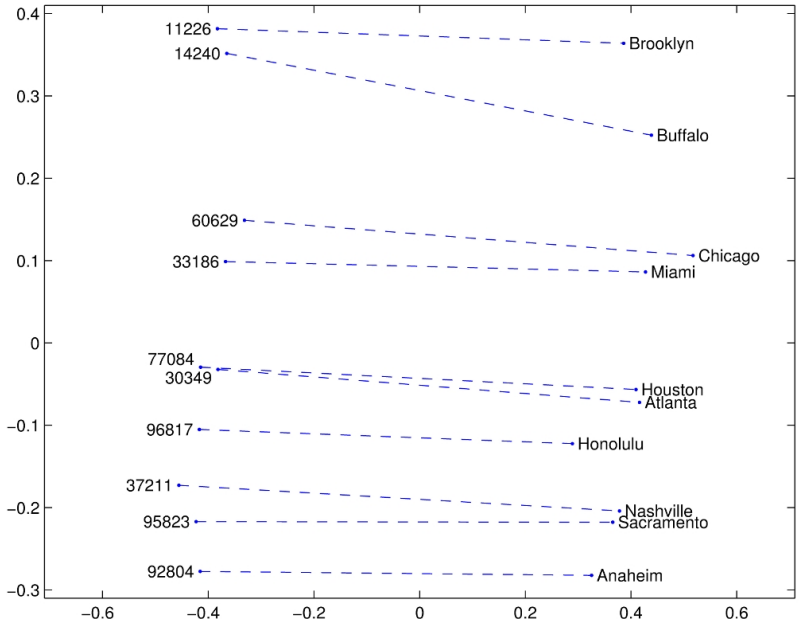
Επιλέγεται, επομένως, η εξής συνάρτηση:

$$f(x) = \begin{cases} (\frac{x}{x_{max}})^a, & x < x_{max} \\ 1, & otherwise \end{cases} \tag{4.11}$$

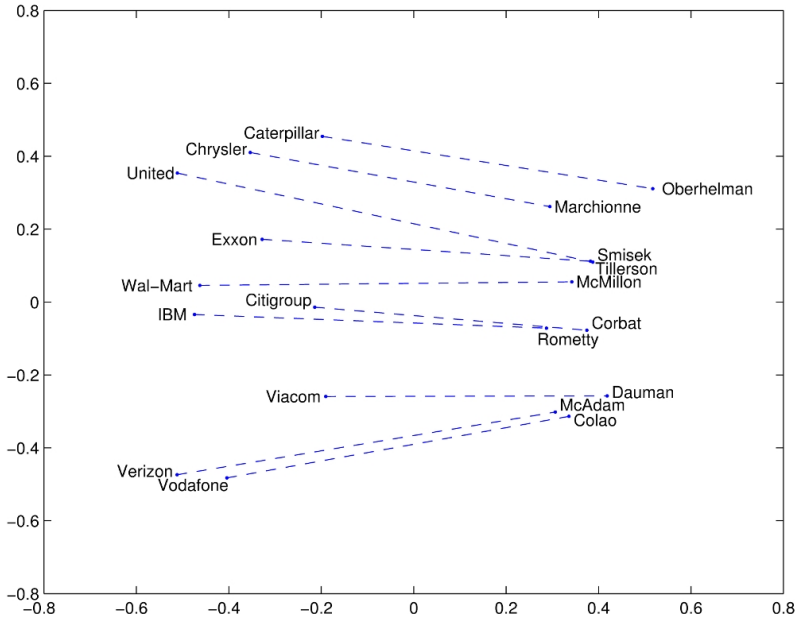
Παρακάτω, παρουσιάζονται αποτελέσματα αναπαράστασης της μεθόδου GloVE.



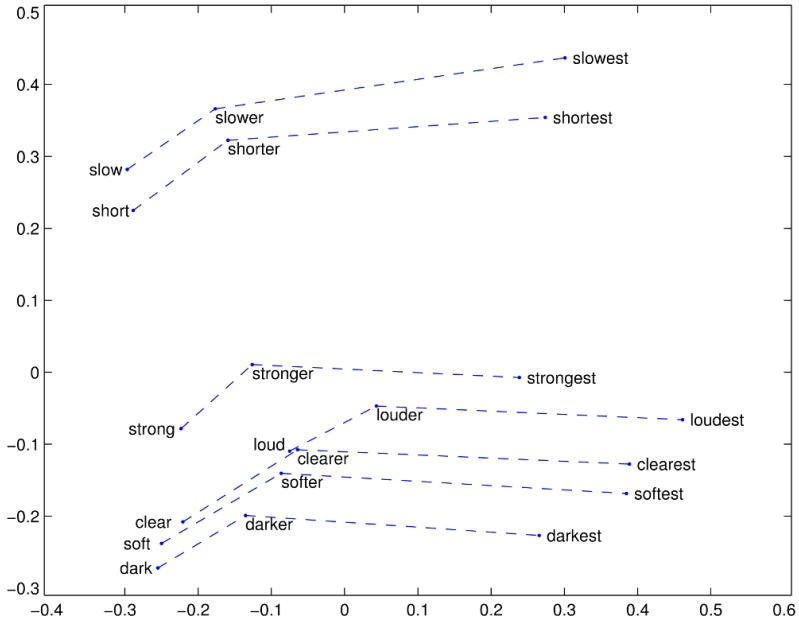
Εικόνα 4.1: Σχέση Φύλου



Εικόνα 4.2: Σχέση πόλης-ταχυδρομικού κώδικα



Εικόνα 4.3: Σχέση εταιρείας-ceo



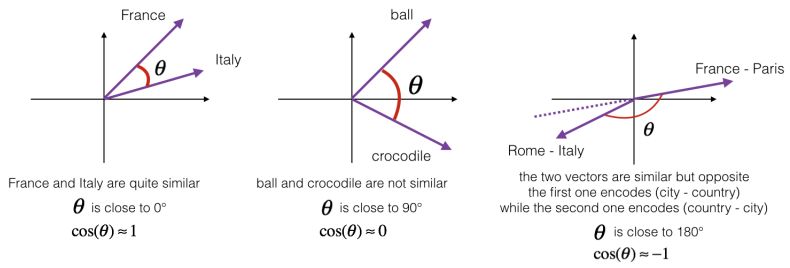
Εικόνα 4.4: Σχέση επιθέτου-συγκριτικού-υπερθετικού βαθμού

4.2 Ομοιότητα μεταξύ Λέξεων

Έχοντας αντικαταστήσει τις λέξεις με τις διανυσματικές τους αναπαραστάσεις στον χώρο, μπορούμε να βρούμε την ομοιότητα μεταξύ των διανυσμάτων των λέξεων ή φράσεων. Παρακάτω παρουσιάζονται ορισμένες από τις μεθόδους, που χρησιμοποιούνται.

- **Cosine Similarity:** Η ομοιότητα συνημίτονου είναι ένας δείκτης συνάφειας ανάμεσα σε δυο διανύσματα, βασισμένος στη γωνία ανάμεσα στα διανύσματα στο χώρο χαρακτηριστικών τους. Η τιμή του κυμαίνεται στο διάστημα [-1,1], καθώς όπως δηλώνει και το όνομά του είναι ουσιαστικά ένα συνημίτονο. Είναι, επομένως, ένας δείκτης φοράς και όχι πλάτους ή μεγέθους. Δύο διανύσματα με ίδια φορά έχουν δείκτη ομοιότητας συνημίτονου ίσο με 1, δύο διανύσματα κάθετα μεταξύ τους (δηλαδή με γωνία 90°) έχουν δείκτη ομοιότητας συνημίτονου ίσο με 0, και τέλος δύο διανύσματα διαμετρικά αντίθετα (δηλαδή με γωνία 180°) έχουν δείκτη ομοιότητας συνημίτονου ίσο με -1. Δίνεται από τον παρακάτω τύπο:

$$cosine\ similarity = \cos(\vartheta) = \frac{\vec{v_1} \cdot \vec{v_2}}{\|\vec{v_1}\| \cdot \|\vec{v_2}\|} \tag{4.12}$$



Εικόνα 4.5: Ομοιότητα Συνημιτόνου (Cosine Similarity)

- **Ευκλείδεια απόσταση (Euclidean distance)**

$$d(v_1, v_2) = \sqrt{\sum_{i=1}^{N}(v_{1i} - v_{2i})^2} \tag{4.13}$$

- **Word Mover's Distance (WMD)**

  Οι Kusner et al. [81] ορίζουν την απόσταση μεταξύ δύο κειμένων ως το ελάχιστο κόστος μετατροπής των λέξεων του ενός κειμένου σε λέξεις του άλλου, όπου το κόστος υπολογίζεται ως η απόσταση μεταξύ διανυσματικών αναπαραστάσεων των λέξεων, όπως θα δούμε στη συνέχεια. Η μέθοδος αυτή αποτελεί περίπτωση της Earth Mover's Distance και χρησιμοποιεί τις έννοιες των word embeddings & bag of words. Συγκεκριμένα, ορίζεται ο πίνακας $X \in R^{d \times n}$, ο οποίος περιέχει n λέξεις (στήλες). Έτσι, η i-οστή στήλη $x_i \in R^d$ αποτελεί την αναπαράσταση της i-οστής λέξης στο χώρο των d διαστάσεων. Στη συνέχεια, υποθέτουμε ότι κάθε κείμενο ορίζεται ως ένα διάνυσμα $d \in R^n$ με την τεχνική του bag of words. Δηλαδή, εάν η λέξη i εμφανίζεται c_i φορές στο κείμενο d, τότε ορίζουμε ως $d_i = \frac{c_i}{\sum_{j=1}^{n} c_j}$.

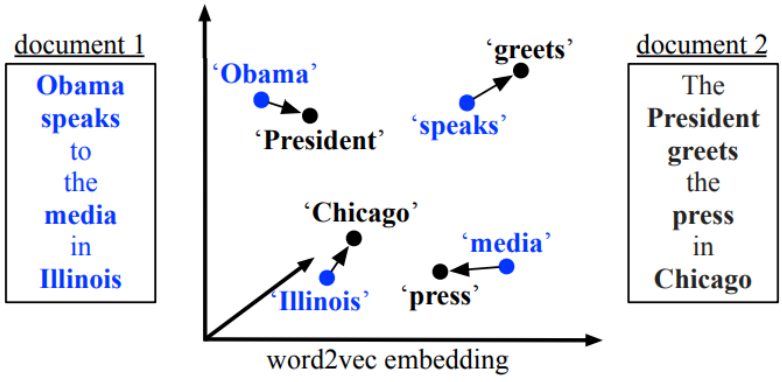
  Έπειτα, ορίζουμε το κόστος μετατροπής των λέξεων του ενός κειμένου σε λέξεις του άλλου ως την απόσταση μεταξύ διανυσματικών αναπαραστάσεων των λέξεων i & j και χρησιμοποιούμε τον ορισμό της ευκλείδειας απόστασης, δηλαδή: $c(i,j) = \|x_i - x_j\|_2$

  Η WMD μεταξύ των κειμένων D & D' ορίζεται ως η τιμή, που προκύπτει από τη λύση του ακόλουθου προβλήματος ελαχιστοποίησης:

$$\min_{T \geq 0} T_{ij} c(i,j) \tag{4.14}$$

$$subject\ to: \sum_{j=1}^{n} T_{ij} = d_i, \forall i \in 1, ..., n \tag{4.15}$$

$$\sum_{i=1}^{n} T_{ij} = d'_j, \forall j \in 1, ..., n \tag{4.16}$$



Εικόνα 4.6: Word Mover's Distance

- **Tanimoto:**

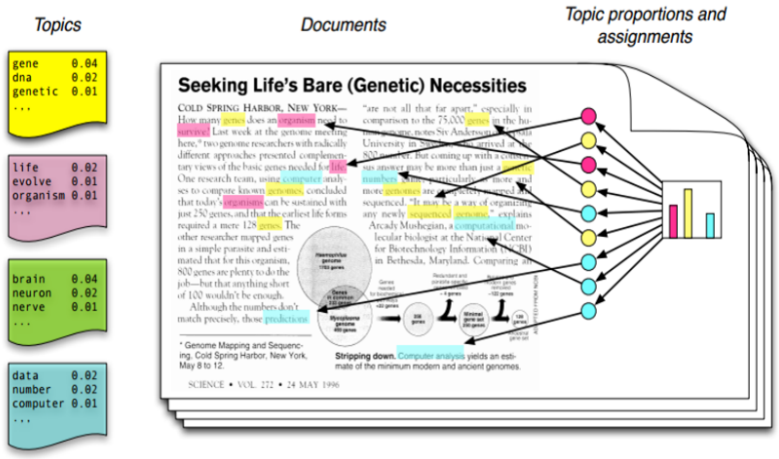
$$T(\vec{v_1}, \vec{v_2}) = \frac{\vec{v_1} \cdot \vec{v_2}}{\|\vec{v_1}\|^2 + \|\vec{v_2}\|^2 - \vec{v_1} \cdot \vec{v_2}} \tag{4.17}$$

4.3 Topic Modeling

Topic Modeling είναι η διαδικασία εύρεσης των topics σε ένα σύνολο κειμένων. Ένα κείμενο αποτελείται από πολλαπλά topics σε διαφορετικές αναλογίες. Τα topics, που παράγονται με διάφορες τεχνικές topic modeling, αποτελούν συστάδες παρόμοιων λέξεων. Η διαδικασία του topic modeling μπορεί να είναι χρήσιμη για μηχανές αναζήτησης, αυτόματη εξυπηρέτηση πελατών και κάθε άλλο παράδειγμα, όπου η γνώση των topics σε κείμενα είναι σημαντική. Μία από τις μεθόδους εξαγωγής των topics είναι η μέθοδος Latent Dirichlet Allocation (LDA), που θα εξηγηθεί παρακάτω.

4.3.1 Latent Dirichlet Allocation (LDA)

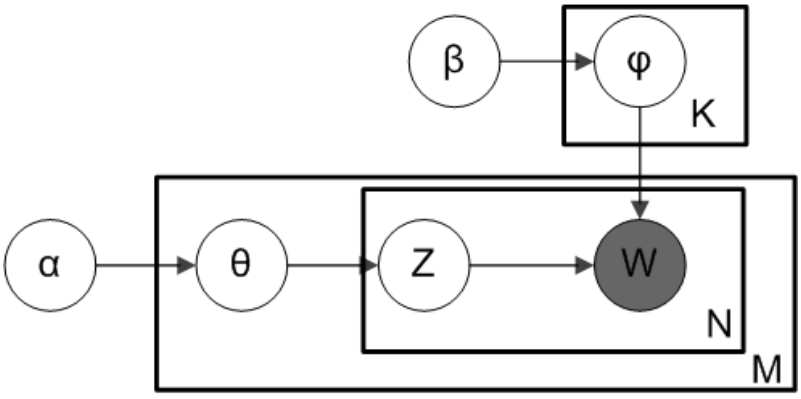
Η μέθοδος LDA [82] αποτελεί ένα γεννητικό πιθανοτικό μοντέλο σε μία συλλογή κειμένων. Η βασική ιδέα είναι ότι τα κείμενα αναπαρίστανται ως τυχαίες μίξεις διαφόρων κρυφών θεμάτων (latent topics), όπου κάθε topic ορίζεται ως μία κατανομή λέξεων πάνω σε ένα προκαθορισμένο λεξιλόγιο, όπως φαίνεται στην εικόνα.



Εικόνα 4.7: Κατανομή θεμάτων (topics) ανά κείμενο

Η μέθοδος LDA υποθέτει την ακόλουθη γεννητική διαδικασία για μία συλλογή κειμένων D, που αποτελείται από M κείμενα, το καθένα μήκους N_i.

1. Επίλεξε $\theta_i \sim Dir(\alpha)$, όπου $i \in 1, ..., M$ & $Dir(\alpha)$ είναι μία κατανομή Dirichlet με μία συμμετρική παράμετρο α, που είναι τυπικά αραιή ($\alpha < 1$).
2. Επίλεξε $\varphi_k \sim Dir(\beta)$, όπου $k \in 1, ..., K$ & β μία παράμετρος τυπικά αραιή.
3. Για κάθε θέση λέξεων i & j, όπου $i \in 1, ..., M$ & $j \in 1, ..., N_i$
   (a) Επίλεξε ένα topic $z_{ij} \sim Multinomial(\theta_i)$.
   (b) Επίλεξε μία λέξη $w_{ij} \sim Multinomial(\varphi_{z_{i,j}})$



Εικόνα 4.8: Γραφική αναπαράσταση του LDA model

- M είναι ο αριθμός των κειμένων
- N είναι ο αριθμός των λέξεων σε ένα δεδομένο κείμενο (το κείμενο i έχει N_i λέξεις.
- α είναι η παράμετρος Dirichlet, που δηλώνει την κατανομή των topics ανά κείμενο
- β είναι η παράμετρος Dirichlet, που δηλώνει την κατανομή των λέξεων ανά topic

- θ_i είναι η κατανομή των topics για το κείμενο i
- φ_k είναι η κατανομή των λέξεων για το topic k
- z_{ij} είναι το topic για τη j^{th} λέξη στο κείμενο i
- w_{ij} είναι η συγκεκριμένη λέξη
- K είναι ο αριθμός των topics

Το γεγονός ότι το W στην εικόνα είναι γραμμοσκιασμένο, δηλώνει ότι οι λέξεις w_{ij} αποτελούν μεταβλητές, που μπορούμε να παρατηρήσουμε, σε αντίθεση με τις υπόλοιπες μεταβλητές, που αποτελούν κρυφές δομές (latent).

Κεφάλαιο 5

Περιγραφή Συνόλων Δεδομένων

Στο Κεφάλαιο αυτό, παρουσιάζονται τα σύνολα δεδομένων, που χρησιμοποιήθηκαν για την πραγματοποίηση όλων των πειραμάτων μας, τα οποία θα αναλυθούν στα επόμενα κεφάλαια.

5.1 Σύνολο Δεδομένων Cresci 2017

Για τα πειράματα που περιγράφονται στην παρούσα εργασία, αποκτήσαμε πρόσβαση στο dataset που έκαναν δημόσια διαθέσιμο οι Cresci S. et al. [7]. Ο πίνακας 5.1 αναφέρει τα ονόματα των datasets, μία σύντομη περιγραφή τους, καθώς και τον αριθμό των λογαριασμών και των δημοσιεύσεων, που αυτά περιέχουν. Ο χρόνος αναπαριστά τον μέσο των χρόνων δημιουργίας των λογαριασμών, που ανήκουν στο κάθε dataset.

		statistics		
dataset	**description**	**accounts**	**tweets**	**year**
genuine accounts	verified accounts that are human-operated	3,474	8,377,522	2011
social spambots #1	retweeters of an Italian political candidate	991	1,610,176	2012
social spambots #2	spammers of paid apps for mobile devices	3,457	428,542	2014
social spambots #3	spammers of products on sale at Amazon.com	464	1,418,626	2011
traditional spambots #1	training set of spammers used by Yang et al. in [83]	1,000	145,094	2009
traditional spambots #2	spammers of scam URLs	100	74,957	2014
traditional spambots #3	automated accounts spamming job offers	433	5,794,931	2013
traditional spambots #4	another group of automated accounts spamming job offers	1,128	133,311	2009
fake followers	simple accounts that inflate the number of followers of another account	3,351	196,027	2012
test set #1	mixed set of 50 % genuine accounts + 50 % social spambots #1	1,982	4,061,598	-
test set #2	mixed set of 50 % genuine accounts + 50 % social spambots #3	928	2,628,181	-

Πίνακας 5.1: Dataset Cresci 2017

Συγκεκριμένα, για τα πειράματά μας χρησιμοποιήσαμε τα παρακάτω datasets, τα οποία επεξεργαστήκαμε με τη βιβλιοθήκη ***pandas*** [84] της python.

- Genuine Accounts
- Social spambots #1, #2, #3

Το κάθε dataset περιέχει 2 αρχεία .csv (ένα για τους χρήστες και ένα για τα tweets).

Το dataset των χρηστών περιέχει τις ακόλουθες πληροφορίες σε στήλες:

'id','name','screen_name','statuses_count','followers_count', 'friends_count','favourites_count','listed_count','url','lang', 'time_zone','location','default_profile','default_profile_image', 'geo_enabled','profile_image_url','profile_banner_url', 'profile_use_background_image', 'profile_background_image_url_https', 'profile_text_color', 'profile_image_url_https', 'profile_sidebar_border_color', 'profile_background_tile', 'profile_sidebar_fill_color','profile_background_image_url', 'profile_background_color','profile_link_color','utc_offset', 'is_translator','follow_request_sent','protected','verified', 'notifications','description','contributors_enabled','following', 'created_at','timestamp','crawled_at','updated','test_set_1', 'test_set_2'

Το dataset των tweets περιέχει τις ακόλουθες πληροφορίες σε στήλες:

'id','text','source','user_id','truncated', 'in_reply_to_status_id','in_reply_to_user_id', 'in_reply_to_screen_name','retweeted_status_id','geo','place', 'contributors','retweet_count','reply_count','favorite_count', 'favorited','retweeted','possibly_sensitive','num_hashtags', 'num_urls','num_mentions','created_at','timestamp','crawled_at', 'updated'

5.2 Social Honeypot Dataset

Αυτό το σύνολο δεδομένων [11] αφορά χρήστες του Twitter και συλλέχθηκε από τον Δεκέμβριο 2009 έως τον Αύγουστο 2010. Περιέχει δύο είδη χρηστών, τους content polluters & legitimate users. Συγκεκριμένα, περιέχει 22,223 content polluters με συνολικά 2,353,473 tweets καθώς και έναν αριθμό ακολούθων τους. Ομοίως, περιέχει 19,276 legitimate users με συνολικά 3,259,693 tweets καθώς και έναν αριθμό ακολούθων τους. Για κάθε είδος χρήστη (content polluters & legitimate users) δίνονται τα ακόλουθα αρχεία σε μορφή .txt που περιέχουν τις εξής πληροφορίες για τον κάθε χρήστη:

- **"type_of_user.txt":** Για κάθε χρήστη περιέχει τις εξής πληροφορίες του προφίλ του ανά στήλη: "UserID/tCreatedAt/tCollectedAt/tNumerOfFollowings /tNumberOfFollowers/tNumberOfTweets/tLengthOfScreenName/t LengthOfDescriptionInUserProfile"
- **"type_of_user_followings.txt":** Περιέχει τις στήλες: "UserID/tSeriesOfNumberOfFollowings"
- **type_of_users_tweets.txt":** Περιέχει tweets των χρηστών καθώς και το χρόνο δημιουργίας τους: "UserID/tTweetID/tTweet/tCreatedAt"

Κεφάλαιο 6

Κατηγοριοποίηση των Χρηστών του Twitter μέσω της Εξαγωγής και Επιλογής Χαρακτηριστικών - Υβριδική Προσέγγιση

Στο Κεφάλαιο αυτό, μετά την ενδελεχή ανάλυση του απαραίτητου θεωρητικού υποβάθρου και των τεχνικών, που χρησιμοποιήθηκαν για τους σκοπούς της παρούσας εργασίας, παρουσιάζονται οι μέθοδοι και τα αποτελέσματα του πρώτου σκέλους της υλοποίησής μας, το οποίο αφορά τη διαδικασία εξαγωγής χαρακτηριστικών (feature engineering). Συγκεκριμένα, στην ενότητα 6.1 παρουσιάζεται το σύνολο των features, το οποίο βρέθηκε βάσει της βιβλιογραφικής επισκόπησης, όπως περιγράφηκε στο Κεφάλαιο 2. Στην ενότητα 6.2 παρουσιάζεται μία συγκριτική μελέτη των μετρικών αξιολόγησης, όπως προέκυψαν βάσει των διαφόρων τεχνικών επιλογής χαρακτηριστικών και των αλγορίθμων Μηχανικής και Βαθιάς Μάθησης, που υλοποιήθηκαν στην παρούσα εργασία.

6.1 Εξαγωγή Χαρακτηριστικών - Feature extraction

Στο Κεφάλαιο αυτό, χρησιμοποιήσαμε τα ακόλουθα δεδομένα:

group name	accounts	tweets
genuine accounts	3,474	8,377,522
social spambots #1	991	1,610,176
social spambots #2	3,457	428,542
social spambots #3	464	1,418,626

Πίνακας 6.1: Σύνολο δεδομένων που χρησιμοποιήθηκε

Πριν προχωρήσουμε στη διαδικασία της εξαγωγής των χαρακτηριστικών, επεξεργαστήκαμε το dataset με τους ακόλουθους τρόπους. Αρχικά, από τα αρχεία .csv, που αφορούν τους **χρήστες**, κρατήσαμε τις στήλες: *'id', 'name', 'screen_name', 'statuses_count', 'followers_count', 'friends_count', 'favourites_count', 'listed_count', 'description', 'created_at', 'timestamp', 'updated', 'crawled_at'*. Από τα αρχεία .csv, που αφορούν τα **tweets**, κρατήσαμε τις στήλες: *'user_id', 'text', 'source', 'in_reply_to_user_id', 'favorite_count', 'retweet_count', 'num_hashtags', 'num_urls', 'num_mentions', 'created_at', 'timestamp'. Στη συνέχεια, όσον αφορά τις στήλες 'name', 'screen_name', 'description', 'text', 'source'* αντικαταστήσαμε τις τιμές NaN με το κενό ' ', ενώ όσον αφορά τις στήλες *'statuses_count', 'followers_count', 'friends_count', 'listed_count', 'favorite_count', 'retweet_count', 'num_hashtags', 'num_urls', 'num_mentions' & 'in_reply_to_user_id'* αντικαταστήσαμε τις τιμές NaN με το 0. Για την εύρεση hashtags, mentions, URLs στα tweets χρησιμοποιήθηκε η βιβλιοθήκη **re** της python, η οποία επιτρέπει λειτουργίες ταιριάσματος κανονικών εκφράσεων (Regular Expressions).

Στη συνέχεια, παρουσιάζονται τα χαρακτηριστικά, που υπολογίστηκαν στην παρούσα εργασία.

- **F1: Tweet time interval standard deviation (TISD)**

$$TISD(u) = \frac{\sum_{i=1}^{n}(T_i - \bar{T})^2}{N(u)} \tag{6.1}$$

όπου $T_1, T_2, \ldots, T_n$ αποτελούν τα χρονικά διαστήματα μεταξύ δύο διαδοχικών tweets και $\bar{T}$ αποτελεί το μέσο χρονικό διάστημα.

Η τιμή του TISD είναι γενικά εξαιρετικά χαμηλή για τους spammers, δεδομένου ότι είναι ενεργοί σε συγκεκριμένες χρονικές στιγμές.

- **F2: Retweet Ratio (RR)**

$$RR(u) = \frac{RT(u)}{N(u)} \tag{6.2}$$

όπου RT(u) είναι ο αριθμός των retweets του χρήστη u (tweets που ξεκινάνε με το πρόθεμα RT στο dataset μας) & N(u) ο συνολικός αριθμός των tweets του χρήστη u.

Η τιμή RR αναμένεται να είναι μικρή για bots και μεγάλη για αληθινούς χρήστες [39].

- **F3: URL ratio (UR)**

$$UR(u) = \frac{U(u)}{N(u)} \tag{6.3}$$

όπου U(u) είναι ο αριθμός των URLs, που χρησιμοποιήθηκαν στα tweets του χρήστη u & N(u) ο συνολικός αριθμός των tweets του χρήστη u. Οι spammers χρησιμοποιούν URLs στα περισσότερα tweets τους, με αποτέλεσμα η τιμή του UR να τείνει στη μονάδα και σε κάποιες περιπτώσεις να γίνεται μεγαλύτερη της μονάδας. Για τους αληθινούς λογαριασμούς, η τιμή αυτή είναι πολύ μικρή, σχεδόν ίση με 0, επιβεβαιώνοντας έτσι το γεγονός ότι τα tweets των αληθινών χρηστών αποτελούν σκέψεις και απόψεις σε ένα συγκεκριμένο topic.

- **F4: Maximum number of URLs in tweets**
- **F5: Unique URL ratio (UUR)**

$$UUR = \frac{UU(u)}{U(u)} \tag{6.4}$$

όπου UU(u) είναι ο αριθμός των μοναδικών URLs και U(u) είναι ο αριθμός των URLs, που χρησιμοποιήθηκαν στα tweets του χρήστη u.

Η τιμή UUR αναμένεται να είναι γενικά χαμηλή για τους spammers και υψηλή για αληθινούς χρήστες.

- **F6: Mention Ratio (MR)**

$$MR(u) = \frac{M(u)}{N(u)} \tag{6.5}$$

όπου M(u) είναι ο αριθμός των αναφορών (συμβολίζονται ως '@') στα tweets του χρήστη & N(u) ο αριθμός των tweets του χρήστη u.

Η τιμή του MR(u) αναμένεται να είναι χαμηλή για αληθινούς χρήστες και υψηλή για spammers.

- **F7: Maximum number of mentions in tweets**
- **F8: Unique Mention Ratio (UMR)**

$$UMR(u) = \frac{UM(u)}{M(u)} \tag{6.6}$$

όπου UM(u) είναι ο αριθμός των μοναδικών αναφορών στα tweets του χρήστη.

Γενικά, η τιμή UMR(u) αναμένεται να είναι χαμηλή για αληθινούς χρήστες, δεδομένου ότι αλληλεπιδρούν με ένα συγκεκριμένο πλήθος ατόμων, ενώ τείνει να είναι υψηλή για τους spammers.

- **F9: Hashtag Ratio (HTR)**

$$HTR(u) = \frac{HT(u)}{N(u)} \tag{6.7}$$

όπου HT(u) είναι ο αριθμός των hashtags, που χρησιμοποιήθηκαν στα tweets του χρήστη u.

Γενικά, η τιμή HTR(u) τείνει να είναι υψηλή για τους spammers και χαμηλή για τους πραγματικούς χρήστες.

- **F10: Maximum number of hashtags in tweets**
- **F11: Tweet Similarity**

$$TS(u) = \frac{2 \times \sum_{i=1}^{N(u)} \sum_{j=i+1}^{N(u)} \frac{NU^i(u) \cdot NU^j(u)}{\|NU^i(u)\| \cdot \|NU^j(u)\|}}{N(u) \cdot (N(u) - 1)} \tag{6.8}$$

Στόχος του χαρακτηριστικού αυτού είναι να βρει την ομοιότητα ανάμεσα στα tweets των χρηστών, δεδομένου ότι οι spammers ενδιαφέρονται για συγκεκριμένα topics. Προκειμένου να εξάγουμε το feature αυτό, ακολουθούμε την παρακάτω διαδικασία:

– Για την επεξεργασία των tweets, χρησιμοποιούμε τη βιβλιοθήκη ***ekphrasis*** [85] της python. Το ekphrasis, εκτελεί: **(1)** λεκτική ανάλυση (tokenization) η οποία διατηρεί εκφράσεις οι οποίες είναι χρήσιμες για τον προσδιορισμό του συναισθήματος, **(2)** ορθογραφική διόρθωση, **(3)** κανονικοποίηση λέξεων και φράσεων (text normalization), **(4)** σημείωση λέξεων και φράσεων (word annotation), **(5)** διαχωρισμό ενοποιημένων λέξεων (text segmentation) στις επιμέρους λέξεις (για τον διαχωρισμό των hashtags). Έχοντας, λοιπόν, επεξεργαστεί τα tweets, χρησιμοποιούμε ένα προ-εκπαιδευμένο μοντέλο GloVE και μετατρέπουμε κάθε token σε έναν πίνακα αριθμών. Για tokens, που δεν βρίσκονται στο μοντέλο GloVE, τα αντικαθιστούμε με τον χαρακτήρα "unk". Ως pre-trained model χρησιμοποιήσαμε το μοντέλο των 50 διαστάσεων.

  Στη συνέχεια, υπολογίσαμε για κάθε tweet του χρήστη τον μέσο όρο των embeddings (αναπαράσταση του κάθε tweet).

  Τέλος, υπολογίσαμε την ομοιότητα συνημιτόνου (cosine similarity) για όλα τα ζεύγη των tweets.

- **F12: Ratio between friends and followers**

$$\frac{Number\ of\ followers}{Number\ of\ friends} \tag{6.9}$$

  Η τιμή του χαρακτηριστικού αυτού αναμένεται να είναι ιδιαίτερα χαμηλή για τους spammers.

- **F13: Reputation of the account**

$$\frac{Number\ of\ followers + 1}{Number\ of\ friends + Number\ of\ followers + 1} \tag{6.10}$$

  Η τιμή αυτή αναμένεται να είναι ιδιαίτερα χαμηλή και κοντά στο 0 για τους spammers, καθώς οι spammers τείνουν να αποκτούν περισσότερους followers.

- **F14: Age of the account**

$$\frac{\sum(Time(Tweet) - User\ creation\ date)}{Total\ number\ of\ tweets} \tag{6.11}$$

  Οι spammers, συνήθως, έχουν μικρή ηλικία, καθώς δημιουργούν συνεχώς νέους λογαριασμούς, όταν διαγράφονται από τους περισσότερους χρήστες [1].

- **F15: Time between posts**

$$\frac{\sum(Time(Tweet_i) - Time(Tweet_j))}{Total\ number\ of\ tweets} \tag{6.12}$$

  όπου τα $Tweet_i$ & $Tweet_j$ αναφέρονται σε διαδοχικά tweets. Η τιμή του χαρακτηριστικού αυτού αναμένεται να είναι μικρή για τους spammers, καθώς δημοσιεύουν tweets με πολύ γρήγορο ρυθμό [86].

- **F16: Idle hours**

$$\frac{Max(Time(Tweet_i) - Time(Tweet_j))}{Total\ number\ of\ tweets} \tag{6.13}$$

  Η τιμή αυτή αναμένεται να είναι χαμηλή για τους spammers, καθώς δεν μένουν αδρανείς για μεγάλο διάστημα [50].

- **F17: Minimum time between posts**

- **F18: Average length of tweet**

$$\frac{\sum Tweet\ length}{Total\ number\ of\ tweets} \tag{6.14}$$

  Οι spammers δημοσιεύουν tweets με μικρό αριθμό χαρακτήρων, σε αντίθεση με τους πραγματικούς χρήστες. Επομένως, η τιμή του χαρακτηριστικού αυτού αναμένεται να είναι υψηλότερη για πραγματικούς χρήστες σε σχέση με τους spammers [39].

- **F19: Standard deviation of the length of tweets**
- **F20: Minimum length of a tweet**
- **F21: Maximum length of a tweet**
- **F22: Average of favourite tweets**

$$\frac{Total\ number\ of\ favorites}{Total\ number\ of\ tweets} \tag{6.15}$$

Συνήθως, τα tweets που δημοσιεύονται από spammers έχουν πολύ μικρό αριθμό favorites. Επομένως, η τιμή του χαρακτηριστικού αυτού αναμένεται να είναι υψηλότερη για πραγματικούς χρήστες σε σχέση με τους spammers.

- **F23: Average of retweets of the tweets**

$$\frac{Total\ number\ of\ retweets}{Total\ number\ of\ tweets} \tag{6.16}$$

Τα tweets, που δημοσιεύονται από spammers, δεν αναδημοσιεύονται από άλλους χρήστες [87].

Επομένως, η τιμή του χαρακτηριστικού αυτού αναμένεται να είναι υψηλότερη για πραγματικούς χρήστες σε σχέση με τους spammers.

- **F24: Tweet Sources (TwSc) [38]**

$$TwSc = \frac{Number\ of\ different\ sources\ a\ user\ may\ use}{Number\ of\ different\ sources\ in\ total} \tag{6.17}$$

Οι χρήστες δημοσιεύουν τα tweets με διάφορους τρόπους, είτε μέσω του twitter στο web είτε μέσω του Twitter API ή χρησιμοποιώντας διάφορες άλλες εφαρμογές. Ένας πραγματικός χρήστης δεν χρησιμοποιεί, συνήθως, έναν συγκεκριμένο τρόπο σε αντίθεση με τους spammers, που περιορίζονται σε συγκεκριμένους τρόπους δημοσίευσης των μηνυμάτων τους.

- **F25: Following Rate**

$$FR = \frac{Number\ of\ friends/followings}{Age\ of\ the\ account} \tag{6.18}$$

- **F26: Length of username**
- **F27: Length of screen-name**
- **F28: Length of profile description**
- **F29: Number of URL links per word**
- **F30: Number of mentions per word**
- **F31: Levenshtein distance between username - screenname [47]**

Οι spammers, συνήθως, επιλέγουν παρόμοια usernames & screennames, σε αντίθεση με τους αληθινούς χρήστες, που είναι περισσότερο δημιουργικοί.

- **F32: Maximum time between retweets**
- **F33: Minimum time between retweets**
- **F34: Mean Time between retweets**
- **F35: Standard deviation of time between retweets**
- **F36: Uppercase word rate**
- **F37: elongated word rate**

- **F38: repeated mixed punctuation rate**
- **F39 - F40: Average Polarity/Subjectivity** Στην εργασία αυτή χρησιμοποιήσαμε τη βιβλιοθήκη της python, TextBlob [88], καθώς οι spammers τείνουν να δημοσιεύουν tweets με πολύ θετικό ή πολύ αρνητικό περιεχόμενο.

  Συγκεκριμένα υπολογίσαμε για κάθε tweet την πολικότητά του καθώς και πόσο υποκειμενικό ή αντικειμενικό είναι. Η πολικότητα είναι ένας δεκαδικός αριθμός που βρίσκεται στο εύρος του [-1,1] όπου 1 σημαίνει απόλυτα θετική δήλωση και -1 σημαίνει απόλυτα αρνητική δήλωση. Η υποκειμενικότητα δηλώνει κατά πόσο μια φράση αναφέρεται σε προσωπική γνώμη ή σε πραγματικές πληροφορίες. Με αυτόν τον τρόπο, υπολογίσαμε τη μέση τιμή του σκορ αυτού για κάθε λογαριασμό.

  Για τον υπολογισμό των χαρακτηριστικών **F41-F43** χρησιμοποιήθηκε η βιβλιοθήκη vaderSentiment [89].

- **F41: Average positive sentiment per tweet**

$$\frac{Total\ positive\ sentiment}{Total\ number\ of\ tweets} \tag{6.19}$$

- **F42: Average neutral sentiment per tweet**

$$\frac{Total\ neutral\ sentiment}{Total\ number\ of\ tweets} \tag{6.20}$$

- **F43: Average negative sentiment per tweet**

$$\frac{Total\ negative\ sentiment}{Total\ number\ of\ tweets} \tag{6.21}$$

- **F44: Mean time between replies**
- **F45: Maximum time between replies**
- **F46: Minimum time between replies**
- **F47: Standard deviation of time between replies**
- **F48: Number of hashtags in description**
- **F49: Number of URLs in description**
- **F50: Length of screen_name / Length of username**
- **F51: favourites / Age of the account**
- **F52: friends / Age of the account**
- **F53: Unique Tweet Ratio (UTR)**

$$UTR(u) = \frac{Number\ of\ unique\ tweets\ posted\ by\ user\ u}{Total\ number\ of\ tweets\ posted\ by\ user\ u} \tag{6.22}$$

- **F54: listed count / Age of the account**
- **F55: Number of followers**
- **F56: Number of friends**
- **F57: favourites_count**
- **F58: listed_count**
- **F59: statuses_count**
- **F60: Number of tweets posted per day**

- **F61: Number of tweets posted per week**
- **F62: Unique Replies**

$$\frac{Unique\ replies}{Total\ tweets} \tag{6.23}$$

- **F63: Average number of tweets containing only URL**

$$\frac{Number\ of\ tweets\ containing\ only\ URLs}{Total\ number\ of\ tweets} \tag{6.24}$$

- **F64: Unique n-grams per tweet**

  Ακολουθήσαμε τα παρακάτω στάδια προεπεξεργασίας των tweets:

  - Μετατρέψαμε όλους τους χαρακτήρες των tweets σε lowercase.
  - Αντικαταστήσαμε όλες τις ιστοσελίδες (URLs), που βρέθηκαν στα tweets των χρηστών με το token '<url>'.
  - Αντικαταστήσαμε όλα τα hashtags, που βρέθηκαν στα tweets των χρηστών με το token '<hashtag>'.
  - Αντικαταστήσαμε όλα τα mentions, που χρησιμοποιεί ο χρήστης στα tweets του, με το token '<user>'.
  - Με τη χρήση της βιβλιοθήκης ekphrasis, λέξεις όπως 'yaaaaaaaayyyyyyy' αντικαταστάθηκαν από την κανονική λέξη ακολουθούμενη από το token '<elongated>'. Στην προκειμένη περίπτωση, η λέξη 'yaaaaaaaayyyyyyy' αντικαταστάθηκε ως 'yay' '<elongated>'. Επιπρόσθετα, εκφράσεις οι οποίες δεν προσφέρουν επιπλέον πληροφορία για την ανίχνευση των bots, αλλά αποτελούν απλά θόρυβο για το μοντέλο αντικαταστάθηκαν με τα κατάλληλα tokens. Παραδείγματα αποτελούν τα εξής:
    - Τηλεφωνικοί Αριθμοί αντικαταστάθηκαν με τον όρο '<phone>'
    - Συναλλάγματα αντικαταστάθηκαν με τον όρο '<money>'
    - Ημερομηνίες αντικαταστάθηκαν με τον όρο '<date>'
    - Ώρες αντικαταστάθηκαν με τον όρο '<time>'
    - Ποσοστά αντικαταστάθηκαν με τον όρο '<percent>'
  - Ένα τελευταίο βήμα προεπεξεργασίας ήταν να εφαρμόσουμε αποκατάληξη (stemming) σε κάθε λέξη, που χρησιμοποιεί ο κάθε χρήστης στα tweets του, προκειμένου λέξεις όπως 'document' & 'documents' να προστεθούν μία και όχι δύο φορές ως δύο διαφορετικές λέξεις.
  - Τέλος, για τα tweets του κάθε χρήστη υπολογίσαμε τα ακόλουθα δύο χαρακτηριστικά.

$$F64A : \frac{Unique\ unigram\ (1-gram)\ words}{Total\ number\ of\ tweets} \tag{6.25}$$

$$F64B : \frac{Unique\ bigram\ (2-gram)\ words}{Total\ number\ of\ tweets} \tag{6.26}$$

- **F65: Digital DNA (type of tweets)**

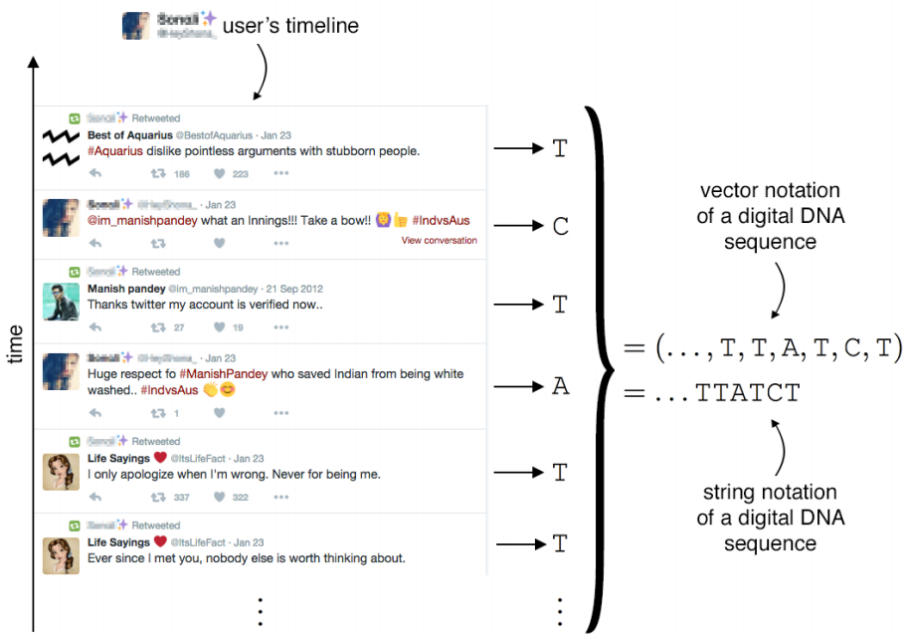
  Εξάγουμε χαρακτηριστικά ακολουθώντας τη μέθοδο του **Digital DNA**, όπως προτάθηκε από τους ***Cresci et al.*** [31]. Συγκεκριμένα, ακολουθήσαμε τη μεθοδολογία των ***Pasricha & Hayes*** [32]. Αρχικά, ταξινομήσαμε τα tweets του χρήστη κατά χρονολογική σειρά. Στη συνέχεια, αντικαταστήσαμε κάθε tweet, retweet & reply του χρήστη με τους χαρακτήρες A (αδενίνη), T (θυμίνη) & C (κυτοσίνη) δημιουργώντας με αυτόν τον τρόπο μία αλληλουχία DNA. Μοντελοποιούμε, δηλαδή τη συμπεριφορά του χρήστη στο Twitter βάσει του τύπου των δημοσιεύσεών του. Έπειτα, αφού μετατρέψαμε τη συμβολοσειρά των ASCII χαρακτήρων σε ένα αντικείμενο bytes στη μνήμη, συμπιέσαμε την ακολουθία. Έτσι, προέκυψαν τα ακόλουθα τρία χαρακτηριστικά:

  - **F65A: Μέγεθος (σε bytes) ακολουθίας** (S_μ) **πριν τη συμπίεση**

– **F65B: Μέγεθος (σε bytes) ακολουθίας** (C_μ) **μετά τη συμπίεση**

– **F65C: Μέσος Συντελεστής συμπίεσης** (R_μ)

Ένα παράδειγμα δημιουργίας μίας αλληλουχίας DNA φαίνεται στην παρακάτω εικόνα [90], όπου κάθε tweet, retweet & reply του χρήστη έχει αντικατασταθεί από τους χαρακτήρες A, T & C.



Εικόνα 6.1: Digital DNA (type of tweets)

- **F66: Digital DNA (tweet content)**

  Για την εξαγωγή του χαρακτηριστικού αυτού ακολουθούμε την ίδια διαδικασία που ακολουθήσαμε κατά την εξαγωγή του χαρακτηριστικού F80 με μία διαφορά. Μοντελοποιούμε τη συμπεριφορά του χρήστη στο Twitter βάσει του περιεχομένου αντί του τύπου των δημοσιεύσεών. Συγκεκριμένα, σχεδιάσαμε τις παρακάτω βάσεις βάσει του περιεχομένου των δημοσιεύσεων:

base	content of tweet
N	tweet contains no entities (plain text)
U	tweet contains one or more URLs
H	tweet contains one or more hashtags
M	tweet contains one or more mentions
X	tweet contains entities of mixed types

Πίνακας 6.2: Digital DNA (tweet content)

Όπως και προηγουμένως, υλοποιήσαμε για κάθε χρήστη μία αλληλουχία DNA. Έπειτα, αφού μετατρέψαμε τη συμβολοσειρά των ASCII χαρακτήρων σε ένα αντικείμενο bytes στη μνήμη, συμπιέσαμε την ακολουθία. Έτσι, προέκυψαν τα ακόλουθα τρία χαρακτηριστικά:

– **F66A: Μέγεθος (σε bytes) ακολουθίας** (S_μ) **πριν τη συμπίεση**

– **F66B: Μέγεθος (σε bytes) ακολουθίας** (C_μ) **μετά τη συμπίεση**

– **F66C: Μέσος Συντελεστής συμπίεσης** (R_μ)

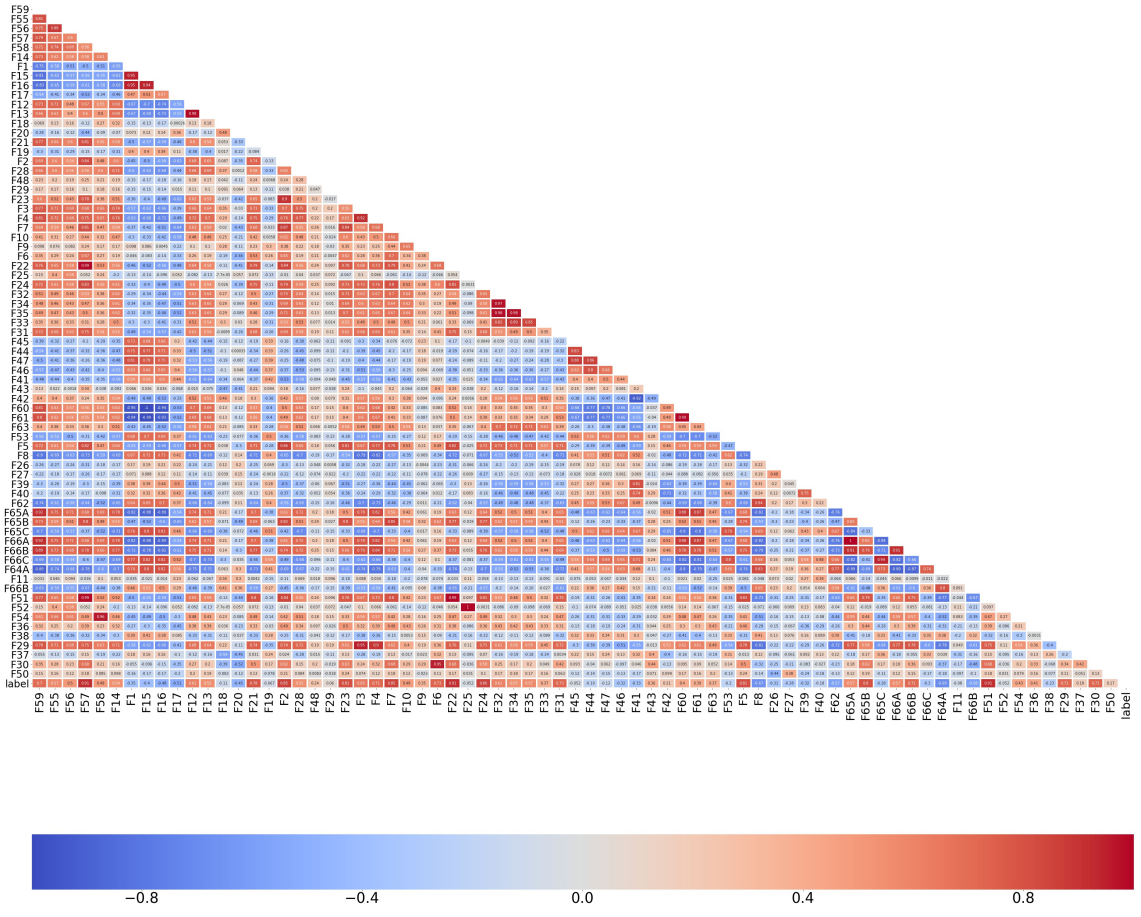
6.2 Υλοποίηση και Αποτελέσματα

Γνωρίζουμε, όμως ότι δύο features, που έχουν υψηλό αριθμό συσχέτισης, είναι πλεονάζοντα και προτιμάται να χρησιμοποιηθεί μόνο το ένα εκ των δύο κατά την εκπαίδευση του αλγορίθμου μηχανικής μάθησης, με την έννοια ότι δεν πρόκειται να ληφθεί κάποια επιπλέον πληροφορία με τον συνδυασμό των δύο αυτών features.

Για τον υπολογισμό της συσχέτισης στο πείραμά μας χρησιμοποιήσαμε το Spearman Correlation Coefficient, επειδή τα features παίρνουν συνεχείς τιμές, ενώ η κλάση (target value) παίρνει διακριτές τιμές, 0 (bots) & 1 (humans). Πρόκειται για έναν αριθμό μεταξύ -1 και 1, όπου το 1 δηλώνει πλήρη γραμμική συσχέτιση και το 0 καμία συσχέτιση των δύο σειρών δεδομένων. Συγκεκριμένα, όσο μεγαλύτερη είναι η απόλυτη τιμή, τόσο μεγαλύτερη είναι και η συσχέτιση των δύο μεταβλητών. Ο συντελεστής συσχέτισης Spearman ορίζεται όπως ο συντελεστής συσχέτισης Pearson μεταξύ των μεταβλητών κατάταξης. Δίνεται από τον ακόλουθο τύπο:

$$\rho = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i (x_i - \bar{x})^2 \sum_i (y_i - \bar{y})^2}} \tag{6.27}$$

Στην παρακάτω εικόνα, φαίνεται η συσχέτιση τόσο μεταξύ των features όσο και η συσχέτιση του κάθε feature με την έξοδο (label), δηλαδή το κατά πόσο το κάθε feature συμμετέχει στην τελική πρόβλεψη.



Εικόνα 6.2: Πίνακας Συσχέτισης

Για την επιλογή χαρακτηριστικών (feature selection) αλλά και την εκπαίδευση όλων των αλγορίθμων μηχανικής μάθησης χρησιμοποιήθηκε η βιβλιοθήκη ***scikit learn*** [91] της python.

Εύρεση βέλτιστου συνδυασμού features + machine learning algorithms: Για την επιλογή της καλύτερης μεθόδου επιλογής χαρακτηριστικών και της εύρεσης των βέλτιστων παραμέτρων των αλγορίθμων μηχανικής μάθησης, χρησιμοποιήσαμε μπεϋζιανή βελτιστοποίηση.

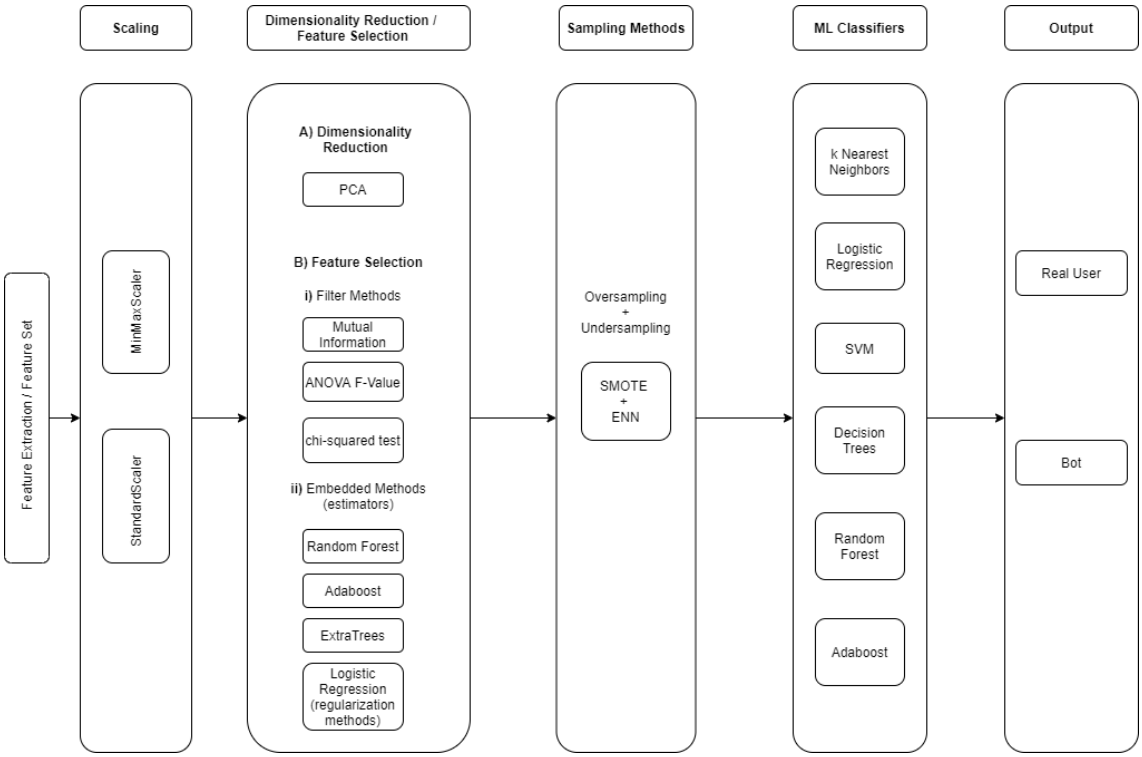
Ανομοιογενές Σύνολο Δεδομένων (Imbalanced Dataset): Παρατηρούμε ότι το σύνολο δεδομένων είναι ανομοιογενές. Συγκεκριμένα, περιλαμβάνονται 3474 real users και 4912 bots. Αυτό

έχει ως αποτέλεσμα, κάθε νέο σύνολο χαρακτηριστικών σε έναν αλγόριθμο μηχανικής μάθησης να ταξινομεί τον συγκεκριμένο χρήστη στην κλάση με τα περισσότερα δεδομένα, δηλαδή στην κλάση bot. Προκειμένου να αντιμετωπίσουμε αυτό το πρόβλημα, χρησιμοποιούμε τεχνικές παραγωγής συνθετικών δεδομένων. Συγεκριμένα, χρησιμοποιούμε ένα συνδυασμό του αλγορίθμου SMOTE (oversampling) με Edited Nearest Neighbors (ENN), που αποτελεί μία τεχνική υποδειγματοληψίας (undersampling). Για την υλοποίηση της τεχνικής αυτής χρησιμοποιήσαμε τη βιβλιοθήκη imbalanced-learn [92].

Κανονικοποίηση Χαρακτηριστικών: Πραγματοποιήθηκε κανονικοποίηση των χαρακτηριστικών στην ίδια κλίμακα.

Σε όλα τα πειράματα χρησιμοποίησαμε nested cross validation. Συγκεκριμένα, στο εσωτερικό loop κατά την εύρεση των βέλτιστων υπερπαραμέτρων χρησιμοποιήθηκε 5-fold cross validation. Ομοίως, στο εξωτερικό loop κατά την εκπαίδευση και αξιολόγηση των αλγορίθμων Μηχανικής Μάθησης χρησιμοποιήθηκε 5-fold cross validation.

Στην παρακάτω εικόνα παρουσιάζεται η διαδικασία, που ακολουθήθηκε, για την κατηγοριοποίηση των χρηστών του Twitter σε real users & bots.



Εικόνα 6.3: Our approach for detecting bots in Twitter

Στη συνέχεια, γίνεται μία συγκριτική αξιολόγηση των μεθόδων, που χρησιμοποιήθηκαν, για την επιλογή χαρακτηριστικών (feature selection) και αλγορίθμων μηχανικής μάθησης.

- **Principal Component Analysis (PCA):** Χρησιμοποιούμε την τεχνική PCA, που όπως είδαμε αποτελεί μία τεχνική μείωσης διαστάσεων. Προτού εφαρμόσουμε την τεχνική PCA, πραγματοποιήσαμε κανονικοποίηση χαρακτηριστικών.

Classifier/ Evaluation Metrics	F1-Score	Accuracy	Recall	Precision	AUC/ROC
SVM	0.9854 ± 0.0033	0.9879 ± 0.0028	0.9926 ± 0.0023	0.978 ± 0.0047	0.99 ± 0.0018
Decision Trees	0.9824 ± 0.0039	0.9855 ± 0.0032	0.989 ± 0.0048	0.9759 ± 0.0055	0.9861 ± 0.0032
Random Forest	0.9844 ± 0.002	0.9871 ± 0.0017	0.9932 ± 0.0027	0.9757 ± 0.0055	0.9963 ± 0.0019
AdaBoost	0.9831 ± 0.0035	0.9861 ± 0.0029	0.9897 ± 0.0033	0.9767 ± 0.0065	0.992 ± 0.0024
Logistic Regression	0.979 ± 0.0044	0.9827 ± 0.0036	0.9832 ± 0.0053	0.9748 ± 0.0048	0.9954 ± 0.0015
kNN	0.9844 ± 0.0031	0.9872 ± 0.0026	0.9896 ± 0.0026	0.9793 ± 0.0045	0.9902 ± 0.0018

Πίνακας 6.3: Evaluation Metrics Using PCA

Στη συνέχεια, παρατίθενται οι **μέθοδοι φιλτραρίσματος (filter methods)**, που υλοποιήθηκαν για την επιλογή των χαρακτηριστικών.

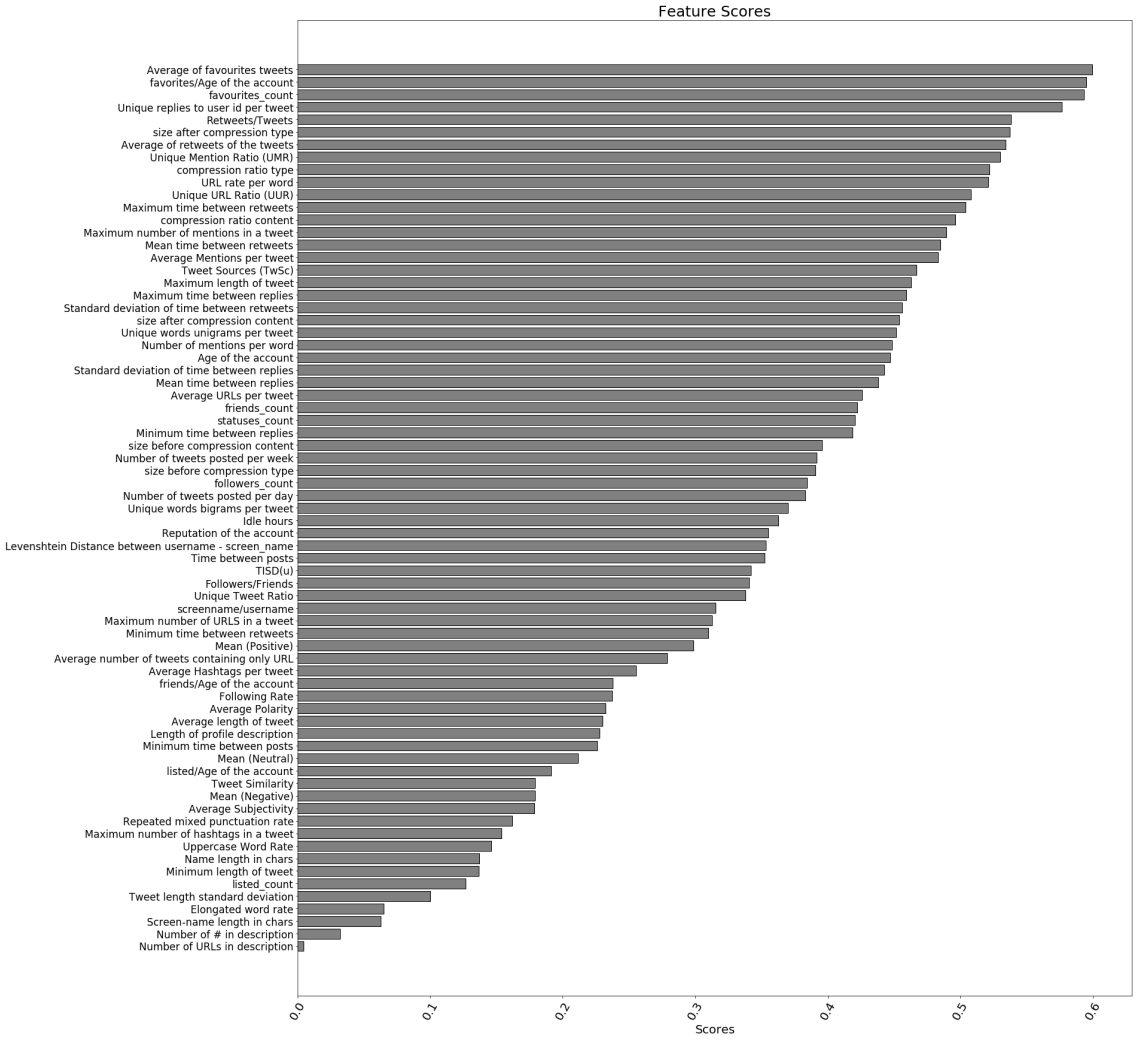
- **Mutual Information:** Αφού κανονικοποιήσαμε τα χαρακτηριστικά στην κλίμακα (0,1), εφαρμόσαμε την τεχνική αυτή, προκειμένου να βρεθεί το βέλτιστο υποσύνολο χαρακτηριστικών κάθε φορά. Στη συνέχεια, υλοποιήσαμε την τεχνική SMOTE+ENN αξιολογήσαμε την επίδοση των χαρακτηριστικών που επιλέχθησαν με αλγορίθμους μηχανικής μάθησης, όπως φαίνεται στον πίνακα που ακολουθεί.

Classifier/ Evaluation Metrics	F1-Score	Accuracy	Recall	Precision	AUC/ROC
SVM	0.9833 ± 0.0055	0.9862 ± 0.0028	0.9902 ± 0.0048	0.9765 ± 0.0071	0.9933 ± 0.0021
Decision Trees	0.9819 ± 0.0043	0.9851 ± 0.0035	0.9876 ± 0.0045	0.9764 ± 0.006	0.9860 ± 0.0028
Random Forest	0.9851 ± 0.0039	0.9877 ± 0.0033	0.9938 ± 0.0017	0.9765 ± 0.0065	0.9964 ± 0.0017
AdaBoost	0.9848 ± 0.0039	0.9874 ± 0.0032	0.9923 ± 0.0025	0.9774 ± 0.0076	0.9947 ± 0.0016
Logistic Regression	0.98 ± 0.0011	0.9836 ± 0.0009	0.9849 ± 0.0050	0.975 ± 0.0053	0.9921 ± 0.0017
kNN	0.9826 ± 0.0054	0.9857 ± 0.0044	0.9864 ± 0.0053	0.979 ± 0.0079	0.9882 ± 0.0040

Πίνακας 6.4: Evaluation Metrics Using Mutual Information

Πρέπει να τονιστεί ότι κατά την εκπαίδευση του αλγορίθμου SVM χρησιμοποιήθηκαν τα πρώτα 16 χαρακτηριστικά, όπως παρουσιάζονται στην εικόνα 6.3. Κατά την εκπαίδευση των Δέντρων Απόφασης, του AdaBoost, της Λογιστικής Παλινδρόμησης και του kNN χρησιμοποιήθηκαν τα 18 πρώτα χαρακτηριστικά, ενώ κατά την εκπαίδευση των Τυχαίων Δασών χρησιμοποιήθηκαν τα πρώτα 17.

Στην παρακάτω εικόνα παρουσιάζεται η κατάταξη των χαρακτηριστικών βάσει των σκορ τους.



Εικόνα 6.4: Feature Scores Using Mutual Information

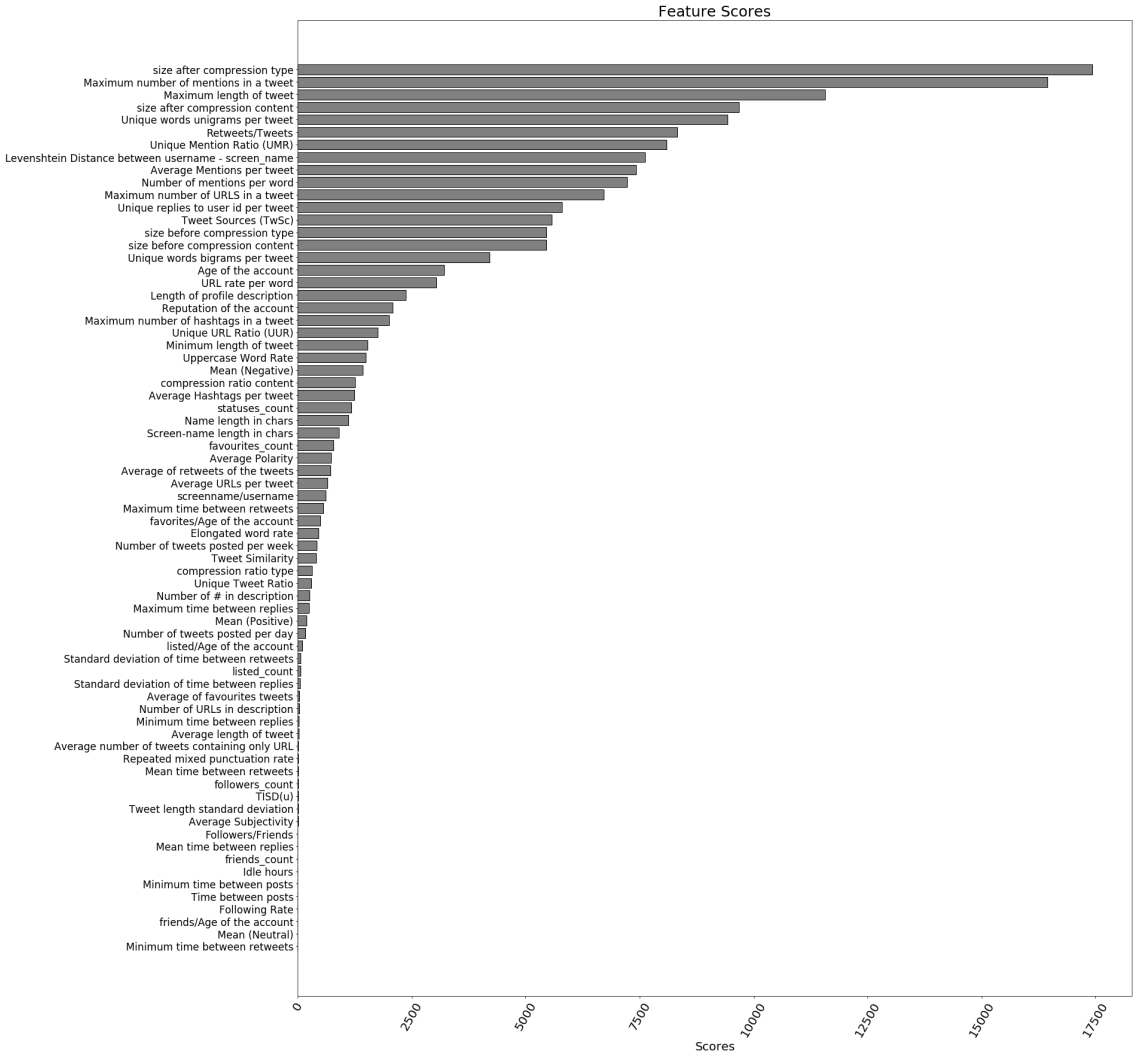
- **ANOVA F-Value:** Αρχικά, πραγματοποιήθηκε κανονικοποίηση των χαρακτηριστικών στην κλίμακα (0,1). Στη συνέχεια, κρίθηκε αναγκαία η αφαίρεση χαρακτηριστικών, των οποίων η τιμή είναι σταθερή σε όλα τα δεδομένα εκπαίδευσης. Η τυπική απόκλιση των χαρακτηριστικών αυτών είναι ίση με 0. Έπειτα, εφαρμόσαμε τις τεχνικές ANOVA-F Value, SMOTE+ENN & εκπαίδευση των αλγορίθμων Μηχανικής Μάθησης.

Classifier/ Evaluation Metrics	F1-Score	Accuracy	Recall	Precision	AUC/ROC
SVM	0.9877 ± 0.0044	0.9898 ± 0.0037	0.9955 ± 0.0026	0.98 ± 0.008	0.9976 ± 0.0013
Decision Trees	0.9824 ± 0.0043	0.9855 ± 0.0036	0.9885 ± 0.0046	0.9764 ± 0.0074	0.986 ± 0.0035
Random Forest	0.9857 ± 0.0046	0.9882 ± 0.0038	0.9946 ± 0.0027	0.9768 ± 0.0069	0.9972 ± 0.0018
AdaBoost	0.9843 ± 0.0048	0.9871 ± 0.0039	0.9914 ± 0.0051	0.9773 ± 0.0055	0.9959 ± 0.0014
Logistic Regression	0.9798 ± 0.0037	0.9833 ± 0.0031	0.9870 ± 0.0021	0.9727 ± 0.0061	0.9933 ± 0.0029
kNN	0.9843 ± 0.0044	0.9871 ± 0.0036	0.9893 ± 0.0054	0.9793 ± 0.0047	0.9897 ± 0.0032

Πίνακας 6.5: Evaluation Metrics Using ANOVA-F Value

Πρέπει να τονιστεί ότι κατά την εκπαίδευση του αλγορίθμου SVM χρησιμοποιήθηκαν τα πρώτα 17 χαρακτηριστικά, όπως παρουσιάζονται στην εικόνα 6.4. Κατά την εκπαίδευση των υπολοίπων αλγορίθμων Μηχανικής Μάθησης χρησιμοποιήθηκαν τα 18 πρώτα χαρακτηριστικά.

Στην παρακάτω εικόνα παρουσιάζεται η κατάταξη των χαρακτηριστικών βάσει των σκορ τους.



Εικόνα 6.5: Feature Scores Using ANOVA-F Value

- **Chi-squared test:** Όπως και προηγουμένως, κανονικοποιήσαμε τα χαρακτηριστικά στην κλίμακα (0,1), εφαρμόσαμε την τεχνική φιλτραρίσματος, τη μέθοδο SMOTE+ENN και τέλος εκπαιδεύσαμε τους αλγορίθμους Μηχανικής Μάθησης.

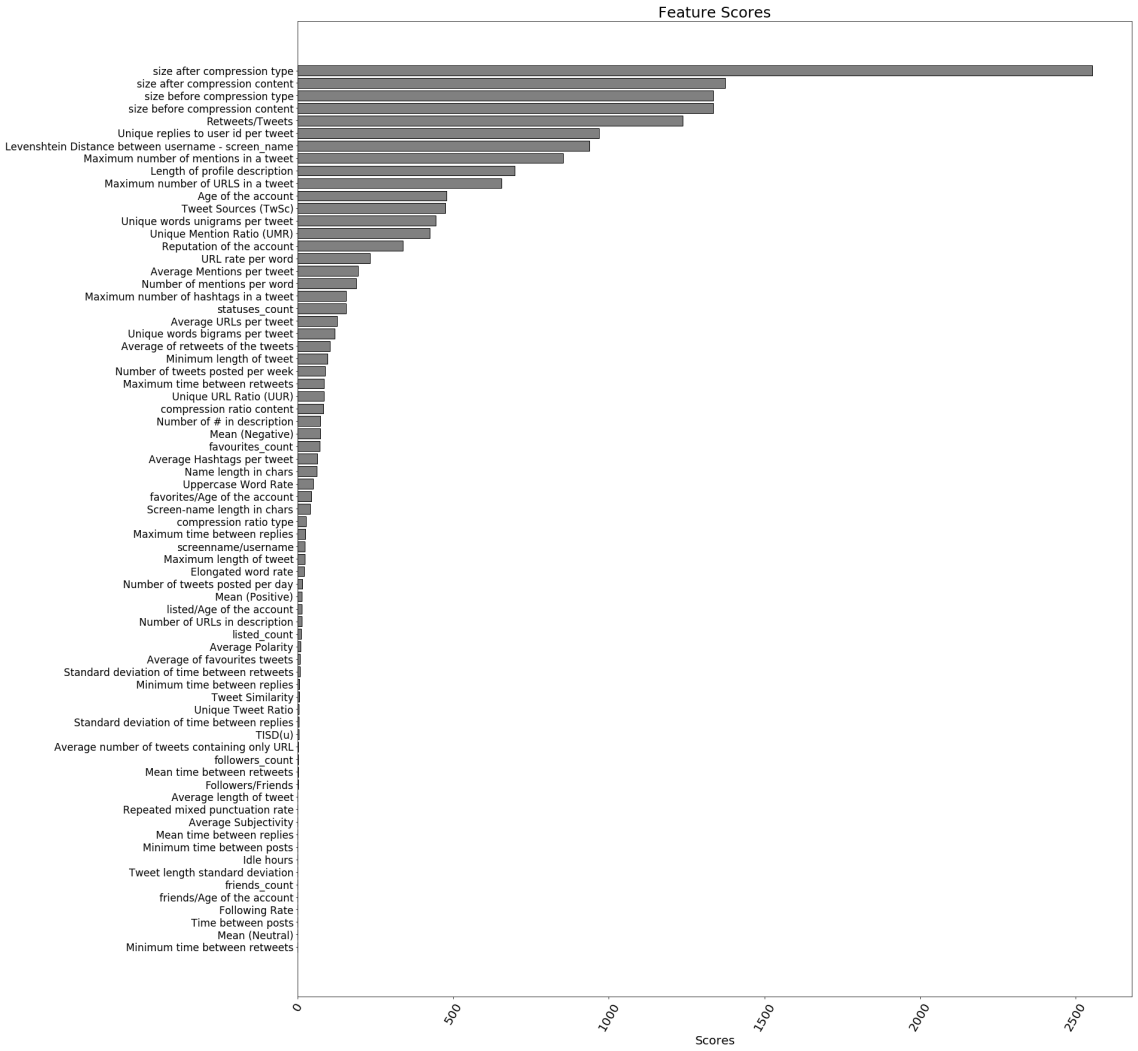
Classifier/ Evaluation Metrics	F1-Score	Accuracy	Recall	Precision	AUC/ROC
SVM	0.9829 ± 0.0017	0.9859 ± 0.0014	0.9917 ± 0.0044	0.9742 ± 0.0032	0.9960 ± 0.0007
Decision Trees	0.9821 ± 0.0019	0.9853 ± 0.0016	0.9891 ± 0.0053	0.9753 ± 0.0029	0.9859 ± 0.002
Random Forest	0.9865 ± 0.0015	0.9889 ± 0.0013	0.9961 ± 0.0015	0.9771 ± 0.0031	0.9958 ± 0.0022
AdaBoost	0.9858 ± 0.0023	0.9883 ± 0.0019	0.9938 ± 0.0043	0.9779 ± 0.0032	0.9943 ± 0.0013
Logistic Regression	0.9816 ± 0.0047	0.9849 ± 0.0038	0.9873 ± 0.0075	0.9761 ± 0.0059	0.9925 ± 0.0014
kNN	0.9828 ± 0.0022	0.9859 ± 0.0018	0.9882 ± 0.0052	0.9775 ± 0.0018	0.9891 ± 0.0017

Πίνακας 6.6: Evaluation Metrics Using Chi-squared test

Πρέπει να τονιστεί ότι κατά την εκπαίδευση των αλγορίθμου SVM, Decision Trees & Logistic Regression χρησιμοποιήθηκαν τα πρώτα 18 χαρακτηριστικά, όπως παρουσιάζονται στην εικόνα

6.5. Κατά την εκπαίδευση του αλγορίθμου Random Forest χρησιμοποιήθηκαν τα 14 πρώτα χαρακτηριστικά, κατά την εκπαίδευση του αλγορίθμου AdaBoost χρησιμοποιήθηκαν τα 15 πρώτα χαρακτηριστικά και κατά την εκπαίδευση του αλγορίθμου kNN χρησιμοποιήθηκαν τα 16 πρώτα χαρακτηριστικά.

Στην παρακάτω εικόνα παρουσιάζεται η κατάταξη των χαρακτηριστικών βάσει των σκορ τους.



Εικόνα 6.6: Feature Scores using Chi-squared test

Στη συνέχεια, χρησιμοποιούμε **ενσωματωμένες μεθόδους (embedded methods)** για την εύρεση του βέλτιστου υποσυνόλου χαρακτηριστικών. Στις μεθόδους αυτές υλοποιούμε τα ακόλουθα βήματα. Αρχικά, κανονικοποιούμε τα χαρακτηριστικά. Στη συνέχεια, για την εύρεση του υποσυνόλου των χαρακτηριστικών εκπαιδεύουμε κάποιον αλγόριθμο (Random Forests, ExtraTrees, AdaBoost, Logistic Regression), ο οποίος περιέχει ενσωματωμένη την τεχνική επιλογής χαρακτηριστικών. Επιλέγουμε τα χαρακτηριστικά εκείνα, των οποίων η σημασία (importance) ή το βάρος (coefficient) είναι μεγαλύτερη του μέσου όρου της σημασίας/των βαρών όλων των χαρακτηριστικών. Στη συνέχεια, υλοποιούμε την τεχνική SMOTE+ENN και τέλος εκπαιδεύουμε αλγορίθμους Μηχανικής Μάθησης.

- **Estimator: Random Forest Classifier**

Classifier/ Evaluation Metrics	F1-Score	Accuracy	Recall	Precision	AUC/ROC
SVM	0.9856 ± 0.0023	0.9882 ± 0.0019	0.9941 ± 0.0018	0.9774 ± 0.0060	0.9946 ± 0.0016
Decision Trees	0.9811 ± 0.0038	0.9844 ± 0.0031	0.9882 ± 0.0069	0.9741 ± 0.0053	0.9853 ± 0.0032
Random Forest	0.9857 ± 0.005	0.9882 ± 0.0042	0.9955 ± 0.0018	0.9760 ± 0.0084	0.9962 ± 0.0013
AdaBoost	0.9860 ± 0.0022	0.9884 ± 0.0019	0.9955 ± 0.0018	0.9766 ± 0.0055	0.9964 ± 0.0011
Logistic Regression	0.9791 ± 0.0039	0.9829 ± 0.0033	0.9817 ± 0.0034	0.9766 ± 0.0081	0.9927 ± 0.0026
kNN	0.9819 ± 0.0052	0.9851 ± 0.0043	0.9855 ± 0.0043	0.9784 ± 0.0093	0.9885 ± 0.0024

Πίνακας 6.7: Evaluation Metrics Using Embedded Method (Random Forest)

Ο αλγόριθμος SVM εκπαιδεύτηκε με τα ακόλουθα χαρακτηριστικά: F59, F56, F57, F21, F2, F23, F7, F6, F22, F24, F32, F35, F47, F5, F8, F62, F64A, F65B, F51, F29 & F30.
Ο αλγόριθμος Decision Trees εκπαιδεύτηκε με τα ακόλουθα χαρακτηριστικά: F557, F2, F23, F6, F22, F24, F32, F5, F8, F62, F65B, F66C, F51, F29 & F30.
Ο αλγόριθμος Random Forest εκπαιδεύτηκε με τα ακόλουθα χαρακτηριστικά: F57, F21, F2, F23, F7, F6, F22, F24, F5, F8, F65B, F65C, F64A, F51, F29 & F30.
Ο αλγόριθμος Adaboost εκπαιδεύτηκε με τα ακόλουθα χαρακτηριστικά: F56, F57, F2, F23, F7, F6, F22, F24, F34, F42, F65B & F29.
Ο αλγόριθμος kNN εκπαιδεύτηκε με τα ακόλουθα χαρακτηριστικά: F59, F57, F21, F2, F23, F7, F6, F22, F24, F32, F5, F8, F65B, F64A, F51, F29 & F30.
Ο αλγόριθμος Logistic Regression εκπαιδεύτηκε με τα ακόλουθα χαρακτηριστικά: F57, F21, F2, F23, F3, F7, F6, F22, F24, F32, F35, F8, F65B, F65C, F66B, F64A, F51, F29 & F30.

- **Estimator: AdaBoost Classifier**

Classifier/ Evaluation Metrics	F1-Score	Accuracy	Recall	Precision	AUC/ROC
SVM	0.9874 ± 0.0017	0.9896 ± 0.0015	0.9967 ± 0.0011	0.9783 ± 0.0041	0.9978 ± 0.0006
Decision Trees	0.9827 ± 0.0064	0.9857 ± 0.0053	0.9899 ± 0.0051	0.9757 ± 0.01	0.9878 ± 0.005
Random Forest	0.9871 ± 0.0045	0.9894 ± 0.0038	0.9973 ± 0.0019	0.9772 ± 0.0091	0.9978 ± 0.0011
AdaBoost	0.9831 ± 0.0035	0.9861 ± 0.0029	0.9897 ± 0.0033	0.9767 ± 0.0065	0.992 ± 0.0024
Logistic Regression	0.9818 ± 0.0032	0.985 ± 0.0026	0.987 ± 0.0048	0.9767 ± 0.0054	0.9945 ± 0.0008
kNN	0.9847 ± 0.0034	0.9874 ± 0.0028	0.9899 ± 0.0038	0.9796 ± 0.0059	0.9906 ± 0.0014

Πίνακας 6.8: Evaluation Metrics Using Embedded Method (AdaBoost)

Ο αλγόριθμος SVM εκπαιδεύτηκε με τα ακόλουθα χαρακτηριστικά: F59, F55, F14, F18, F21, F19, F22, F25, F43, F53, F5, F8, F62, F40, F64A, F66C, F64B, F29 & F50.
Ο αλγόριθμος Decision Trees εκπαιδεύτηκε με τα ακόλουθα χαρακτηριστικά: F55, F18, F21, F22, F43, F53, F8, F31 & F61.
Ο αλγόριθμος Random Forest εκπαιδεύτηκε με τα ακόλουθα χαρακτηριστικά: F55, F18, F21, F22, F6, F25, F31, F43, F61, F53 & F8.
Ο αλγόριθμος Adaboost εκπαιδεύτηκε με τα ακόλουθα χαρακτηριστικά: F59, F55, F14, F18, F21, F19, F22, F25, F43, F53, F5, F8, F62, F40, F64A, F66C, F64B, F29 & F50.
Ο αλγόριθμος kNN εκπαιδεύτηκε με τα ακόλουθα χαρακτηριστικά: F59, F55, F14, F18, F21, F19, F22, F52, F43, F53, F5, F8, F62, F40, F64A, F66C, F64B, F29 & F50.
Ο αλγόριθμος Logistic Regression εκπαιδεύτηκε με τα ακόλουθα χαρακτηριστικά: F59, F55, F56, F14, F18, F20, F21, F19, F48, F7, F22, F25, F24, F32, F31, F45, F43, F42, F66, F61, F53, F5, F8, F26, F39, F40, F62, F66C, F64A, F11, F64B, F51, F52, F36, F39, F37, F50 & F30.

- **Estimator: ExtraTrees Classifier**

Classifier/ Evaluation Metrics	F1-Score	Accuracy	Recall	Precision	AUC/ROC
SVM	0.987 ± 0.0025	0.9892 ± 0.0021	0.9958 ± 0.0031	0.9783 ± 0.0050	0.9974 ± 0.0008
Decision Trees	0.9830 ± 0.0040	0.9860 ± 0.0033	0.9902 ± 0.0053	0.9758 ± 0.0032	0.9867 ± 0.0037
Random Forest	0.9874 ± 0.0007	0.9896 ± 0.0005	0.9973 ± 0.0019	0.9777 ± 0.0020	0.9968 ± 0.0005
AdaBoost	0.9868 ± 0.0017	0.9891 ± 0.0014	0.9961 ± 0.0007	0.9777 ± 0.0029	0.9961 ± 0.0013
Logistic Regression	0.9846 ± 0.0024	0.9873 ± 0.0020	0.9917 ± 0.0040	0.9776 ± 0.0029	0.9957 ± 0.0014
kNN	0.9874 ± 0.0008	0.9896 ± 0.0007	0.9938 ± 0.003	0.9811 ± 0.0033	0.9915 ± 0.0008

Πίνακας 6.9: Evaluation Metrics Using Embedded Method (ExtraTrees)

Ο αλγόριθμος SVM εκπαιδεύτηκε με τα ακόλουθα χαρακτηριστικά: F14, F13, F21, F2, F28, F4, F7, F6, F24, F50, F42, F53, F8, F62, F65A, F65B, F65C, F66A, F66B, F66C, F64A, F64B, F29 & F30.
Ο αλγόριθμος Decision Trees εκπαιδεύτηκε με τα ακόλουθα χαρακτηριστικά: F14, F13, F21, F28, F23, F4, F7, F24, F42, F53, F62, F65B, F65C, F66A, F66B, F64B, F51, F29 & F30.
Ο αλγόριθμος Random Forest εκπαιδεύτηκε με τα ακόλουθα χαρακτηριστικά: F14, F2, F7, F6, F50, F53, F8, F65, F66 & F30.
Ο αλγόριθμος Adaboost εκπαιδεύτηκε με τα ακόλουθα χαρακτηριστικά: F14, F13, F20, F21, F2, F28, F4, F7, F6, F24, F50, F42, F53, F8, F62, F65, F66, F64A, F64B, F29 & F30.
Ο αλγόριθμος kNN εκπαιδεύτηκε με τα ακόλουθα χαρακτηριστικά: F14, F13, F21, F2, F28, F3, F4, F7, F6, F24, F50, F41, F42, F53, F8, F40, F62, F65, F66A, F66B, F64A, F64B & F30.
Ο αλγόριθμος Logistic Regression εκπαιδεύτηκε με τα ακόλουθα χαρακτηριστικά: F2, F7, F24, F43, F53, F8, F39, F62, F65, F66B, F64A, F64B & F50.

- **Estimator: Logistic Regression**

Classifier/ Evaluation Metrics	F1-Score	Accuracy	Recall	Precision	AUC/ROC
SVM	0.9880 ± 0.0031	0.9901 ± 0.0026	0.9952 ± 0.0011	0.9808 ± 0.0055	0.9977 ± 0.0009
Decision Trees	0.9831 ± 0.0015	0.9861 ± 0.0013	0.9896 ± 0.0036	0.9767 ± 0.0049	0.9867 ± 0.0010
Random Forest	0.9886 ± 0.0020	0.9906 ± 0.0017	0.9973 ± 0.0017	0.98 ± 0.0039	0.9974 ± 0.0006
AdaBoost	0.9880 ± 0.0022	0.99012 ± 0.0018	0.9967 ± 0.003	0.9794 ± 0.0019	0.9974 ± 0.0007
Logistic Regression	0.984 ± 0.0031	0.9868 ± 0.0026	0.989 ± 0.0035	0.979 ± 0.0038	0.9963 ± 0.0019
kNN	0.9862 ± 0.0045	0.9886 ± 0.0037	0.9908 ± 0.0051	0.9816 ± 0.0043	0.9913 ± 0.0035

Πίνακας 6.10: Evaluation Metrics Using Embedded Method (Logistic Regression)

Ο αλγόριθμος SVM εκπαιδεύτηκε με τα ακόλουθα χαρακτηριστικά: F13, F18, F21, F2, F48, F7, F24, F32, F50, F45, F41, F43, F42, F60, F53, F8, F26, F39, F40, F65C, F66C, F64A & F11.
Ο αλγόριθμος Decision Trees εκπαιδεύτηκε με τα ακόλουθα χαρακτηριστικά: F13, F21, F2, F48, F7, F24, F32, F50, F45, F41, F43, F42, F4, F62, F65B, F64B, F53, F8, F26, F39, F40, F65C, F66C, F64A & F11.
Ο αλγόριθμος Random Forest εκπαιδεύτηκε με τα ακόλουθα χαρακτηριστικά: F13, F21, F2, F48, F7, F24, F32, F50, F45, F43, F4, F62, F65B, F64B, F53, F8, F26, F39, F40, F65C, F66C, F64A & F66B.
Ο αλγόριθμος Adaboost εκπαιδεύτηκε με τα ακόλουθα χαρακτηριστικά: F13,

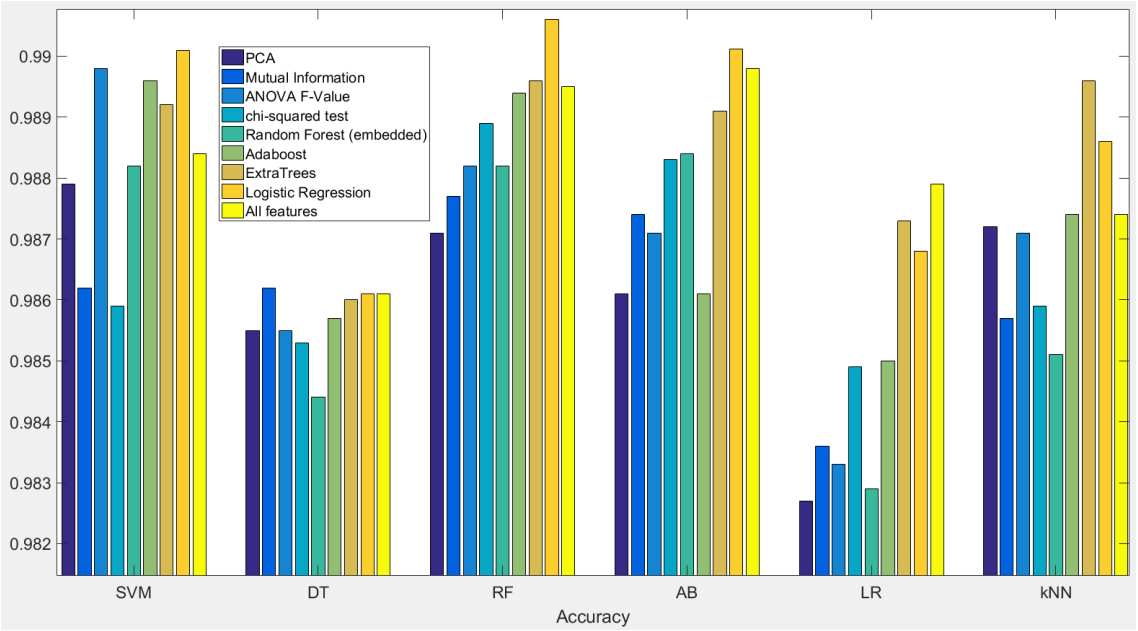
F21, F2, F48, F7, F24, F32, F50, F45, F43, F4, F64B, F53, F8, F26, F39, F40, F65C, F66B, F64A & F11.
Ο αλγόριθμος kNN εκπαιδεύτηκε με τα ακόλουθα χαρακτηριστικά: F13, F21, F28, F62, F65A, F65B, F66A, F66C, F2, F7, F24, F29, F32, F50, F45, F43, F4, F53, F8, F26, F39, F40, F65C, F64A & F64B.
Ο αλγόριθμος Logistic Regression εκπαιδεύτηκε με τα ακόλουθα χαρακτηριστικά: F13, F21, F28, F62, F65A, F65B, F66A, F66C, F42, F2, F7, F24, F29, F32, F50, F45, F43, F4, F53, F8, F26, F39, F40, F65C, F64A & F64B.

Στη συνέχεια, δίνουμε ως είσοδο στους αλγορίθμους Μηχανικής Μάθησης όλα τα χαρακτηριστικά, χωρίς να χρησιμοποιούμε δηλαδή κάποια τεχνικής επιλογής χαρακτηριστικών.

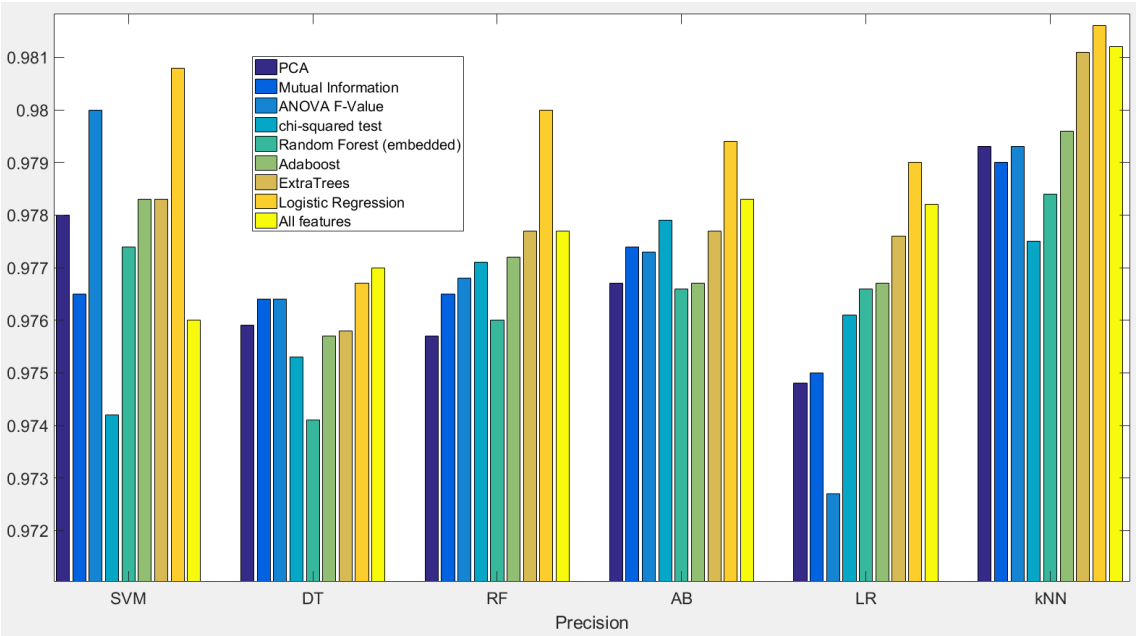
- **All features (without feature selection)**

Classifier/ Evaluation Metrics	F1-Score	Accuracy	Recall	Precision	AUC/ROC
SVM	0.9859 ± 0.0022	0.9884 ± 0.0018	0.9961 ± 0.0042	0.976 ± 0.0018	0.9974 ± 0.0027
Decision Trees	0.9831 ± 0.0019	0.9861 ± 0.0015	0.9893 ± 0.0052	0.9770 ± 0.0039	0.9866 ± 0.0019
Random Forest	0.9873 ± 0.0016	0.9895 ± 0.0013	0.997 ± 0.0016	0.9777 ± 0.0034	0.9973 ± 0.0007
AdaBoost	0.9877 ± 0.0014	0.9898 ± 0.0012	0.9973 ± 0.0014	0.9783 ± 0.0032	0.9978 ± 0.0007
Logistic Regression	0.9853 ± 0.0026	0.9879 ± 0.0022	0.9926 ± 0.0048	0.9782 ± 0.0067	0.9960 ± 0.0018
kNN	0.9847 ± 0.0018	0.9874 ± 0.0014	0.9882 ± 0.0034	0.9812 ± 0.0022	0.9904 ± 0.0015

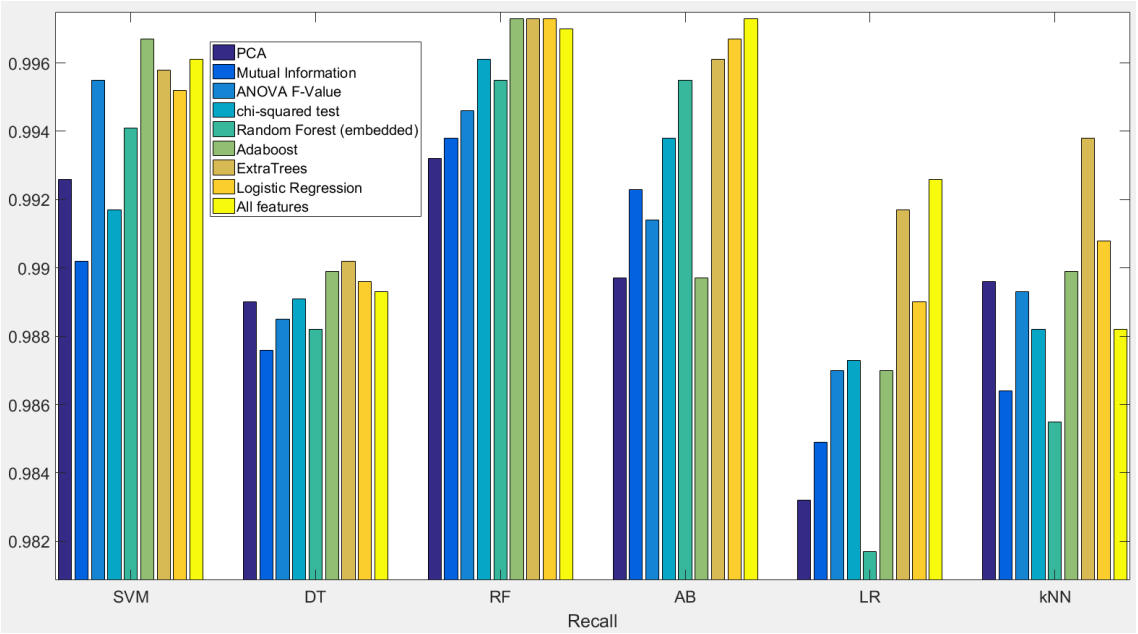
Πίνακας 6.11: All features (without feature selection techniques)



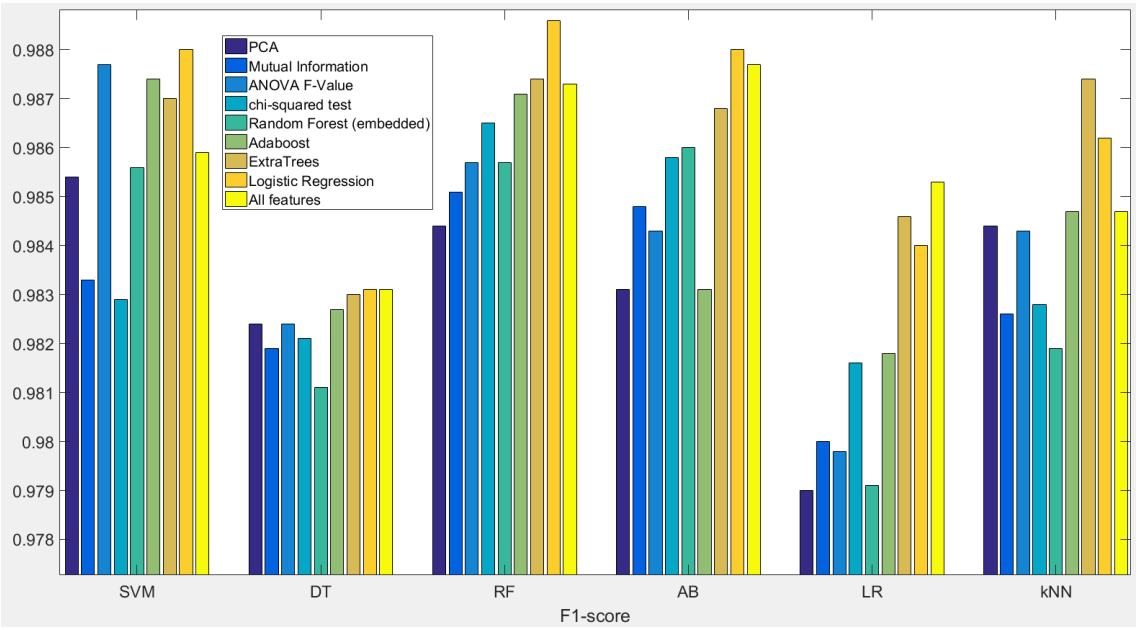
(a) Accuracy



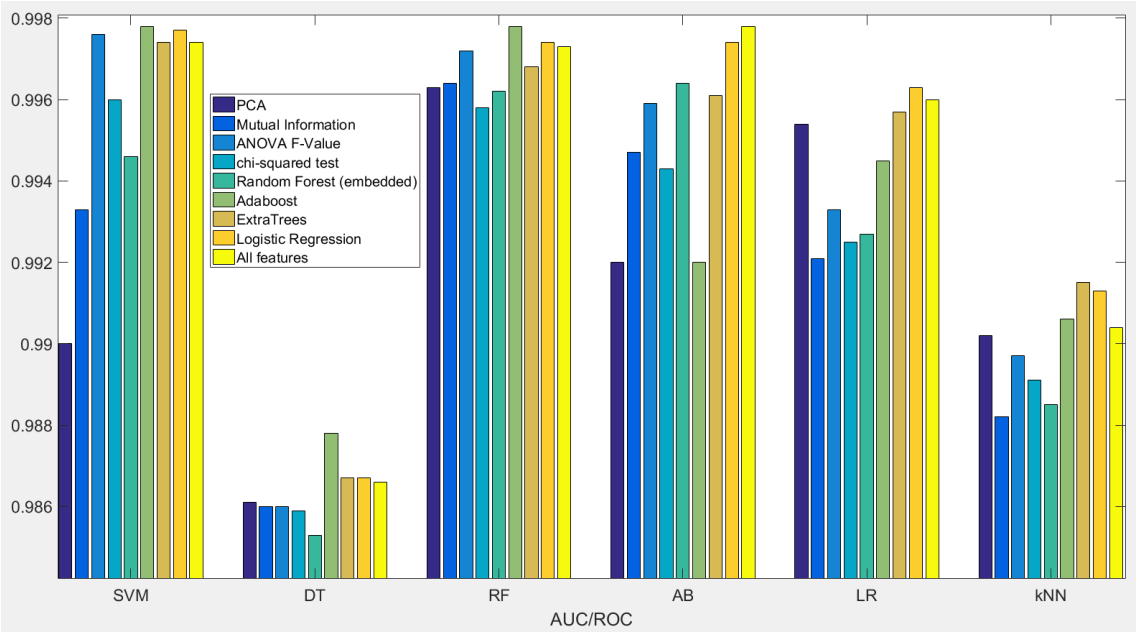
(b) Precision



(c) Recall



(d) F1-score



(e) ROC/AUC

Εικόνα 6.5: Results

Κεφάλαιο 7

Εξαγωγή Αποτελεσμάτων με Τεχνικές Βαθιάς Μηχανικής Μάθησης κάνοντας χρήση μόνο των tweets (χωρίς feature engineering)

Η εξαγωγή χαρακτηριστικών αποτελεί μία διαδικασία, η οποία απαιτεί αρκετό χρόνο για την υλοποίηση και καθίσταται υπολογιστικά πολύπλοκη. Επομένως, σε αντίθεση με το προηγούμενο Κεφάλαιο, που απαιτούσε εξαγωγή χαρακτηριστικών για την κατηγοριοποίηση των χρηστών του Twitter σε humans & bots, στο Κεφάλαιο αυτό κάνουμε χρήση μόνο των tweets και επιχειρούμε μέσω ενός μοντέλου βαθιάς μάθησης να συμπεράνουμε εάν το tweet ανήκει σε human ή bot. Έτσι, στην ενότητα 7.1 παραθέτουμε τα σύνολα δεδομένων, που χρησιμοποιήσαμε. Στην ενότητα 7.2 παρουσιάζεται το μοντέλο, που χρησιμοποιήθηκε. Στην ενότητα 7.3 αναλύονται λεπτομερώς οι μέθοδοι, που χρησιμοποιήθηκαν για την υλοποίηση του μοντέλου. Τέλος, το Κεφάλαιο αυτό ολοκληρώνεται με την ενότητα 7.4, στην οποία παρουσιάζονται τα αποτελέσματα αξιολόγησης του μοντέλου.

7.1 Δεδομένα που χρησιμοποιήθηκαν

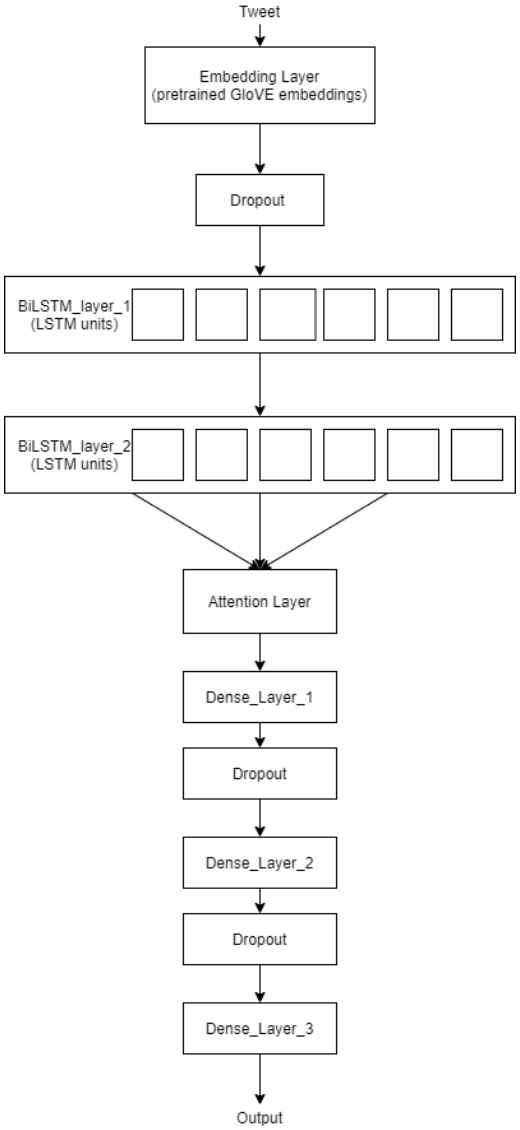
Χρησιμοποιήσαμε το Social Honeypot Dataset, όπως περιγράφηκε στην ενότητα 5.2. Συγκεκριμένα, κρατήσαμε μόνο τα διαφορετικά tweets, τα οποία στη συνέχεια χωρίσαμε σε training, validation & test sets. Τα sets αυτά περιγράφονται στον πίνακα, που ακολουθεί:

set	legitimate tweets	polluters tweets
Training set	1,787,115	1,092,988
Validation set	446,380	273,646
Test set	956,350	586,563

Πίνακας 7.1: Training, validation & test sets of Social Honeypot Dataset

7.2 Μοντέλο

Το μοντέλο που χρησιμοποιήσαμε φαίνεται στην παρακάτω εικόνα.



Εικόνα 7.1: Αρχιτεκτονική Βαθιάς Μάθησης

Για την υλοποίηση της αρχιτεκτονικής χρησιμοποιήθηκε το tensorflow [93] και η βιβλιοθήκη keras [94] της Python.

7.3 Μέθοδοι - Υλοποίηση

Για τη διανυσματική αναπαράσταση των λέξεων έχει χρησιμοποιηθεί η μέθοδος GloVE. Χρησιμοποιήθηκε 1 αρχείο των εκπαιδευμένων λέξεων - διανυσμάτων, που αποτελεί μία συλλογή με συνολικά 1.2m λέξεις, όπου καθεμία από τις λέξεις αναπαριστάνεται με ένα διάνυσμα 50 διαστάσεων.

Αρχικά, φορτώσαμε τις προ-εκπαιδευμένες διανυσματικές αναπαραστάσεις των λέξεων, όπου κάθε λέξη - όρος (token) χαρτογραφήθηκε σε έναν αριθμό (id), προκειμένου να μπορεί το embedding layer στη συνέχεια να το αντιστοιχίσει στη σωστή διανυσματική αναπαράσταση (word embedding). Έτσι, δημιουργήθηκε ένας πίνακας (embedding matrix).

Στο σημείο αυτό, ορίσαμε το μοντέλο μας. Αρχικά, ορίσαμε το Embedding Layer, στο οποίο αρχικοποιήσαμε τα βάρη του δικτύου με τα προ-εκπαιδευμένα διανύσματα λέξεων (embedding matrix). Επίσης, "παγώσαμε" τα επίπεδα εμφύτευσης κατά τη διαδικασία εκπαίδευσης (trainable = False). Αυτό σημαίνει ότι τα επίπεδα εμφύτευσης έμειναν σταθερά, με τις τιμές τους να αντιστοιχούν σε αυτές των διανυσμάτων λέξεων τα οποία χρησιμοποιήθηκαν, για να τα αρχικοποιήσουν. Ο λόγος που έγινε αυτό, ήταν για να μην μετακινηθούν οι λέξεις στον διανυσματικό χώρο σε νέες θέσεις, οι οποίες δεν θα ανταποκρίνονται στον πραγματικό προσανατολισμό όλων των λέξεων.

Χρειάστηκε να ορίσουμε ένα σταθερό μέγεθος ακολουθίας, δηλαδή να ορίσουμε το πλήθος των λέξεων κάθε tweet, που θα αποτελέσει είσοδο στο BiLSTM layer. Αρχικά, βρήκαμε το μεγαλύτερο πλήθος λέξεων, που περιέχει ένα tweet (max_length_tweet). Έπειτα, προσθέσαμε μηδενικά (zero-padding) σε όλα τα υπόλοιπα tweets, προκειμένου το πλήθος λέξεων σε κάθε tweet να ισούται με το πλήθος των λέξεων του max_length_tweet. Στη συνέχεια, εφαρμόσαμε masking στο embedding layer, προκειμένου το μοντέλο μας να "βλέπει" το τελευταίο πραγματικό timestep και να αγνοεί την προσθήκη των μηδενικών. Αυτό το masking προωθείται και στα επόμενα επίπεδα, που είναι σε θέση να το χρησιμοποιήσουν (BiLSTM layers, attention mechanism).

Στη συνέχεια, αντικαταστήσαμε κάθε λέξη (token) με το id της, προκειμένου το embedding layer στη συνέχεια να το αντιστοιχίσει στη σωστή διανυσματική αναπαράσταση (word embedding). Αν μία λέξη δεν βρίσκεται στο αρχείο των προ-εκπαιδευμένων διανυσματικών αναπαραστάσεων των λέξεων, τότε αντικαθιστούμε τη λέξη με τον χαρακτήρα "unk".

Ορίσαμε τα επίπεδα με τα αμφίδρομα LSTM, στα οποία επιλέξαμε μέγεθος 175 για κάθε LSTM, δηλαδή συνολικά 350. Εφαρμόσαμε dropout [95] με συντελεστή 0.5 στις εισόδους των LSTM και 0.25 στις αναδρομικές συνδέσεις τους. Ο λόγος, που εφαρμόζουμε την τεχνική dropout, είναι προκειμένου να διασφαλίσουμε την γενικότητα (generalization) των μοντέλων και να αποφύγουμε φαινόμενα υπερπροσαρμογής (overfitting). Η τεχνική αυτή αποτελεί μια μέθοδο εξομάλυνσης (regularization) σύμφωνα με την οποία, σε κάθε βήμα της εκπαίδευσης τυχαία ένα ποσοστό των νευρώνων απενεργοποιείται από το δίκτυο.

Στη συνέχεια, ορίσαμε τον μηχανισμό προσοχής, ο οποίος υποστηρίζει masking, προκειμένου να δοθεί μεγαλύτερη σημασία στις πιο σημαντικές λέξεις του κάθε tweet.

Τέλος, ορίσαμε τα τρία dense layers με συναρτήσεις ενεργοποίησης relu στα κρυφά επίπεδα και sigmoid στην έξοδο, προκειμένου να λάβουμε την τελική έξοδο (legitimate user ή bot).

Loss: Ως συνάρτηση κόστους ορίσαμε το binary cross entropy loss.

Optimizer: Ως μέθοδο βελτιστοποίησης επιλέξαμε τη μέθοδο Adam [58].

Learning rate: Ως αρχική τιμή του ρυθμού μάθησης επιλέξαμε την τιμή 0.001. Χρησιμοποιήσαμε την τεχνική ReduceLROnPlateau, κατά την οποία μειώνουμε την τιμή του ρυθμού μάθησης, όταν το validation loss έχει σταματήσει να μειώνεται. Στον πίνακα, που ακολουθεί, παρουσιάζονται τα χαρακτηριστικά του ReduceLROnPlateau scheduler.

characteristics	values
monitor	val_loss
factor	0.2
patience	3
min_delta	0.0001
min_lr	0.0002

Πίνακας 7.2: ReduceLROnPlateau

Επιλογή αριθμού εποχών: Η επιλογή του πλήθους των εποχών εκπαίδευσης του δικτύου αποτελεί μία από τις υπερπαραμέτρους, που θα πρέπει να ληφθεί υπόψη. Αυτό συμβαίνει, γιατί αν ο αριθμός των εποχών είναι μεγάλος, ενδέχεται να παρατηρηθούν φαινόμενα υπερπροσαρμογής (overfitting) στο σύνολο εκπαίδευσης, με αποτέλεσμα το εκπαιδευόμενο μοντέλο να χάσει την γενικότητά του (generalization). Για τον λόγο αυτό, χρησιμοποιούμε τη μέθοδο EarlyStopping. Κατά τη μέθοδο αυτή, εκτός από το training & test set, θεωρούμε το validation set, πάνω στο οποίο εξετάζουμε την απόδοση του μοντέλου μας στο τέλος κάθε εποχής, αφού έχει χρησιμοποιηθεί δηλαδή όλο το training set. Με τον τρόπο αυτό, εάν το validation loss έχει σταματήσει να μειώνεται, τότε σταματάμε την εκπαίδευση. Στον πίνακα, που ακολουθεί, παρουσιάζονται τα χαρακτηριστικά της τεχνικής EarlyStopping.

characteristics	values
monitor	val_loss
min_delta	0.0001
patience	4

Πίνακας 7.3: EarlyStopping

Στάθμιση Κλάσεων (class weight): Η στάθμιση των κλάσεων αποτελεί μία τεχνική, που ανήκει στην ευαίσθητη στο κόστος μάθηση (cost sensitive learning). Από την περιγραφή του συνόλου των δεδομένων, που χρησιμοποιήθηκε στο κεφάλαιο αυτό, παρατηρήθηκε ότι οι κλάσεις είναι δυσανάλογες (ανομοιογενές σύνολο δεδομένων). Αυτό θα έχει ως αποτέλεσμα το μοντέλο μας να παρουσιάζει χαμηλές επιδόσεις σε μετρικές αξιολόγησης, καθώς επειδή συναντά πιο συχνά μηνύματα της κλάσης real user, είναι πιθανό να ταξινομεί ένα νέο tweet στην κλάση αυτή. Για τον λόγο αυτό, εισάγουμε βάρη στην αντικειμενική συνάρτηση, των οποίων η εξίσωση για τον υπολογισμό τους δίνεται από τον ακόλουθο τύπο:

$$class\_weight\_0 = \frac{\text{Συνολικό Πλήθος Δεδομένων}}{2 \times (\text{Πλήθος δεδομένων που ανήκουν στην κλάση } 0)} \tag{7.1}$$

$$class\_weight\_1 = \frac{\text{Συνολικό Πλήθος Δεδομένων}}{2 \times (\text{Πλήθος δεδομένων που ανήκουν στην κλάση } 1)} \tag{7.2}$$

Μέθοδος εύρεσης βέλτιστων υπερπαραμέτρων: Συγκεντρωτικά, οι υπερπαράμετροι, που επιλέχθηκαν, παρατίθενται στον ακόλουθο πίνακα:

Hyperparameters	Value
Dropout rate at the embedding layer	0.3
Dropout rate at the lstm inputs	0.5
recurrent dropout	0.25
Number of LSTM units (BiLSTM_layer_1)	175
Number of LSTM units (BiLSTM_layer_2)	175
l2 regularizer (BiLSTM_layer_1)	0.0001
l2 regularizer (BiLSTM_layer_2)	0.0001
Number of units (Dense_layer_1)	128
Number of units (Dense_layer_2)	64
Number of units (Dense_layer_3)	1
Activation function at (Dense_layer_1)	ReLU
Activation function at (Dense_layer_2)	ReLU
Activation function at (Dense_layer_3)	sigmoid
Dropout rate at (Dense_layer_1)	0.4
Dropout rate at (Dense_layer_2)	0.4
Number of epochs	54
Optimizer	Adam
Initial Learning rate	0.001
Loss Function	Binary cross entropy

Πίνακας 7.4: Τιμές υπερπαραμέτρων - Hyperparameter values

7.4 Αποτελέσματα - Μετρικές Αξιολόγησης

Στον παρακάτω πίνακα παρουσιάζονται τα αποτελέσματα αξιολόγησης του μοντέλου μας. Συγκρίνουμε την αξιολόγηση του μοντέλου μας με υπάρχουσες τεχνικές, που έχουν αναπτυχθεί για την ανίχνευση των bots και οι οποίες χρησιμοποιούν το Social Honeypot Dataset.

	architecture	evaluation metrics				
		Precision	**Recall**	**F-Measure**	**Accuracy**	**ROC-AUC**
[34]	*MLP 0*	0.753	0.400	0.563	-	0.775
	MLP 1	0.791	0.756	0.773	-	0.791
	MLP 2	0.794	0.757	0.775	-	0.787
	MLP 3	0.797	0.734	0.764	-	0.784
	MLP 4	0.787	0.737	0.705	-	0.753
[35]	*Tweet Text*	0.865	0.7382	0.7965	0.7668	-
[36]	*Character 1-16 grams TFIDF*	0.795	0.794	0.794	-	-
[43]	*Uni + Bi-gram (Binary)*	0.759	0.727	0.743	-	-
	Uni + Bi-gram (Tf)	0.783	0.767	0.775	-	-
	Uni + Bi-gram (Tfidf)	0.784	0.765	0.775	-	-
	Bi + Tri-gram (Tfidf)	0.797	0.656	0.720	-	-
	Uni & Bi-gram (Tf) + S	0.797	0.744	0.770	-	-
	Content (C) + S	0.778	0.762	0.77	-	-
	C + Uni & Bi-gram (Tf)	0.783	0.757	0.770	-	-
ours	7.1	**0.8365**	**0.8941**	**0.8643**	**0.8261**	**0.8870**

Πίνακας 7.5: Performance comparison among various content polluters detection techniques reported on Social Honeypot Dataset

Κεφάλαιο 8

Επίλογος

Στο Κεφάλαιο αυτό, συνοψίζουμε τα αποτελέσματα της εργασίας και παραθέτουμε προτάσεις για μελλοντική έρευνα σχετικά με την ανίχνευση των bots στα μέσα κοινωνικής δικτύωσης.

8.1 Σύνοψη και Συμπεράσματα

Στην παρούσα διπλωματική εργασία μελετήθηκε το πρόβλημα της ανίχνευσης αυτοματοποιημένων λογαριασμών, γνωστών ως bots, στο Twitter. Το Twitter αποτελεί ένα μέσο κοινωνικής δικτύωσης, που έχει γνωρίσει ραγδαία άνθιση στις μέρες μας λόγω των δυνατοτήτων, που προσφέρει. Ωστόσο, λόγω της άνθισης αυτής χρησιμοποιείται όλο και περισσότερο από αυτοματοποιημένους λογαριασμούς, οι οποίοι έχουν ως κύριο στόχο τη διάδοση ψευδών ειδήσεων, την προώθηση συγκεκριμένων προϊόντων και γενικότερα τη χειραγώγηση των αληθινών χρηστών. Ιδιαίτερα τα τελευταία χρόνια, τα bots έχουν εφεύρει μηχανισμούς, προκειμένου να μιμούνται την ανθρώπινη συμπεριφορά και κατ' επέκταση να μην μπορούν να ανιχνευθούν από αλγορίθμους, που έχουν αναπτυχθεί έως σήμερα. Στην εργασία αυτή προτείναμε δύο μεθόδους ανίχνευσης των bots στο Twitter.

Στην πρώτη μέθοδο, υπολογίσαμε έναν μεγάλο αριθμό χαρακτηριστικών ανά χρήστη, βασιζόμενοι κυρίως σε χαρακτηριστικά, που έχουν χρησιμοποιηθεί στη βιβλιογραφία. Δεν περιοριστήκαμε σε χαρακτηριστικά, όπως αριθμός φίλων και ακολούθων του χρήστη, αλλά υπολογίσαμε χαρακτηριστικά από τα tweets του χρήστη με τεχνικές, που εμπίπτουν στον τομέα της Επεξεργασίας Φυσικής Γλώσσας. Χαρακτηριστικά παραδείγματα αποτελούν η ομοιότητα για κάθε ζεύγος tweets καθώς και ο υπολογισμός unigrams & bigrams. Επίσης, λόγω των μηχανισμών, που έχουν εφεύρει τα bots για να μην ανιχνεύονται από τους υπάρχοντες αλγορίθμους, κρίνεται αναγκαίο να υπολογισθούν νέα χαρακτηριστικά. Βασιζόμενοι, λοιπόν, στη μεθοδολογία που ακολούθησαν οι [32], μοντελοποιήσαμε τη συμπεριφορά των χρηστών του Twitter ως προς το περιεχόμενο των tweets και υπολογίσαμε τα αντίστοιχα χαρακτηριστικά. Έχοντας υπολογίσει τα χαρακτηριστικά, υλοποιήσαμε τεχνικές επιλογής χαρακτηριστικών για την εύρεση του βέλτιστου υποσυνόλου. Στη συνέχεια, προκειμενου να αντιμετωπίσουμε το ανομοιόμορφο σύνολο δεδομένων, υλοποιήσαμε την τεχνική SMOTE+ENN, μία τεχνική που συνδυάζει υπερδειγματοληψία με υποδειγματοληψία. Τέλος, χρησιμοποιήσαμε αλγορίθμους Μηχανικής Μάθησης και αξιολογήσαμε την επίδοσή τους με διάφορες μετρικές.

Στη δεύτερη μέθοδο, προτείναμε μία μέθοδο, η οποία χωρίς καμία πρότερη γνώση των προφίλ των χρηστών (φίλων, ακολούθων, ώρας δημοσίευσης των tweets) και με χρήση μόνο των tweets, εξάγει εάν ένα tweet ανήκει σε αληθινό χρήστη ή σε αυτοματοποιημένο λογαριασμό. Συγκεκριμένα, υλοποιήσαμε ένα βαθύ νευρωνικό δίκτυο, το οποίο αποτελείται από *(a)* στάδιο εμφύτευσης, στο οποίο χρησιμοποιήσαμε προ-εκπαιδευμένα διανύσματα λέξεων GloVE, *(b)* δύο επίπεδα Αμφίδρομων Αναδρομικών Νευρωνικών Δικτύων Μακράς Βραχυπρόθεσμης Μνήμης (BiLSTM), *(c)* μηχανισμό προσοχής & *(d)* 3 επίπεδα dense με 128, 64, 1 νευρώνες το καθένα. Για την αντιμετώπιση του ανομοιόμορφου συνόλου δεδομένων χρησιμοποιήσαμε την τεχνική της στάθμισης κλάσεων, μέσω της οποίας η συνάρτηση κόστους δίνει μεγαλύτερη σημασία στην κλάση με το μικρότερο πλήθος δεδομένων. Αξιολογήσαμε την επίδοση του μοντέλου με τις μετρικές: accuracy, precision, recall, F1-score, ROC/AUC.

Στην πρώτη μέθοδο χρησιμοποιήσαμε το dataset των Cresci et al. [7], ένα δημόσια διαθέσιμο dataset, που έχει χρησιμοποιηθεί από πολλούς ερευνητές τα τελευταία χρόνια. Στη δεύτερη μέθοδο, χρησιμοποιήσαμε το Social Honeypot Dataset.

8.2 Μελλοντικές Επεκτάσεις

Τα αποτελέσματα, που προκύπτουν από την παρούσα εργασία, είναι ιδιαίτερα ικανοποιητικά και ενθαρρυντικά προς την κατεύθυνση της περαιτέρω έρευνας στην ανίχνευση bots στα μέσα κοινωνικής δικτύωσης.

Ένας πρώτος στόχος επέκτασης της εργασίας αυτής θα ήταν η εφαρμογή των πειραμάτων, που χρησιμοποιήθηκαν, και σε άλλα datasets.

Ένας άλλος στόχος αποτελεί η χρήση γεννητικών ανταγωνιστικών δικτύων (GANs) για την παραγωγή fake χαρακτηριστικών, που θα βοηθούσε στη δημιουργία balanced datasets.

Μία άλλη αντιμετώπιση του προβλήματος αυτού θα μπορούσε να είναι η εκπαίδευση ενός μοντέλου μη-επιβλεπόμενης μάθησης. Σε αυτή την περίπτωση τα δεδομένα δεν είναι απαραίτητο να συνοδεύονται από ετικέτες (labels) και μπορούν να ανακτηθούν σε μεγάλους όγκους.

Μία ακόμα κατεύθυνση για μελλοντική ανάπτυξη θα μπορούσε να είναι η εξαγωγή χαρακτηριστικών, που δεν έχουν αποτελέσει αντικείμενο έρευνας έως τώρα. Χαρακτηριστικό παράδειγμα αποτελεί η μελέτη της ζώνης ώρας (timezone) των φίλων και ακολούθων του χρήστη. Αν οι περισσότεροι φίλοι ή ακόλουθοι του χρήστη δημοσιεύουν tweets σε διαφορετική ζώνη ώρας, που σημαίνει ότι ζουν σε άλλη χώρα, τότε ο χρήστης ίσως είναι bot.

Επίσης, στο πλαίσιο της αποτελεσματικής αξιοποίησης των ευρημάτων της εργασίας αυτής, προτείνεται ως ιδέα η ενσωμάτωσή τους στη δημιουργία μίας εφαρμογής, η οποία θα μπορούσε να ανιχνεύσει τα bots σε πραγματικό χρόνο (real time).

Μία άλλη πιθανή κατεύθυνση θα ήταν η εφαρμογή του μοντέλου βαθιάς μάθησης, που αναπτύχθηκε στην παρούσα εργασία, σε άλλα προβλήματα (transfer learning), όπως εξαγωγή δημογραφικών χαρακτηριστικών (ανίχνευση φύλου, ηλικίας, εθνικότητας κτλ.).

Τέλος, μελλοντικός στόχος αποτελεί η ανίχνευση των bots σε άλλα μέσα κοινωνικής δικτύωσης, όπως το Facebook.

Acknowledgments

This work was supported by computational time granted from the National Infrastructures for Research and Technology S.A. (GRNET) in the National HPC facility - ARIS - under project ID pa200501.

Βιβλιογραφία

[1] F. Benevenuto, G. Magno, T. Rodrigues, and V. Almeida, "Detecting spammers on twitter," in *Collaboration, electronic messaging, anti-abuse and spam conference (CEAS)*, vol. 6, p. 12, 2010.

[2] K. Lee, J. Caverlee, and S. Webb, "Uncovering social spammers: social honeypots+ machine learning," in *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pp. 435–442, 2010.

[3] F. Morstatter, L. Wu, T. H. Nazer, K. M. Carley, and H. Liu, "A new approach to bot detection: striking the balance between precision and recall," in *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pp. 533–540, IEEE, 2016.

[4] G. Tavares, S. Mastelini, *et al.*, "User classification on online social networks by post frequency," in *Anais do XIII Simpósio Brasileiro de Sistemas de Informação*, pp. 464–471, SBC, 2017.

[5] P. Andriotis and A. Takasu, "Emotional bots: Content-based spammer detection on social media," in *2018 IEEE International Workshop on Information Forensics and Security (WIFS)*, pp. 1–8, IEEE, 2018.

[6] S. Kudugunta and E. Ferrara, "Deep neural networks for bot detection," *Information Sciences*, vol. 467, pp. 312–322, 2018.

[7] S. Cresci, R. Di Pietro, M. Petrocchi, A. Spognardi, and M. Tesconi, "The paradigm-shift of social spambots: Evidence, theories, and tools for the arms race," in *Proceedings of the 26th international conference on world wide web companion*, pp. 963–972, 2017.

[8] C. Cai, L. Li, and D. Zengi, "Behavior enhanced deep bot detection in social media," in *2017 IEEE International Conference on Intelligence and Security Informatics (ISI)*, pp. 128–130, IEEE, 2017.

[9] H. Ping and S. Qin, "A social bots detection model based on deep learning algorithm," in *2018 IEEE 18th International Conference on Communication Technology (ICCT)*, pp. 1435–1439, IEEE, 2018.

[10] G. Lingam, R. R. Rout, and D. Somayajulu, "Adaptive deep q-learning model for detecting social bots and influential users in online social networks," *Applied Intelligence*, vol. 49, no. 11, pp. 3947–3964, 2019.

[11] K. Lee, B. D. Eoff, and J. Caverlee, "Seven months with the devils: A long-term study of content polluters on twitter," in *Fifth international AAAI conference on weblogs and social media*, 2011.

[12] Z. Gilani, L. Wang, J. Crowcroft, M. Almeida, and R. Farahbakhsh, "Stweeler: A framework for twitter bot analysis," in *Proceedings of the 25th International Conference Companion on World Wide Web*, pp. 37–38, 2016.

[13] F. Wei and U. T. Nguyen, "Twitter bot detection using bidirectional long short-term memory neural networks and word embeddings," in *2019 First IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA)*, pp. 101–109, IEEE, 2019.

[14] B. Wu, L. Liu, Z. Dai, X. Wang, and K. Zheng, "Detecting malicious social robots with generative adversarial networks," *KSII Transactions on Internet and Information Systems (TIIS)*, vol. 13, no. 11, pp. 5594–5615, 2019.

[15] O. Varol, E. Ferrara, C. A. Davis, F. Menczer, and A. Flammini, "Online human-bot interactions: Detection, estimation, and characterization," in *Eleventh international AAAI conference on web and social media*, 2017.

[16] S. Khaled, N. El-Tazi, and H. M. Mokhtar, "Detecting fake accounts on social media," in *2018 IEEE International Conference on Big Data (Big Data)*, pp. 3672–3681, IEEE, 2018.

[17] S. Cresci, R. Di Pietro, M. Petrocchi, A. Spognardi, and M. Tesconi, "Fame for sale: Efficient detection of fake twitter followers," *Decision Support Systems*, vol. 80, pp. 56–71, 2015.

[18] C. A. Davis, O. Varol, E. Ferrara, A. Flammini, and F. Menczer, "Botornot: A system to evaluate social bots," in *Proceedings of the 25th international conference companion on world wide web*, pp. 273–274, 2016.

[19] H. Alvari, G. Beigi, S. Sarkar, S. W. Ruston, S. R. Corman, H. Davulcu, and P. Shakarian, "A feature-driven approach for identifying pathogenic social media accounts," *arXiv preprint arXiv:2001.04624*, 2020.

[20] J. Kaubiyal and A. K. Jain, "A feature based approach to detect fake profiles in twitter," in *Proceedings of the 3rd International Conference on Big Data and Internet of Things*, pp. 135–139, 2019.

[21] D. Niranjan Koggalahewa, Y. Xu, and E. Foo, "Spam detection in social networks based on peer acceptance," in *Proceedings of the Australasian Computer Science Week Multiconference*, pp. 1–7, 2020.

[22] S. Sedhai and A. Sun, "Hspam14: A collection of 14 million tweets for hashtag-oriented spam research," in *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 223–232, 2015.

[23] Q. Gong, Y. Chen, X. He, Z. Zhuang, T. Wang, H. Huang, X. Wang, and X. Fu, "Deepscan: Exploiting deep learning for malicious account detection in location-based social networks," *IEEE Communications Magazine*, vol. 56, no. 11, pp. 21–27, 2018.

[24] R. Wald, T. Khoshgoftaar, and A. Napolitano, "Filter-and wrapper-based feature selection for predicting user interaction with twitter bots," in *2013 IEEE 14th International Conference on Information Reuse & Integration (IRI)*, pp. 416–423, IEEE, 2013.

[25] L. Luo, X. Zhang, X. Yang, and W. Yang, "Deepbot: A deep neural network based approach for detecting twitter bots," in *IOP Conference Series: Materials Science and Engineering*, vol. 719, p. 012063, IOP Publishing, 2020.

[26] S. B. Jr, G. F. Campos, G. M. Tavares, R. A. Igawa, M. L. P. Jr, and R. C. Guido, "Detection of human, legitimate bot, and malicious bot in online social networks based on wavelets," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 14, no. 1s, pp. 1–17, 2018.

[27] E. M. Clark, J. R. Williams, C. A. Jones, R. A. Galbraith, C. M. Danforth, and P. S. Dodds, "Sifting robotic from organic text: a natural language approach for detecting automation on twitter," *Journal of Computational Science*, vol. 16, pp. 1–7, 2016.

[28] C. Zhao, Y. Xin, X. Li, Y. Yang, and Y. Chen, "A heterogeneous ensemble learning framework for spam detection in social networks with imbalanced data," *Applied Sciences*, vol. 10, no. 3, p. 936, 2020.

[29] C. Chen, J. Zhang, X. Chen, Y. Xiang, and W. Zhou, "6 million spam tweets: A large ground truth for timely twitter spam detection," in *2015 IEEE international conference on communications (ICC)*, pp. 7065–7070, IEEE, 2015.

[30] J. P. Dickerson, V. Kagan, and V. Subrahmanian, "Using sentiment to detect bots on twitter: Are humans more opinionated than bots?," in *2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2014)*, pp. 620–627, IEEE, 2014.

[31] S. Cresci, R. Di Pietro, M. Petrocchi, A. Spognardi, and M. Tesconi, "Social fingerprinting: detection of spambot groups through dna-inspired behavioral modeling," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 4, pp. 561–576, 2017.

[32] N. Pasricha and C. Hayes, "Detecting bot behaviour in social media using digital dna compression," in *27th AIAI Irish Conference on Artificial Intelligence and Cognitive Science*, AICS (Artificial Intelligence and Cognitive Science) 2019, 2019.

[33] D. Kosmajac and V. Keselj, "Twitter bot detection using diversity measures," in *Proceedings of the 3rd International Conference on Natural Language and Speech Processing*, pp. 1–8, 2019.

[34] F. Martinelli, F. Mercaldo, and A. Santone, "Social network polluting contents detection through deep learning techniques," in *2019 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–10, IEEE, 2019.

[35] Z. Alom, B. Carminati, and E. Ferrari, "A deep learning model for twitter spam detection," *Online Social Networks and Media*, vol. 18, p. 100079, 2020.

[36] M. Ashour, C. Salama, and M. W. El-Kharashi, "Detecting spam tweets using character n-gram features," in *2018 13th International conference on computer engineering and systems (ICCES)*, pp. 190–195, IEEE, 2018.

[37] M. Fazil and M. Abulaish, "A hybrid approach for detecting automated spammers in twitter," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 11, pp. 2707–2719, 2018.

[38] A. A. Amleshwaram, N. Reddy, S. Yadav, G. Gu, and C. Yang, "Cats: Characterizing automation of twitter spammers," in *2013 Fifth International Conference on Communication Systems and Networks (COMSNETS)*, pp. 1–10, IEEE, 2013.

[39] A. Chakraborty, J. Sundi, S. Satapathy, *et al.*, "Spam: a framework for social profile abuse monitoring," *CSE508 report, Stony Brook University, Stony Brook, NY*, 2012.

[40] M. Mateen, M. A. Iqbal, M. Aleem, and M. A. Islam, "A hybrid approach for spam detection for twitter," in *2017 14th International Bhurban Conference on Applied Sciences and Technology (IBCAST)*, pp. 466–471, IEEE, 2017.

[41] B. Alghamdi, Y. Xu, and J. Watson, "A hybrid approach for detecting spammers in online social networks," in *International Conference on Web Information Systems Engineering*, pp. 189–198, Springer, 2018.

[42] W. Herzallah, H. Faris, and O. Adwan, "Feature engineering for detecting spammers on twitter: Modelling and analysis," *Journal of Information Science*, vol. 44, no. 2, pp. 230–247, 2018.

[43] B. Wang, A. Zubiaga, M. Liakata, and R. Procter, "Making the most of tweet-inherent features for social spam detection on twitter," *arXiv preprint arXiv:1503.07405*, 2015.

[44] D. Stukal, S. Sanovich, R. Bonneau, and J. A. Tucker, "Detecting bots on russian political twitter," *Big data*, vol. 5, no. 4, pp. 310–324, 2017.

[45] S. Volkova and E. Bell, "Identifying effective signals to predict deleted and suspended accounts on twitter across languages," in *Eleventh International AAAI Conference on Web and Social Media*, 2017.

[46] I. Inuwa-Dutse, M. Liptrott, and I. Korkontzelos, "Detection of spam-posting accounts on twitter," *Neurocomputing*, vol. 315, pp. 496–511, 2018.

[47] J. Knauth, "Language-agnostic twitter-bot detection," in *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pp. 550–558, 2019.

[48] P. G. Efthimion, S. Payne, and N. Proferes, "Supervised machine learning bot detection techniques to identify social twitter bots," *SMU Data Science Review*, vol. 1, no. 2, p. 5, 2018.

[49] F. Fathaliani and M. Bouguessa, "A model-based approach for identifying spammers in social networks," in *2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 1–9, IEEE, 2015.

[50] G. Gee and H. Teh, "Twitter spammer profile detection," *Available online: cs229. stanford. edu/proj2010/GeeTeh-Twitter Spammer Profile Detection. pdf*, 2010.

[51] A. H. Wang, "Don't follow me: Spam detection in twitter," in *2010 international conference on security and cryptography (SECRYPT)*, pp. 1–10, IEEE, 2010.

[52] M. Mccord and M. Chuah, "Spam detection on twitter using traditional classifiers," in *international conference on Autonomic and trusted computing*, pp. 175–186, Springer, 2011.

[53] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[54] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794, 2016.

[55] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[56] F. Rosenblatt, "The perceptron: a probabilistic model for information storage and organization in the brain.," *Psychological review*, vol. 65, no. 6, p. 386, 1958.

[57] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[58] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[59] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[60] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[61] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of machine learning research*, vol. 13, no. Feb, pp. 281–305, 2012.

[62] J. Bergstra, D. Yamins, and D. D. Cox, "Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures," 2013.

[63] G. C. Cawley and N. L. Talbot, "On over-fitting in model selection and subsequent selection bias in performance evaluation," *Journal of Machine Learning Research*, vol. 11, no. Jul, pp. 2079–2107, 2010.

[64] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Transactions on knowledge and data engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.

[65] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[66] E. Tuv, A. Borisov, G. Runger, and K. Torkkola, "Feature selection with ensembles, artificial variables, and redundancy elimination," *Journal of Machine Learning Research*, vol. 10, no. Jul, pp. 1341–1366, 2009.

[67] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *Journal of machine learning research*, vol. 3, no. Mar, pp. 1157–1182, 2003.

[68] J. Novaković, "Toward optimal feature selection using ranking methods and classification algorithms," *Yugoslav Journal of Operations Research*, vol. 21, no. 1, 2016.

[69] H. Liu and R. Setiono, "Chi2: Feature selection and discretization of numeric attributes," in *Proceedings of 7th IEEE International Conference on Tools with Artificial Intelligence*, pp. 388–391, IEEE, 1995.

[70] K. Kira and L. A. Rendell, "A practical approach to feature selection," in *Machine Learning Proceedings 1992*, pp. 249–256, Elsevier, 1992.

[71] M. Robnik-Šikonja and I. Kononenko, "Theoretical and empirical analysis of relieff and rrelieff," *Machine learning*, vol. 53, no. 1-2, pp. 23–69, 2003.

[72] C. S. Greene, N. M. Penrod, J. Kiralis, and J. H. Moore, "Spatially uniform relieff (surf) for computationally-efficient filtering of gene-gene interactions," *BioData mining*, vol. 2, no. 1, p. 5, 2009.

[73] C. S. Greene, D. S. Himmelstein, J. Kiralis, and J. H. Moore, "The informative extremes: using both nearest and farthest individuals can improve relief algorithms in the domain of human genetics," in *European Conference on Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics*, pp. 182–193, Springer, 2010.

[74] D. Granizo-Mackenzie and J. H. Moore, "Multiple threshold spatially uniform relieff for the genetic analysis of complex human diseases," in *European Conference on Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics*, pp. 1–10, Springer, 2013.

[75] J. H. Moore and B. C. White, "Tuning relieff for genome-wide genetic analysis," in *European Conference on Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics*, pp. 166–175, Springer, 2007.

[76] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene selection for cancer classification using support vector machines," *Machine learning*, vol. 46, no. 1-3, pp. 389–422, 2002.

[77] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, *et al.*, "Molecular classification of cancer: class discovery and class prediction by gene expression monitoring," *science*, vol. 286, no. 5439, pp. 531–537, 1999.

[78] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification and regression trees*. CRC press, 1984.

[79] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[80] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

[81] M. Kusner, Y. Sun, N. Kolkin, and K. Weinberger, "From word embeddings to document distances," in *International conference on machine learning*, pp. 957–966, 2015.

[82] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of machine Learning research*, vol. 3, no. Jan, pp. 993–1022, 2003.

[83] C. Yang, R. Harkreader, and G. Gu, "Empirical evaluation and new design for fighting evolving twitter spammers," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 8, pp. 1280–1293, 2013.

[84] W. McKinney *et al.*, "Data structures for statistical computing in python," in *Proceedings of the 9th Python in Science Conference*, vol. 445, pp. 51–56, Austin, TX, 2010.

[85] C. Baziotis, N. Pelekis, and C. Doulkeridis, "Datastories at semeval-2017 task 4: Deep lstm with attention for message-level and topic-based sentiment analysis," in *Proceedings of the 11th international workshop on semantic evaluation (SemEval-2017)*, pp. 747–754, 2017.

[86] D. Guo and C. Chen, "Detecting non-personal and spam users on geo-tagged twitter network," *Transactions in GIS*, vol. 18, no. 3, pp. 370–384, 2014.

[87] P.-C. Lin and P.-M. Huang, "A study of effective features for detecting long-surviving twitter spam accounts," in *2013 15th International Conference on Advanced Communications Technology (ICACT)*, pp. 841–846, IEEE, 2013.

[88] S. Loria, "textblob documentation," *Release 0.15*, vol. 2, 2018.

[89] C. J. Hutto and E. Gilbert, "Vader: A parsimonious rule-based model for sentiment analysis of social media text," in *Eighth international AAAI conference on weblogs and social media*, 2014.

[90] S. Cresci, R. Di Pietro, M. Petrocchi, A. Spognardi, and M. Tesconi, "Exploiting digital dna for the analysis of similarities in twitter behaviours," in *2017 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 686–695, IEEE, 2017.

[91] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, *et al.*, "Scikit-learn: Machine learning in python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011.

[92] G. Lemaître, F. Nogueira, and C. K. Aridas, "Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning," *Journal of Machine Learning Research*, vol. 18, no. 17, pp. 1–5, 2017.

[93] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Software available from tensorflow.org.

[94] F. Chollet *et al.*, "keras," 2015.

[95] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.