ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΤΕΧΝΟΛΟΓΙΑΣ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ: Η/Υ
ΕΡΓΑΣΤΗΡΙΟ: HCI

Διπλωματική Εργασία

του φοιτητή του Τμήματος Ηλεκτρολόγων Μηχανικών και Τεχνολογίας Υπολογιστών της Πολυτεχνικής Σχολής του Πανεπιστημίου Πατρών

Κωνσταντίνου Π. Καρποδίνη

Αριθμός Μητρώου: 227522

Θέμα

«Ανάλυση συναισθημάτων σε δεδομένα από το Twitter χρησιμοποιώντας εργαλεία της R και μοντέλα μηχανικής μάθησης.»

Επιβλέπων
Αβούρης Νικόλαος

Αριθμός Διπλωματικής Εργασίας:

Πάτρα, Ιούλιος 2016

ΠΙΣΤΟΠΟΙΗΣΗ

Πιστοποιείται ότι η Διπλωματική Εργασία με θέμα

«Ανάλυση συναισθημάτων σε δεδομένα από το Twitter χρησιμοποιώντας εργαλεία της R και μοντέλα μηχανικής μάθησης.»

Του φοιτητή του Τμήματος Ηλεκτρολόγων Μηχανικών και Τεχνολογίας Υπολογιστών

Κωνσταντίνου Π. Καρποδίνη

Αριθμός Μητρώου: 227522

Παρουσιάστηκε δημόσια και εξετάστηκε στο Τμήμα Ηλεκτρολόγων Μηχανικών και Τεχνολογίας Υπολογιστών στις
……../……../……

Ο Επιβλέπων	Ο Διευθυντής του Τομέα
Αβούρης Νικόλαος Καθηγητής	Χούσος Ευθύμιος Καθηγητής

Αριθμός Διπλωματικής Εργασίας:

Θέμα: « Ανάλυση συναισθημάτων σε δεδομένα από το Twitter χρησιμοποιώντας εργαλεία της R και μοντέλα μηχανικής μάθησης.»

Φοιτητής: Επιβλέπων:

Καρποδίνης Κωνσταντίνος **Αβούρης Νικόλαος**

Περίληψη

Ο στόχος αυτής της διπλωματικής είναι η ταξινόμηση μικρών μηνυμάτων από το Twitter με γνώμονα το συναίσθημα τους, χρησιμοποιώντας τεχνικές εξόρυξης δεδομένων. Τα μηνύματα στο Twitter, ή αλλιώς tweets, όπως είναι ευρέως γνωστά, περιορίζονται στους 140 χαρακτήρες. Αυτός ο περιορισμός εισάγει μια επιπρόσθετη δυσκολία για τους ανθρώπους στο να εκφράσουν τα συναισθήματα τους και συνεπώς η ταξινόμηση αυτού του συναισθήματος σε θετικό ή αρνητικό θα είναι ακόμα πιο δύσκολη.

Γνωστοί αλγόριθμοι επιβλεπόμενης μάθησης όπως ο SVM και ο Naive Bayes χρησιμοποιούνται για να δημιουργηθεί ένα μοντέλο πρόβλεψης. Πριν μπορέσει να δημιουργηθεί το μοντέλο πρόβλεψης, τα δεδομένα πρέπει να προ-επεξεργαστούν από απλό κείμενο σε ένα διάνυσμα συγκεκριμένου μεγέθους χαρακτηριστικών. Τα χαρακτηριστικά αποτελούνται από λέξεις με συναίσθημα και συχνά εμφανιζόμενες λέξεις οι οποίες είναι ικανές να προβλέψουν το γενικότερο συναίσθημα. Έπειτα, ο αλγόριθμος μάθησης εφαρμόζεται σε ένα σύνολο δεδομένων ελέγχου με σκοπό να γίνει αξιολόγηση του μοντέλου.

Table of Contents

1 ΕΙΣΑΓΩΓΗ

1.1 Γενική εισαγωγή

Στην σύγχρονη κοινωνία όλα περιστρέφονται γύρω από την γνώση και την απόκτηση της. Η γνώση μπορεί να προέλθει από διάφορες πηγές, όμως σε κάθε περίπτωση, ακόμα και στην εμπειρική της διάσταση, αποτελείται από ένα σύνολο επεξεργασμένων δεδομένων. Τα δεδομένα αυτά μπορεί να υπάρχουν γύρω μας έτοιμα προς συλλογή, όμως η σωστή ανάλυση και διερμηνεία αυτών είναι που δημιουργεί την πληροφορία που θα μετατραπεί σε τελική γνώση.

Με την ραγδαία εξέλιξη της τεχνολογίας και την ολοένα και μεγαλύτερη χρήση πληροφοριακών συσκευών στην καθημερινότητα μας, βρισκόμαστε αντιμέτωποι με ένα, τεραστίων διαστάσεων, εν δυνάμει κινητήρα γνώσης. Γνώση δεν είναι μόνο η πληροφορία που υπάρχει ήδη στα ηλεκτρονικά βιβλία, ούτε τα εκατομμύρια διαθέσιμα προς ανάγνωση αποτελέσματα οποιασδήποτε μηχανής αναζήτησης στο διαδίκτυο.

Η εξόρυξη γνώσης (data mining) μπορεί να γίνει εφικτή με την ανάλυση οποιουδήποτε πακέτου δεδομένων, αν ακολουθηθεί μια σωστή και μελετημένη διαδικασία, προσαρμοσμένη στο εκάστοτε πεδίο έρευνας. Τέτοια δεδομένα μπορεί να προέρχονται από τεχνητά και μη μέσα, και να αφορούν τόσο τεχνολογικά όσο και φυσικά περιβάλλοντα εφαρμογής.

1.2 Εισαγωγή στην Ανάλυση Δεδομένων

Η Ανάλυση Δεδομένων (*data analysis*), αποτελεί στην ουσία μια διαδικασία συλλογής (παρατήρηση, απόκτηση), επεξεργασίας (καθαρισμός, μετατροπή) και μοντελοποίησης των δεδομένων με στόχο την εξεύρεση χρήσιμης πληροφορίας για την υποστήριξη διαφόρων ειδών λήψεων αποφάσεων (decision-making). Η ¨εξόρυξη γνώσης¨ είναι μια συγκεκριμένη υποκατηγορία ή αλλιώς τεχνική ανάλυσης που επικεντρώνεται στην εύρεση μοντέλων (προτύπων) και την αναζήτηση γνώσης σκοπεύοντας κυρίως στην πρόβλεψη και όχι στην περιγραφή φαινομένων και συμπεριφορών.

Η προγνωστική ανάλυση (predictive analytics), στοχεύει στην εφαρμογή στατιστικών μοντέλων για την πρόβλεψη ή κατηγοριοποίηση δεδομένων, καθώς επίσης και η ανάλυση

κειμένου (text analytics) επιτυγχάνεται με την εφαρμογή στατιστικών εργαλείων σε συνδυασμό με γλωσσολογικές τεχνικές ώστε να εξαχθεί και να κατηγοριοποιηθεί πληροφορία από πηγές με δεδομένα χωρίς δομή (unstructured data).

Ο μεγάλος στατιστικός John Tukey το 1961 όρισε για πρώτη φορά την ευρύτερη επιστήμη της Ανάλυσης Δεδομένων ως τις:

> *"Διαδικασίες για την ανάλυση δεδομένων, τεχνικές για την διερμηνεία των αποτελεσμάτων τέτοιων διαδικασιών, τρόποι οργάνωσης για την συλλογή δεδομένων ώστε να κάνουν την ανάλυση ευκολότερη, πιο συγκεκριμένη και με μεγαλύτερη ακρίβεια, και όλη η μηχανική σε συνδυασμό με τα αποτελέσματα μαθηματικών συναρτήσεων και στατιστικών μεθόδων τα οποία εφαρμόζονται για την ανάλυση των δεδομένων."*

Ο Tukey ήταν ο άνθρωπος που εισήγαγε την λέξη «bit» ως συντόμευση του «binary digit», καθώς εργαζόταν στενά με τον John von Neumann στα αρχικά στάδια σχεδίασης υπολογιστών στα Bell Labs. Ο όρος « bit» χρησιμοποιήθηκε πρώτη φορά σε ένα άρθρο από τον Claude Shannon το 1948 και φυσικά έκτοτε αποτελεί την βασική δομική μονάδα κάθε στοιχείου πληροφορίας σε ηλεκτρονική μορφή.

1.3 Εισαγωγή στην έννοια των Big Data

Η ανάλυση μεγάλου όγκου δεδομένων έχει εξελιχθεί ραγδαία τα τελευταία χρόνια και ενώ παλιότερα γινόταν αναφορά σε αυτή απλά με την ορολογία analytics ή business intelligence, έφτασε κάποια στιγμή που η αγορά ένιωσε την ανάγκη για την χρήση ενός πιο ελκυστικού όρου, γνωστό και ως big data.

Πως όμως διαφέρει η ανάλυση που περιγράφεται από τα big data, με την κλασσική ανάλυση δεδομένων σε κάθε άλλη περίπτωση με μικρό όγκο πληροφορίας ("little data")? Ας δούμε ένα παράδειγμα, έστω ότι έχουμε μια διαρροή σε ένα σωλήνα νερού στον κήπο. Παίρνουμε ένα κουβά και ένα υλικό στεγανοποίησης για να φτιάξουμε το πρόβλημα. Μετά από λίγο, βλέπουμε ότι η διαρροή είναι πολύ μεγαλύτερη οπότε χρειαζόμαστε έναν ειδικό(υδραυλικό) ώστε να φέρει αποτελεσματικότερα εργαλεία. Στο μεταξύ ακόμα χρησιμοποιούμε τον κουβά για να μαζεύουμε το νερό. Έπειτα παρατηρούμε ότι ένα τεράστιο

υπόγειο ρεύμα έχει ανοίξει και πρέπει να διαχειριστούμε εκατομμύρια λίτρα νερού κάθε δευτερόλεπτο.

Δεν χρειαζόμαστε απλά περισσότερους κουβάδες, αλλά μια εντελώς καινούρια προσέγγιση απέναντι στο πρόβλημα επειδή ο όγκος και η ταχύτητα του νερού έχει αυξηθεί δραματικά. Για να εμποδίσουμε την καταστροφή και να μην πλημμυρίσει η πόλη ίσως χρειαστεί να κτιστεί ένα φράγμα το οποίο απαιτεί υπέρμετρη εξειδίκευση και πολύπλοκα συστήματα ελέγχου. Ωστόσο, για να έρθουμε λίγο ακόμα πιο κοντά, ας κάνουμε τα πράγματα χειρότερα και ας πούμε πως παντού αναβλύζει νερό από το πουθενά και όλοι είναι τρομαγμένοι από τις ραγδαίες εξελίξεις. Καλώς ήρθατε στον κόσμο των big data!

Πιο συγκεκριμένα, έστω ότι θέλουμε να κατανοήσουμε την ψυχολογία της αγοράς μέσω των tweets (δεδομένα από το Twitter). Αν θέλαμε λοιπόν να προβλέψουμε την κίνηση της μετοχής της Apple, θα έπρεπε να δούμε την εικόνα που έχει η συγκεκριμένη εταιρία στα ΜΜΕ κοιτώντας στα tweets των media πόσες φορές έχει αναφερθεί η εταιρία ή κάποιο προϊόν της. Την γνώμη των καταναλωτών για την Apple, αν είναι για παράδειγμα ικανοποιημένοι ή όχι από το καινούριο iPhone6. Την αίσθηση των εργαζομένων για την συγκεκριμένη εταιρία, πόσο χαρούμενοι ή δυστυχισμένοι είναι που εργάζονται εκεί, παρατηρώντας tweets και με χαρακτηριστικά τοποθεσίας όσο αφορά στην έδρα της εταιρίας, και τέλος την αντίληψη των επενδυτών για την Apple, τι πιστεύουν οι κορυφαίοι αναλυτές και πόσο πιθανό είναι να επενδύσουν σε αυτήν.

Ένας συνδυασμός όλων αυτών, αθροίζοντας τα επιμέρους συναισθήματα θα μπορούσε να καθορίσει ποια θα είναι η τιμή της μετοχής της Apple στο μέλλον. Κάτι τέτοιο θα μπορούσε φυσικά να σημαίνει δισεκατομμύρια $ κέρδη με μια σωστή πρόβλεψη. Με πιο λαϊκούς όρους, αν μπορούσαμε πραγματικά να καταλάβουμε τι λένε οι διαφορετικοί άνθρωποι σχετικά με μια συγκεκριμένη εταιρία ή προϊόν, θα μπορούσαμε να μετατρέψουμε αυτή τη γνώση σε πολύτιμο επενδυτικό όφελος με μεγάλο κέρδος.

Ωστόσο υπάρχουν τα εξής προβλήματα:

a. Υπάρχουν περισσότερα από 500 εκατομμύρια tweets κάθε μέρα τα οποία δημιουργούνται κάθε δευτερόλεπτο που περνάει. (Αυξημένος όγκος και ταχύτητα – large volume & high velocity)
b. Πρέπει να κατανοήσουμε τι σημαίνει το κάθε tweet, από που προέρχεται, τι είδους άνθρωπος το δημοσιεύει, αν είναι αξιόπιστο ή όχι. (Μεγάλη ποικιλία – large variety)

c. Πρέπει να αναγνωρίσουμε το συναίσθημα, αν κάποιος δηλαδή αναφέρεται θετικά ή αρνητικά σε κάτι, πχ ένα προϊόν ή στην περίπτωση μας ένα iPhone, κτλ. (Αυξημένη πολυπλοκότητα - high complexity)
d. Τέλος, πρέπει να ποσοτικοποιήσουμε το συναίσθημα και να το ανιχνεύσουμε σε πραγματικό χρόνο (real-time applications). (Αυξημένη μεταβλητότητα – high variability)

Σε πολλές περιπτώσεις ωστόσο, ο όρος Big Data αναφέρεται κυριολεκτικά σε πολλά δεδομένα και στην δυσκολία που επιφέρει αυτός ο όγκος στην ενδεχομένως σύνθετη και πολύπλοκη ανάλυση τους. Για πολλούς υφίσταται μόνο σαν όρος του Marketing για την προώθηση ολοκληρωμένων λύσεων και υπηρεσιών ανάλυσης μεγάλης κλίμακας. Συνεπώς, αντιμετωπίζεται απλά σαν ένας περιορισμός της φυσικής μνήμης (RAM), καθώς ο απαιτούμενος χώρος για την συγκράτηση όλων αυτών των δεδομένων στην μνήμη και την απευθείας ανάλυση τους, δεν είναι διαθέσιμος από τυπικά συστήματα, και κυρίως κλασσικούς σταθμούς εργασίας (PCs).

Συνεπώς επιλέγονται διάφορα εργαλεία που επιτρέπουν την παράλληλη ή αλλιώς κατανεμημένη επεξεργασία αυτών των δεδομένων σε πολλαπλούς servers, και συγκεντρώνοντας όλα τα αποτελέσματα μέσω ειδικών μοντέλων επιτυγχάνεται η πλήρης ανάλυση τους και μάλιστα σε λογικούς χρόνους. Η πιο διαδεδομένη επιχειρηματική και open-source λύση είναι το *Hadoop* framework, που παρέχει distributed αποθήκευση τεράστιου όγκου κάθε είδους δεδομένων σε clusters εμπορικού hardware, και δίνει την δυνατότητα της παράλληλης επεξεργασίας όλων αυτών των δεδομένων καθώς το κάθε σημείο αποθήκευσης μπορεί να χρησιμοποιηθεί με ένα έξυπνο αλγόριθμο και ως κόμβος επεξεργασίας τρέχοντας οποιαδήποτε εφαρμογή εξόρυξης.

Αυτό σημαίνει ότι η ανάλυση πρέπει να γίνει σε τυχαία τμήματα δεδομένων ανάλογα με την διαθεσιμότητα τους στο εκάστοτε μηχάνημα επεξεργασίας, οπότε πρέπει να βρεθούν κατάλληλα μοντέλα για την βέλτιστη επιλογή των χαρακτηριστικών σύμφωνα με τα οποία θα επιλεγούν τα δεδομένα αυτά, και έπειτα να συγκριθούν με τα υπόλοιπα επιλεγμένα δείγματα.

1.4 Εισαγωγή στην Ανάλυση Συναισθημάτων

Σύμφωνα με τα προηγούμενα, είναι πολύ βασικό να μπορεί να επιλεγεί ένα μοντέλο ανάλυσης το οποίο να αποδίδει εξίσου καλά σε κάθε δείγμα και ποσότητα δεδομένων, ανεξαρτήτως όγκου και κατανομής. Συνεπώς για να μπορέσει αυτό να δουλέψει σε ευρεία κλίμακα, θα πρέπει πρώτα να εξετάσουμε πως θα βελτιστοποιήσουμε την εφαρμογή του σε ελεγχόμενο περιβάλλον με ένα φυσιολογικό όγκο δεδομένων.

Επέλεξα λοιπόν να ασχοληθώ μεμονωμένα με ένα σύνθετο κομμάτι του πυρήνα της ανάλυσης δεδομένων που είναι η Ανάλυση Συναισθημάτων (sentiment analysis). Όπως δηλώνει και ο ίδιος ο όρος, αυτό το πεδίο έρευνας αφορά την ανάλυση της πολικότητας του συναισθήματος που περιέχεται σε ένα κείμενο ή κομμάτι κειμένου με βάση την χρήση καθαρά γλωσσολογικών χαρακτηριστικών του περιεχομένου των κειμένων. Σε επόμενο κεφάλαιο θα δούμε αναλυτικότερα το εύρος του συγκεκριμένου τομέα, καθώς και το πεδίο εφαρμογής και πως μπορεί να αξιοποιηθεί από μεγάλους οργανισμούς, εταιρίες και όχι μόνο, για την εξόρυξη πολύτιμης γνώσης και πληροφοριών.

Το συναίσθημα μπορεί να ορισθεί με δύο διαφορετικούς τρόπους περιγράφοντας κάθε φορά το πως αισθάνεται κάποιος ή το ποιά είναι η άποψη του για κάτι. Αυτή η έρευνα επικεντρώνεται αποκλειστικά στη μελέτη του συναισθήματος κάποιας άποψης. Αυτές οι απόψεις μπορούν να διαχωριστούν σε 3 κλάσεις: θετικό, αρνητικό και ουδέτερο. Τα tweets ταξινομούνται α κολούθως σε μια από αυτές τις τρεις κλάσεις, ή όπως συνηθίζεται τις περισσότερες φορές μόνο σε positive/negative μετατρέποντας το πρόβλημα της κατηγοριοποίησης σε δυαδικό. Αυτή η ανάλυση είναι γνωστή και με τον ό ρο “opinion mining”.

2 ΑΝΑΛΥΣΗ ΚΕΙΜΕΝΟΥ

2.1 Επεξεργασία φυσικής γλώσσας (NLP)

Η επεξεργασία φυσικής γλώσσας είναι ένα πεδίο της επιστήμης Η/Υ, της τεχνητής νοημοσύνης (artificial intelligence, AI) και της υπολογιστικής γλωσσολογίας (computational linguistics) που αφορά στην αλληλεπίδραση μεταξύ των υπολογιστών και της ανθρώπινης (φυσικής) γλώσσας. Συνεπώς, η NLP σχετίζεται με την περιοχή της αλληλεπίδρασης ανθρώπου μηχανής (human computer interaction, ***hci***). Κύριο μέλημα αυτής της επιστήμης είναι η κατανόηση της φυσικής γλώσσας δίνοντας έτσι τη δυνατότητα στους υπολογιστές να εξάγουν συμπεράσματα από τέτοιου είδους εισόδους, ενώ άλλες εφαρμογές περιλαμβάνουν την παραγωγή αυτής ως έξοδο.

Οι σύγχρονοι NLP αλγόριθμοι βασίζονται στην μηχανική μάθηση (machine learning), και πιο συγκεκριμένα στην στατιστική μηχανική μάθηση. Οι παλαιότερες εφαρμογές σχετικά με την επεξεργασία γλώσσας περιλάμβαναν κυρίως την άμεση καταγραφή με το χέρι μεγάλων συνόλων από κανόνες. Το πρότυπο της μηχανικής μάθησης μας ωθεί αντιθέτως στην χρήση γενικών αλγορίθμων μάθησης – που συχνά βασίζονται, ό μως όχι πάντα, στη στατιστική συμπερασματολογία (statistical inference) – με σκοπό την αυτόματη μάθηση τέτοιων κανόνων μέσα από την ανάλυση μεγάλων σωμάτων κειμένων (***corpora***) τυπικών παραδειγμάτων. Ένα σώμα (***corpus***, plural “corpora”) είναι ένα σετ από κείμενα (ή ενίοτε αυτόνομες προτάσεις) στα οποία έχουν οριστεί με το χέρι οι σωστές τιμές και στην βάση των οποίων πρέπει να γίνει η εκπαίδευση του αλγορίθμου.

Κατά καιρούς έχουν εφαρμοστεί πολλά διαφορετικά είδη αλγορίθμων μηχανικής μάθησης σε προβλήματα NLP. Αυτοί οι αλγόριθμοι λαμβάνουν σαν είσοδο μεγάλα σύνολα από “χαρακτηριστικά” (*features*), τα οποία δημιουργούνται έπειτα από επεξεργασία των δεδομένων εισόδου. Κάποιοι από τους πρώτους αλγορίθμους όπως τα δέντρα-αποφάσεων, παρήγαγαν συστήματα από απόλυτους κανόνες “if-then” παρόμοια με εκείνα που αναφέραμε προηγουμένως με την ανάθεση κανόνων δια χειρός. Ωστόσο, η σύγχρονη έρευνα που εστίαζε περίσσοτερο στα στατιστικά μοντέλα τα οποία λαμβάνουν απλές αποφάσεις βασισμένες σε πιθανότητες καθώς και στον υπολογισμό και ανάθεση πραγματικών τιμών-βάρη, στα χαρακτηριστικά εισόδου. Τέτοια μοντέλα έχουν το πλεονέκτημα ότι μπορούν να εκφράσουν

σχετική βεβαιότητα πολλών πιθανών διαφορετικών απαντήσεων αντί μόνο μιας, παράγοντας έτσι αξιόπιστα αποτελέσματα όταν ένα τέτοιο μοντέλο αποτελεί κομμάτι ενός μεγαλύτερου συστήματος.

Η παρακάτω λίστα περιέχει μερικούς από τους πιο συνηθισμένους τομείς έρευνας στην επιστήμη της NLP. Να σημειωθεί ωστόσο ότι μερικά από αυτά τα πεδία έχουν άμεση εφαρμογή σε πραγματικά προβλήματα, ενώ άλλα αξιοποιούνται κυρίως ως επιμέρους υπό-ενέργειες και χρησιμεύουν για την επίλυση μεγαλύτερων προβλημάτων.

- ***Αυτόματη περίληψη*** (automatic summarization)

  Παράγει μια ευανάγνωστη περίληψη από ένα κομμάτι κειμένου. Χρησιμοποιείται συχνά για να παράγει περιλήψεις από κείμενα γνωστού τύπου, όπως άρθρα ή οικονομικές αναλύσεις από εφημερίδες.

- ***Μηχανική μετάφραση*** (machine translation)

  Αυτόματη μετάφραση κειμένου από μια ανθρώπινη φυσική γλώσσα σε μια άλλη. Αυτή η τεχνική αποτελεί ένα από τα πιο δύσκολα προβλήματα και είναι μέρος από ένα σύνολο προβλημάτων που χαρακτηρίζονται από τον όρο "AI-complete", αναφερόμενα σε όλους τους διαφορετικούς τύπους γνώσης που επεξεργάζονται οι άνθρωποι (γραμματική, λεξιλόγιο, σημασιολογία, γεγονότα, κτλ.) ώστε να λυθούν κατάλληλα.

- ***Παραγωγή φυσικής γλώσσας*** (natural language generation)

  Μετατροπή πληροφορίας από βάσεις δεδομένων σε γλώσσα που μπορεί εύκολα να διαβαστεί από τον άνθρωπο.

- ***Κατανόηση φυσικής γλώσσας*** (natural language understanding)

  Μετατροπή μέρους κειμένου σε κάποια πιο επίσημη αναπαράσταση όπως για παράδειγμα πρώτης-τάξεως λογικές δομές οι οποίες είναι ευκολότερα διαχειρίσιμες από υπολογιστικά προγράμματα. Η εισαγωγή και η δημιουργία ενός γλωσσικού μετα-μοντέλου και οντολογίας είναι αποδοτικά ωστόσο παραμένουν εμπειρικές λύσεις.

- ***Οπτική αναγνώριση χαρακτήρων*** (optical character recognition)

  Αναγνώριση του αντίστοιχου κειμένου δεδομένης μιας εικόνας εισόδου η οποία απεικονίζει τυπωμένους χαρακτήρες.

- ***Απάντηση ερωτήσεων*** (question answering)

  Μια ερώτηση σε ανθρώπινη γλώσσα μπορεί να αναλυθεί ώστε στη συνέχεια να παραχθεί μια απάντηση. Οι τυπικές ερωτήσεις έχουν μια συγκεκριμένη σωστή ή αλλιώς κατάλληλη απάντηση (όμως π.χ. «Ποια είναι η πρωτεύουσα την Αγγλίας?»), ωστόσο

μερικές φορές μπορούμε να θεωρήσουμε και ερωτήσεις ανοικτού τύπου όπως «Ποιο είναι το νόημα της ζωής?» όπως άλλωστε έχουν δείξει πιο πρόσφατες μελέτες.

- ***Ανάλυση συναισθημάτων*** (sentiment analysis)

Η εξαγωγή υποκειμενικής πληροφορίας συνήθως από ένα σύνολο κειμένων, καθώς και η χρήση δημόσιων κριτικών (online reviews) για τον καθορισμό της «πολικότητας» σχετικά με συγκεκριμένα ζητήματα. Αποτελεί εξαιρετικά χρήσιμο εργαλείο για την αναγνώριση των τάσεων και γενικότερα της κοινής γνώμης στα κοινωνικά δίκτυα, με σκοπό στην αξιοποίηση τους για Marketing, όπως έχει άλλωστε προαναφερθεί.

- ***Αναγνώριση ομιλίας*** (speech recognition)

Η αναγνώριση της λεκτικής αναπαράστασης της ομιλίας, δεδομένου ενός ηχητικού ντοκουμέντου ενός ανθρώπου που μιλάει. Αυτό είναι το ακριβώς αντίθετο από την μετατροπή κειμένου σε ομιλία, και αποτελεί με την σειρά του ένα από τα πιο δύσκολα προβλήματα που ανήκουν στην κατηγορία «AI complete». Στην φυσική ομιλία δεν υπάρχουν σχεδόν καθόλου παύσεις ανάμεσα στις διαδοχικές λέξεις, και συνεπώς η τμηματοποίηση της ομιλίας είναι μια απαραίτητη υπό-διαδικασία της αναγνώρισης. Επιπλέον ακόμα και ο ήχος που αντιπροσωπεύουν πολλά συνεχόμενα γράμματα καθώς αναμειγνύονται το ένα μέσα στο άλλο καθιστούν της μετατροπή του αναλογικού σήματος σε διακριτά σύμβολα μια πολύ δύσκολη διεργασία.

2.2 Κατηγοριοποίηση κειμένου (Text classification)

Η κατηγοριοποίηση κειμένου είναι μια διαδικασία κατά την οποία ανατίθενται προκαθορισμένες κατηγορίες σε έγγραφα ελεύθερου κειμένου. Μπορεί να προσφέρει εννοιολογική εικόνα από διάφορες συλλογές κειμένων και έχει σημαντικό ρόλο σε πραγματικές εφαρμογές. Για παράδειγμα, τα νέα συνήθως οργανώνονται σε θεματικές κατηγορίες ή ενότητες, ή ακόμα και γεωγραφικούς κωδικούς. Οι ακαδημαϊκές εργασίες κατηγοριοποιούνται ανά τεχνικό κλάδο και υπό-κλάδους, οι αναφορές ασθενών σε οργανισμούς υγειονομικής περίθαλψης ταξινομούνται βάσει διάφορων χαρακτηριστικών. Μια άλλη ευρέως διαδεδομένη εφαρμογή της κατηγοριοποίησης κειμένου είναι το “spam filtering”, όπου e-mails διαχωρίζονται σε δύο κατηγορίες, spam και non-spam, αντίστοιχα.

Σε αντίθεση με την δημιουργία αυτόματων κανόνων κατηγοριοποίησης με το χέρι, η στατιστική κατηγοριοποίηση κειμένου χρησιμοποιεί μεθόδους μηχανικής μάθησης για να μάθει αντίστοιχους αυτόματους κανόνες που βασίζονται σε κείμενα εκπαίδευσης στα οποία έχουν ανατεθεί ετικέτες από ανθρώπους.

Ένα ελεύ θερο έγγραφο κειμένου αναπαρίσταται τυπικά από σαν ένα διάνυσμα χαρακτηριστικών (feature vector):

$$\hat{x} = \left(x^{(1)}, \dots, x^{(p)}\right)$$

όπου τα χαρακτηριστικά $x^{(i)}$ συνήθως παριστάνουν την παρουσία ή την συχνότητα εμφάνισης των λέξεων , ή των διαφόρων ν -γραμμάτων (n-grams) των λέξεων, συντακτικών ή σημασιολογικών εννοιών, φράσεων, οντοτήτων, κλπ. που υπάρχουν μέσα στο κείμενο. Μια σταθερή μέθοδος για τον υπολογισμό των τιμών των χαρακτηριστικών αυτών για ένα συγκεκριμένο κείμενο d, είναι γνωστή ως προσέγγιση “bag of words”.

Μία επιμέρους έκδοση αυτής της προσέγγισης είναι η TF-IDF (term frequency – inverse document frequency), "συχνότητα όρου – αντίστροφη συχνότητα εγγράφου", που βασίζεται στην ανάθεση ενός βάρος για κάθε όρο. Στην πιο απλή μορφή της παριστάνεται ως:

$$x^{(i)} = \boldsymbol{TF}(i,d) \cdot \boldsymbol{IDF}(i)$$

όπου:

- $\boldsymbol{TF}(i,d)$ (term frequency) είναι ο αριθμός των εμφανίσεων του όρου i στο έγγραφο (document) d.
- $\boldsymbol{IDF}(i) = log\left(\frac{N}{n_i}\right)$, είναι η inverse document frequency, με το N να δηλώνει τον συνολικό αριθμό των εγγράφων σε μια συλλογή, και το n_i τον αριθμό των εγγράφων που περιέχουν τον όρο i.

Για να μπορέσουμε να καταστήσουμε έγγραφα με διαφορετικό μήκος συγκρίσιμα, κάθε διάνυσμα χαρακτηριστικών κανονικοποιείται σε Ευκλείδειο μήκος 1, διαιρώντας κάθε τιμή του διανύσματος με το μήκος $\{ \|x\| = \sqrt{x \cdot x} \}$ του αντίστοιχου διανύσματος. Ωστόσο, τα διανύσματα που προκύπτουν είναι συνήθως μεγάλων διαστάσεων (high-dimensional, 10^4 *έως* 10^5), όμως αρκετά αραιά με μη μηδενικά στοιχεία της τάξης του 10^2 έως 10^3, υποστηρίζοντας έτσι μεθόδους που εκμεταλλεύονται αυτή την αραιότητα (sparsity). Μερικές φορές οι μέθοδοι επιλογής χαρακτηριστικών, για παράδειγμα αυτών τα οποία έχουν την μεγαλύτερη συχνότητα εγγράφου ή το μεγαλύτερο κέρδος αμοιβαίας πληροφορίας, χρησιμοποιούνται για να μειώσουν τις διαστάσεις του χώρου (feature space).

Είναι πολύ χρήσιμο ωστόσο να διαφοροποιήσουμε τα προβλήματα κατηγοριοποίησης κειμένου ανάλογα με τον αριθμό των κλάσεων στις οποίες μπορεί να ανήκει ένα έγγραφο κειμένου. Αν υπάρχουν ακριβώς 2 κλάσεις (κατηγορίες), όπως για παράδειγμα στην περίπτωση spam/non-spam που συναντήσαμε προηγουμένως, τότε αυτό το πρόβλημα ονομάζεται "δυαδικό» (binary) πρόβλημα κατηγοριοποίησης. Αν υπάρχουν περισσότερες από δύο κατηγορίες, έστω θετικό /αρνητικό /ουδέτερο, και ένα έγγραφο κειμένου κατατάσσεται σε ακριβώς μια από τις προαναφερθείσες κατηγορίες, τότε έχουμε ένα multi-class πρόβλημα. Σε πολλές περιπτώσεις βέβαια, ένα έγγραφο μπορεί να έχει περισσότερες από μια κατηγορίες στις οποίες μπορεί να ανατεθεί, π.χ. ένα δημοσιογραφικό άρθρο θα μπορούσε να ανήκει στην υπολογιστική Βιολογία και την μηχανική μάθηση ταυτόχρονα και σε κάποιες επιπλέον

υποκατηγορίες και των δύο. Αυτού του τύπου τα προβλήματα ονομάζονται “multi-label”.

Προβλήματα τέτοιου είδους με περισσότερες από μια κατηγορίες ανάθεσης συνήθως χειρίζονται με παρόμοιο τρόπο, μειώνοντας την τάξη σε k-binary προβλήματα κατηγοριοποίησης, ένα για κάθε κατηγορία. Για κάθε ένα τέτοιο δυαδικό πρόβλημα έπειτα, τα κομμάτια της συλλογής μας που ανήκουν σε αυτή την κατηγορία που εξετάζει το k_i , χαρακτηρίζονται ως θετικά παραδείγματα, ενώ όσα δεν ανήκουν σε αυτή ως αρνητικά. Συνεπώς, σε αυτή την εργασία θα δώσουμε περισσότερο βάρος στα binary classification προβλήματα.

Ένα δείγμα συνόλου εκπαίδευσης για ένα δυαδικό πρόβλημα κατηγοριοποίησης μπορεί να δηλωθεί ως:

$$S = \{(x_1, y_1), \dots, (x_n, y_n)\}$$

όπου n είναι ο αριθμός των παραδειγμάτων εκπαίδευσης, και το $y_i \epsilon \{-1, +1\}$ υποδεικνύει την ετικέτα της κλάσης που ανήκει το αντίστοιχο έγγραφο κειμένου. Χρησιμοποιώντας αυτό το δείγμα εκπαίδευσης, ένας αλγόριθμος επιβλεπόμενης μάθησης στοχεύει στο να υπολογίσει το βέλτιστο κανόνα κατηγοριοποίησης, δηλαδή μια συνάρτηση η οποία αντιστοιχεί (maps) τον p-πολυδιάστατο χώρο των χαρακτηριστικών στο μονοδιάστατο διάνυσμα κλάσεων. Βέλτιστο στην ουσία εννοούμε ένα κανόνα σύμφωνα με τον οποίο μπορούν να κατηγοριοποιηθούν με ακρίβεια δεδομένα εκπαίδευσης αλλά κυρίως ένα γενικευμένο κανόνα ο οποίος αποδίδει εξίσου καλά σε νέα δεδομένα/έγγραφα προς κατηγοριοποίηση εκτός του συνόλου εκπαίδευσης. Πιο επίσημα θα μπορούσαμε να πούμε ότι το δείγμα εκπαίδευσης S θεωρείται τυπικά ένα ανεξάρτητο και ό μοια κατανεμημένο δείγμα από κάποια άγνωστη πιθανοτική κατανομή $P(x, y)$, η οποία μοντελοποιεί την διαδικασία δημιουργίας εγγράφων κειμένου και ετικετών.

2.3 Ανάλυση συναισθημάτων (Sentiment analysis)

Η ***ανάλυση συναισθημάτων*** ή αλλιώς στην περίπτωση μας η εξόρυξη γνώμης, αναφέρεται στην χρήση NLP αξιοποιώντας βασικές μεθόδους κατηγοριοποίησης κειμένου και υπολογιστικής γλωσσολογίας για την αναγνώριση και εξαγωγή υποκειμενικής πληροφορίας από διάφορες πηγές. Η ανάλυση συναισθημάτων εφαρμόζεται ευρέως σε κριτικές (reviews) και στα μέσα κοινωνικής δικτύωσης (social media) για μια ποικιλία εφαρμογών που αφορούν από διαφήμιση μέχρι εξυπηρέτηση πελατών.

Γενικά, η ανάλυση συναισθημάτων στοχεύει στην εξακρίβωση του ύφους ενός ομιλητή ή συγγραφέα σχετικά με κάποιο θέμα ή με την συνολική συνισταμένη πολικότητα του περιεχομένου ενός εγγράφου κειμένου. Το ύφος μπορεί να αφορά στην κρίση ή εκτίμηση του, στη συναισθηματική του κατάσταση (δηλαδή το πως νιώθει κάποιος όταν γράφει) ή το σκόπιμο επικοινωνιακό συναίσθημα (το συναίσθημα εκείνο που θέλει να περάσει στον αναγνώστη όταν διαβάζει το συγκεκριμένο απόσπασμα κειμένου).

Μια από τις κύριες εφαρμογές της ανάλυσης συναισθημάτων είναι η κατηγοριοποίηση της πολικότητας ενός δοσμένου κειμένου σε επίπεδο εγγράφου, πρότασης ή χαρακτηριστικού, δηλαδή αν η γνώμη που εκφράζεται στο συγκεκριμένο τμήμα που αναλύουμε είναι θετική, αρνητική ή ουδέτερη. Πιο αναπτυγμένες τεχνικές, πέρα από τα όρια της αναγνώρισης της πολικότητας, ελέγχουν για παράδειγμα για συναισθηματικές καταστάσεις όπως είναι ο "θυμός", η "λύπη" και η "χαρά".

Η αξιολόγηση ενός συστήματος ανάλυσης συναισθημάτων έγκειται βασικά στο πόσο καλά συμφωνεί με την ανθρώπινη κρίση. Αυτό καθορίζεται κυρίως με κλασσικές μετρήσεις που μας δ είχνουν την ακρίβεια και την αξιοπιστία ενός τέτοιου μηχανισμού. Ωστόσο, σύμφωνα με έρευνες, οι άνθρωποι που ορίζονται ως εκτιμητές σε αντίστοιχα προβλήματα ανάλυσης συνήθως συμφωνούν **79%** των περιπτώσεων.[13]

Συνεπώς, ένα πρόγραμμα με 70% ακρίβεια περίπου, αποδίδει σχεδόν το ίδιο καλά με τους ανθρώπους, ακόμα και αν σε γενικές γραμμές ένα τέτοιο ποσοστό ακρίβειας μπορεί να μην φαίνεται αρκετά εντυπωσιακό. Αυτό εξηγείται με το γεγονός ότι ακόμα και αν ένα πρόγραμμα ήταν 100% "σωστό" όλες τις φορές, οι άνθρωποι και πάλι θα διαφωνούσαν με αυτό κατά 20% περίπου, καθώς όπως είπαμε διαφωνούν σε αυτό το βαθμό κατά μέσο όρο για κάθε απάντηση. Πιο εξελιγμένες μετρήσεις θα μπορούσαν να εφαρμοστούν, ωστόσο η αξιολόγηση της ανάλυσης συναισθημάτων θα παραμένει πάντα ένα πολύ σύνθετο ζήτημα. Για

διεργασίες που αναλύουν το συναίσθημα επιστρέφοντας αποτελέσματα σε μορφή κλίμακας (scale) και όχι δυαδικές αποφάσεις, η συσχέτιση (correlation) αποτελεί πολύ καλύτερη μέθοδο για την μέτρηση της ακρίβειας επειδή λαμβάνει υπόψη της το πόσο κοντά προκύπτουν οι τιμές της πρόβλεψης σε σχέση με τον τελικό στόχο.

Η άνθιση των κοινωνικών μέσων (social media), όπως είναι τα blogs και τα κοινωνικά δίκτυα (social networks), έχει τροφοδοτήσει με πολύ ενδιαφέρον τον χώρο γύρω από τον κλάδο της ανάλυσης συναισθημάτων. Σε αυτή την κατεύθυνση έχει βοηθήσει σημαντικά η εξέλιξη του διαδικτύου, με τη μετάβαση στη νέα γενιά του παγκοσμίου ιστού, γνωστή ως web 2.0, η οποία βασίζεται στην όλο και μεγαλύτερη δυνατότητα των χρηστών του διαδικτύου να μοιράζονται πληροφορίες και να συνεργάζονται διαδραστικά. Αυτή η νέα γενιά είναι μια δυναμική διαδικτυακή πλατφόρμα στην οποία μπορούν να αλληλοεπιδρούν χρήστες χωρίς εξειδικευμένες γνώσεις σε θέματα υπολογιστών και δικτύων.

Με τον πολλαπλασιασμό λοιπόν των κριτικών, των αξιολογήσεων, των συστάσεων αλλά και πολλών άλλων μορφών online έκφρασης, η διαδικτυακή γνώμη έχει μετατραπεί σε ένα είδος ψηφιακού νομίσματος για επιχειρήσεις που ψάχνουν να προωθήσουν τα προϊόντα τους, να ανακαλύψουν νέες ευκαιρίες και να διαχειριστούν βέλτιστα την φήμη τους στην αγορά. Καθώς οι επιχειρήσεις ψάχνουν τρόπους για να αυτοματοποιήσουν τη διαδικασία γύρω από το φιλτράρισμα του θορύβου, την κατανόηση των συζητήσεων αναγνωρίζοντας το σχετικό περιεχόμενο και τη λήψη κατάλληλων δράσεων σχετικά με αυτό, πολλοί στρέφονται στο τομέα της ανάλυσης συναισθημάτων. Αν το web 2.0 αφορούσε απλά την δημοκρατικοποίηση των δημοσιεύσεων στον ιστό, τότε το επόμενο στάδιο του διαδικτύου ίσως πρέπει να βασίζεται στην δημοκρατικοποίηση της εξόρυξης γνώσης από το περιεχόμενο όλου αυτού του όγκου πληροφορίας που δημοσιεύεται.

Ένα βήμα πιο κοντά στον στόχο αυτό επιτυγχάνεται μέσω της έρευνας. Πολλές ερευνητικές ομάδες σε πανεπιστήμια σε όλο τον κόσμο, αυτό το διάστημα επικεντρώνονται στην κατανόηση της δυναμικότητας του συναισθήματος στις e-κοινωνίες μέσω των τεχνικών της ανάλυσης συναισθημάτων. Το πρόβλημα είναι ότι οι περισσότεροι αλγόριθμοι χρησιμοποιούν απλούς όρους για να εκφράσουν το συναίσθημα σχετικά με κάποιο προϊόν ή υπηρεσία. Ωστόσο, πολιτισμικοί παράγοντες, γλωσσολογικές αποχρώσεις και ιδιαιτερότητες καθιστούν εξαιρετικά δύσκολη την μετατροπή ενός τμήματος γραπτού κειμένου σε κάποιο απλό θετικό ή αρνητικό συναίσθημα. Το γεγονός ότι ακόμα και οι άνθρωποι συχνά διαφωνούν

για το συναίσθημα των κειμένων, αποδεικνύει το πόσο δύσκολο έργο είναι για ένα υπολογιστή να το εξάγει σωστά. Όσο πιο σύντομο είναι το κείμενο, τόσο πιο δύσκολο γίνεται.

Αν και τα κομμάτια κειμένου μικρού μήκους μπορεί να είναι πρόβλημα, η ανάλυση των συναισθημάτων στα πλαίσια του microblogging έχει δείξει ότι το Twitter μπορεί να φανεί ως ένας αξιόπιστος online δείκτης του πολιτικού συναισθήματος. Μελέτες έχουν δείξει ότι το πολιτικό συναίσθημα των tweets φανερώνει πολύ στενή ανταπόκριση στις πολιτικές θέσεις και απόψεις των πολιτικών και των κομμάτων, επιδεικνύοντας έτσι ότι το περιεχόμενο των μηνυμάτων του Twitter αδιαμφησβήτητα αντανακλά το offline πολιτικό σκηνικό.

3 ΜΗΧΑΝΙΚΗ ΜΑΘΗΣΗ

3.1 Εισαγωγή στην Μηχανική Μάθηση

Η **μηχανική μάθηση** (machine learning) είναι μια περιοχή της τεχνητής νοημοσύνης η οποία αφορά αλγορίθμους και μεθόδους που επιτρέπουν στους υπολογιστές να «μαθαίνουν». Με τη μηχανική μάθηση καθίσταται εφικτή η κατασκευή *προσαρμόσιμων* (adaptable) προγραμμάτων υπολογιστών τα οποία λειτουργούν με βάση την αυτοματοποιημένη ανάλυση συνόλων δεδομένων και όχι τη διαίσθηση των μηχανικών που τα προγραμμάτισαν. Παραδείγματα εφαρμογών αποτελούν το spam filtering, η οπτική αναγνώριση χαρακτήρων, οι μηχανές αναζήτησης και η υπολογιστική όραση. Η μηχανική μάθηση βασίζεται σημαντικά στην στατιστική, αφού και τα δύο πεδία χρησιμοποιούν ανάλυση δεδομένων, επικαλύπτονται όμως και με τη εξόρυξη δεδομένων (data mining), ωστόσο η δεύτερη εστιάζει περισσότερο στη διερευνητική ανάλυση δεδομένων.

Το 1959, ο πρωτοπόρος σχεδιαστής παιχνιδιών *Arthur Samuel* όρισε ως μηχανική μάθηση:

> "*Το πεδίο μελέτης που δίνει στους υπολογιστές τη δυνατότητα να μαθαίνουν χωρίς να έχουν προγραμματιστεί ρητά γι' αυτό το σκοπό*".

Το 1997 ο *Tom M. Mitchell* έδωσε ένα πιο επίσημο ορισμό ο οποίος χρησιμοποιείται ευρέως:

> "*Ένα πρόγραμμα υπολογιστή λέμε ότι μαθαίνει από εμπειρία E ως προς μια κλάση εργασιών T και ένα μέτρο επίδοσης P, αν η επίδοσή του σε εργασίες της κλάσης T, όπως αποτιμάται από το μέτρο P, βελτιώνεται με την εμπειρία E*".

Οι αλγόριθμοι μηχανικής μάθησης κατηγοριοποιούνται ανάλογα με το επιθυμητό αποτέλεσμα του αλγορίθμου. Οι συνηθέστερες κατηγορίες είναι οι εξής:

- Επιτηρούμενη μάθηση, *επιβλεπόμενη μάθηση* ή μάθηση με επίβλεψη (supervised learning), όπου ο αλγόριθμος κατασκευάζει μια συνάρτηση για να απεικονίσει δεδομένες εισόδους σε γνωστές ή επιθυμητές εξόδους (σύνολο εκπαίδευσης), με απώτερο όμως στόχο τη γενίκευση της συνάρτησης αυτής και για εισόδους με άγνωστη έξοδο (σύνολο ελέγχου).
- *Μη επιβλεπόμενη μάθηση* ή μάθηση χωρίς επίβλεψη (unsupervised learning), όπου ο αλγόριθμος κατασκευάζει ένα μοντέλο για κάποιο σύνολο εισόδων χωρίς να γνωρίζει τις εξόδους για το σύνολο εκπαίδευσης.

3.2 Τεχνικές Μηχανικής Μάθησης

3.2.1 Επιβλεπόμενη Μάθηση

Η διαδικασία μηχανικής μάθησης σύμφωνα με την οποία εξάγεται μια συνάρτηση από κατηγοριοποιημένα δεδομένα ονομάζεται "*επιβλεπόμενη μάθηση*" (ή όπως είναι γνωστή στον ευρύτερο κλάδο, **supervised learning**). Τα δεδομένα εκπαίδευσης αποτελούνται από ένα σύνολο από στιγμιότυπα κατάλληλα για την εκπαίδευση του αλγορίθμου. Στην επιβλεπόμενη μάθηση, κάθε στιγμιότυπο είναι ένα ζευγάρι το οποίο αποτελείται από ένα αντικείμενο εισόδου (συνήθως ένα *διάνυσμα*) και μια δεδηλομένη τιμή εξόδου η οποία τυπικά μπορεί να ονομαστεί *σήμα ελέγχου*. Ένας αλγόριθμος επιβλεπόμενης μάθησης αναλύει τα δεδομένα εκπαίδευσης και δημιουργεί μια συνάρτηση ικανή να χρησιμοποιηθεί για να κατηγοριοποιήσει νέα δείγματα. Ένα βέλτιστο σενάριο θα επέτρεπε στον αλγόριθμο να καθορίζει σωστά τις ετικέτες κλάσης σε άγνωστα περιστατικά. Αυτή η ανάγκη απαιτεί τη γενικευμένη κατασκευή ενός γενικότερου αλγορίθμου έτσι ώστε από τα δεδομένα εισόδου, με λογικό τρόπο, έτσι ώστε να αποδίδει εξίσου καλά σε άγνωστα συμβάντα.

Για να λύσουμε ένα πρόβλημα επιβλεπόμενης μάθησης, πρέπει να ακολουθήσουμε τα παρακάτω βήματα:

1. Να καθορίσουμε τον τύπο των παραδειγμάτων εκπαίδευσης. Πριν κάνουμε οτιδήποτε άλλο, πρέπει να αποφασίσουμε τι είδους δεδομένα θα χρησιμοποιηθούν σαν σύνολο εκπαίδευσης (*training set*). Στην περίπτωση την ανάλυσης γραφικού χαρακτήρα για παράδειγμα, αυτό θα μπορούσε να αποτελείται από ένα απλό χαρακτήρα γραμμένο με το χέρι, μια ολόκληρη λέξη, ή ακόμα και μια ολόκληρη χειρόγραφη πρόταση. Στην περίπτωση της ανάλυσης συναισθημάτων, αυτά αποτελούνται από δείγματα κειμένων με θετικές ή αρνητικές ετικέτες.
2. Στην συνέχεια, πρέπει να σκεφτούμε για το πως θα συγκεντρώσουμε όλο το επιθυμητό σύνολο των δεδομένων εκπαίδευσης. Αυτό το σύνολο θα πρέπει να είναι αντιπροσωπευτικό της χρήσης της συνάρτησης στον πραγματικό κόσμο. Συνεπώς, ένα σύνολο από α ντικείμενα εισόδων συγκεντρώνεται μαζί με το αντίστοιχο σύνολο επιθυμητών και δεδηλωμένων αποτελεσμάτων, όπως έχει συλλεχθεί είτε από ειδικούς είτε από κατάλληλες μετρήσεις – αναζητήσεις.

3. Έπειτα, θα πρέπει να καθοριστεί η αναπαράσταση των χαρακτηριστικών εισόδου της συνάρτησης εκπαίδευσης. Η ακρίβεια της συνάρτησης εξαρτάται σημαντικά από τη δομή των αντικειμένων εισόδου. Τυπικά, ένα αντικείμενο εισόδου μετατρέπεται σε ένα διάνυσμα χαρακτηριστικών το οποίο περιλαμβάνει έναν αριθμό από χαρακτηριστικά τα οποία περιγράφουν ικανοποιητικά όλο το αντικείμενο. Ο αριθμός των αντικειμένων δεν θα πρέπει να είναι πολύ μεγάλος, γιατί τότε διογκώνεται η διάσταση του παραγόμενου χώρου χαρακτηριστικών, ό μως θα πρέπει να περιέχει αρκετή πληροφορία, τόση ώστε να μπορεί να προβλεφθεί με ακρίβεια η έξοδος του συστήματος.
4. Το επόμενο βήμα είναι η επιλογή της δομής της συνάρτησης και του αλγορίθμου εκμάθησης, για παράδειγμα ο εκάστοτε μηχανικός μπορεί να επιλέξει να χρησιμοποιήσει είτε δέντρα εκμάθησης, είτε τον support vector machines είτε τον naïve Bayes αλγόριθμο. Συνήθως αυτή η επιλογή γίνεται μετά από επάλληλες δοκιμές και ανάλυση της ακρίβειας και προσαρμογής σε τυχαία δεδομένα.
5. Ολοκληρώνοντας την διαδικασία, ο αλγόριθμος θα πρέπει να τρέξει με όλα τα συγκεντρωμένα δεδομένα εκπαίδευσης συγκεντρωμένα για να δημιουργήσουμε την συνάρτηση αντιστοίχισης. Κάποιοι αλγόριθμοι επιβλεπόμενης μάθησης απαιτούν από τον χρήστη να καθορίσει ορισμένες παραμέτρους ελέγχου. Αυτές οι παράμετροι μπορούν να προσαρμοστούν είτε με βελτιστοποίηση της απόδοσης μετά από δοκιμές σε ένα υποσύνολο δεδομένων (validation set), είτε με cross-validation, μια τεχνική που θα αναλυθεί στην πορεία.
6. Στο τέλος, θα πρέπει να αξιολογηθεί η απόδοση του μοντέλου εκμάθησης. Αφού προσαρμόσουμε όλες τις παραμέτρους και εκπαιδεύσουμε τον αλγόριθμο μας, θα πρέπει να υπολογίσουμε την απόδοση της συνάρτησης κατηγοριοποίησης εφαρμόζοντας την σε ένα δείγμα δεδομένων ελέγχου (test set), το οποίο μάλιστα θα πρέπει να είναι διαφορετικό από το training set.

Υπάρχει μια ευρεία ποικιλία διαθέσιμων αλγορίθμων επιβλεπόμενης μάθησης, ο καθένας με τα προτερήματα και τις αδυναμίες του φυσικά. Δεν υπάρχει ωστόσο κανένας από αυτούς ο οποίος να αποδίδει εξίσου καλά σε όλα τα προβλήματα επιβλεπόμενης μάθησης.

Η γενικευμένη λογική που ακολουθεί αυτή η προσέγγιση αναλύεται ακριβώς παρακάτω, και μπορεί να προσαρμοστεί με την χρήση διαφόρων αλγορίθμων μετά από ειδική μελέτη και προσαρμογή στο εκάστοτε πρόβλημα.

Πιο συγκεκριμένα, δεδομένου ενός συνόλου *N* από παραδείγματα εκπαίδευσης της μορφής $\{(x_1, y_1), \dots, (x_n, y_n)\}$, τέτοια ώστε το x_i να είναι το διάνυσμα χαρακτηριστικών του i-στου παραδείγματος και το y_i να είναι η ετικέτα κατηγορίας του, ένας αλγόριθμος μάθησης υπολογίζει μια συνάρτηση $g : X \to Y$, όπου το *X* είναι ο χώρος των δεδομένων εισόδου (input space) και Y είναι ο χώρος των προβλεπόμενων τιμών εξόδου (output space). Η συνάρτηση g αποτελεί ένα αντικείμενο ενός χώρου από πιθανές συναρτήσεις G, που ονομάζεται συνήθως "*υποθετικός χώρος*". Μερικές φορές είναι πιο βολικό να παρουσιάζουμε την g ως μια συνάρτηση βαθμολόγησης $f : X \times Y \to \mathbb{R}$, έστω ότι F είναι ο χώρος όλων την συναρτήσεων βαθμολόγησης, έτσι ώστε η g να ορίζεται σαν την επιστρεφόμενη τιμή y που δίνει την υψηλότερη βαθμολογία στην συνάρτηση:

$$g(x) = \arg\max_y f(x, y)$$

Στα μαθηματικά, τα *ορίσματα των μεγίστων* ή αλλιώς "***the arguments of the maxima***" (σε συντομογραφία *arg max* ή *argmax*), αντιπροσωπεύουν τα σημεία ενός τομέα μιας συνάρτησης στα οποία οι συναρτησιακές τιμές μεγιστοποιούνται.

Αν και οι G και F μπορεί να αποτελούν οποιονδήποτε χώρο συναρτήσεων, οι περισσότεροι αλγόριθμοι μάθησης είναι πιθανοτικά μοντέλα όπου η g παίρνει την μορφή ενός μοντέλου δεσμευμένης πιθανότητας:

$$g(x) = \mathcal{P}(y|x)$$

ή αντίστοιχα η f παίρνει τη μορφή ενός μοντέλου με από κοινού πιθανότητες που περιγράφεται από την σχέση που ακολουθεί:

$$f(x, y) = \mathcal{P}(x, y)$$

Για παράδειγμα, ο *naïve Bayes* και η γραμμική διαχωριστική ανάλυση (*linear discriminant analysis*) είναι μοντέλα με από κοινού πιθανότητες, ενώ η λογιστική παλινδρόμηση (*logistic regression*) είναι ένα μοντέλο δεσμευμένης πιθανότητας.

3.2.2 Μη Επιβλεπόμενη Μάθηση

Η *μη επιβλεπόμενη μάθηση* (**unsupervised learning**) είναι μια διεργασία μηχανικής μάθησης κατά την οποία εξάγεται μια συνάρτηση για να περιγράψει κρυφές δομές από δεδομένα χωρίς ετικέτες. Εφόσον τα παραδείγματα που δίνονται στον μηχανισμό εκμάθησης είναι άγνωστης κατηγορίας, δεν υπάρχει λάθος, σφάλμα καθώς και ούτε κάποιο σήμα αξιολόγησης ώστε να γίνει εκτίμηση της πιθανής λύσης. Αυτό είναι άλλωστε και που ξεχωρίζει τις μη επιβλεπόμενες διαδικασίες από τις επιβλεπόμενες τεχνικές ανάλυσης.

Η μη επιβλεπόμενη μάθηση είναι στενά συνδεδεμένη με το πρόβλημα της εκτίμησης πυκνότητας στην στατιστική. Ωστόσο, η μη επιβλεπόμενη μάθηση εμπλέκει και πολλές άλλες τεχνικές σύμφωνα με τις οποίες ψάχνει να συνοψίσει και να εξηγήσει χαρακτηριστικά κλειδιά των δεδομένων. Πολλές μέθοδοι που αναμιγνύονται σε αυτές τις τεχνικές βασίζονται σε μεθόδους εξόρυξης γνώσης που χρησιμοποιούνται κατά την προ επεξεργασία των δεδομένων. Την πιο σημαντική προσέγγιση της μη επιβλεπόμενης μάθησης αποτελεί:

- Η συσταδοποίηση ή αλλιώς ***clustering*** (π.χ. k-means, mixture models, hierarchical clustering, κτλ.)

Στην επιβλεπόμενη μάθηση (supervised learning) μας δίνεται ένα σύνολο δεδομένων με τις αντίστοιχες κλάσεις-ετικέτες κάθε εγγραφής. Στόχος είναι η δημιουργία ενός μοντέλου, το οποίο να μπορεί να κατηγοριοποιήσει νέα δεδομένα σε κάποια από τις προϋπάρχουσες κλάσεις. Αντίθετα, στη μη επιβλεπομένη μάθηση (unsupervised learning) μας δίνεται ένα σύνολο δεδομένων, χωρίς τις αντίστοιχες κλάσεις-ετικέτες κάθε εγγραφής και στόχος είναι η χρήση κάποιου αλγορίθμου, ώστε αυτόματα να ανακαλύψουμε κάποια ενδεχομένως ενδιαφέρουσα δομή των δεδομένων. Για παράδειγμα, η συσταδοποίηση είναι μια από τις τεχνικές μη επιβλεπόμενης μάθησης που χρησιμοποιείται ευρέως για αυτόν τον σκοπό. Δοθέντων κάποιων δεδομένων χωρίς κλάσεις, οι αλγόριθμοι συσταδοποίησης ομαδοποιούν τα δεδομένα σε συστάδες, έτσι ώστε εγγραφές, οι οποίες ανήκουν στην ίδια συστάδα, να έχουν όμοια ή παραπλήσια χαρακτηριστικά.

Στο πρόβλημα της *συσταδοποίησης* μας δίνεται ένα σύνολο δεδομένων, χωρίς τις αντίστοιχες κλάσεις και χρειαζόμαστε κάποιον αλγόριθμο, ο οποίος θα ομαδοποιήσει αυτόματα τα δεδομένα σε συστάδες (**clusters**). Οι συστάδες που δημιουργούνται θέλουμε να

διαχωρίζουν όσο το δυνατόν πιο ορθά τα δεδομένα. Αυτό πρακτικά σημαίνει ότι μια συστάδα θέλουμε να απαρτίζεται από αντικείμενα, όπου κάθε αντικείμενο είναι πιο κοντά σε κάθε άλλο αντικείμενο της ίδιας συστάδας απ' ότι σε κάποιο άλλο αντικείμενο διαφορετικής συστάδας.

Ο αλγόριθμος **k-means** ξεκινάει με *k* τυχαία σημεία, τα οποία ονομάζονται κεντροειδή της συστάδας και δηλώνουν το κέντρο βάρους της συστάδας. Το *k* υποδηλώνει το πόσες συστάδες θέλουμε να δημιουργήσει ο αλγόριθμος καθώς εκτελεί επαναληπτικά δύο βήματα. Το πρώτο βήμα αφορά την ανάθεση σε κάποια συστάδα, ενώ το δεύτερο βήμα αφορά τον επαναπροσδιορισμό και τη μετατόπιση του κεντροειδούς κάθε συστάδας.

Πιο αναλυτικά, όσον αφορά στο πρώτο βήμα, δηλαδή την ανάθεση σε κάποια συστάδα, ο αλγόριθμος εξετάζει κάθε στιγμιότυπο σε σχέση με τα κεντροειδή των συστάδων. Με χρήση κάποιου μέτρου απόστασης, αναθέτει το εξεταζόμενο στιγμιότυπο στη συστάδα, της οποίας το κεντροειδές είναι το πλησιέστερο ως προς το συγκεκριμένο δείγμα. Στο δεύτερο βήμα, παίρνοντας τον μέσο όρο των δειγμάτων κάθε συστάδας, επανυπολογίζονται τα κεντροειδή της κάθε συστάδας, ώστε το κεντροειδές να είναι πιο αντιπροσωπευτικό στην πρόσφατα διαμορφωμένη συστάδα.

Ο αλγόριθμος εκτελεί επαναληπτικά αυτά τα δύο βήματα, μέχρις ότου τα κεντροειδή των συστάδων να μετατοπίζονται ελάχιστα και σε απόσταση μικρότερη από κάποια δοθείσα τιμή κατωφλίου (threshold). Ως εναλλακτικό κριτήριο τερματισμού του αλγορίθμου μπορεί να χρησιμοποιηθεί και ο αριθμός επαναλήψεων του αλγορίθμου.

3.3 Μοντέλα Επιβλεπόμενης Μάθησης

3.3.1 Naïve Bayes Classifier

Στην μηχανική μάθηση, οι ***naïve Bayes classifiers*** αποτελούν μια οικογένεια από απλούς πιθανοτικούς ταξινομητές που βασίζονται στην εφαρμογή του *θεωρήματος Bayes* (*από τον Thomas Bayes 1702-1761*) με αυστηρή (αφελή - *naïve*) υπόθεση ανεξαρτησίας ανάμεσα στα χαρακτηριστικά.

Ο αλγόριθμος Naïve Bayes έχει μελετηθεί εκτενώς στο παρελθόν σχεδόν από το 1950. Είχε εισαχθεί με ένα διαφορετικό όνομα αρχικά στην κοινότητα της ανάλυσης κειμένου στις αρχές της δεκαετίας του 1960, και παραμένει μια δημοφιλής βασική μέθοδος για τ ην κατηγοριοποίηση κειμένων, αν αυτά δηλαδή ανήκουν στην μια κατηγορία ή την άλλη (όπως spam/non spam, sports/politics, κτλ.), με κύρια χαρακτηριστικά τις συχνότητες εμφάνισης των λέξεων. Με κατάλληλη προ επεξεργασία των δεδομένων μπορεί να γίνει αρκετά ανταγωνιστικός ακόμα και με πιο ανεπτυγμένες μεθόδους στον τομέα αυτό συμπεριλαμβανομένου και του αλγορίθμου support vector machines.

Αποτελεί μια απλή τεχνική για την κατασκευή ταξινομητών, μοντέλων δηλαδή που ορίζουν ετικέτες κλάσης σε οντότητες προβλημάτων που αναπαρίστανται από διανύσματα με τιμές χαρακτηριστικών, όπου οι ετικέτες αυτές καθορίζονται από ένα πεπερασμένο σύνολο. Δεν είναι ένας μεμονωμένος αλγόριθμος για την εκπαίδευση τέτοιων ταξινομητών, αλλά μια οικογένεια αλγορίθμων που βασίζονται σε ένα κοινό πρότυπο.

Όλοι οι naïve Bayes ταξινομητές υποθέτουν ότι η τιμή ενός συγκεκριμένου χαρακτηριστικού είναι ανεξάρτητη από την τιμή οποιουδήποτε άλλου χαρακτηριστικού δεδομένης της μεταβλητής κλάσης. Για παράδειγμα, ένα φρούτο μπορεί να θεωρηθεί ότι είναι ένα μήλο αν είναι κόκκινο, στρογγυλό και έχει περίπου 10cm διάμετρο. Ένας naïve Bayes ταξινομητής θεωρεί πως καθένα από αυτά τα χαρακτηριστικά συνεισφέρει ανεξάρτητα στην πιθανότητα ότι αυτό το φρούτο είναι ένα μήλο, ανεξαρτήτως κάθε ενδεχόμενης συσχέτισης μεταξύ χρώματος, καμπυλότητας και χαρακτηριστικών διαμέτρου.

Γενικά, ο naïve Bayes είναι ένα μοντέλο δεσμευμένης πιθανότητας που ορίζεται ως εξής:

Δεδομένου ενός στιγμιότυπου προβλήματος προς κατηγοριοποίηση, αναπαριστάμενο από ένα διάνυσμα $\widehat{\boldsymbol{x}}$, το οποίο αντιπροσωπεύει κάποια $\boldsymbol{n}$ χαρακτηριστικά (ανεξάρτητες μεταβλητές), ορίζει για το αντικείμενο αυτό πιθανότητες $p(C_k|\widehat{\boldsymbol{x}})$, για κάθε πιθανή έκβαση ή κλάση K :

$$\widehat{\boldsymbol{x}} = (x_1, \dots, x_n)$$
$$p(C_k|\widehat{\boldsymbol{x}}) = p(C_k|x_1, \dots, x_n)$$

Το πρόβλημα με την παραπάνω διατύπωση είναι ότι αν ο αριθμός $\boldsymbol{n}$ των χαρακτηριστικών είναι μεγάλος ή αν ένα χαρακτηριστικό παίρνει ένα μεγάλο πλήθος από τιμές, τότε είναι αδύνατο να βασίσουμε ένα τέτοιο μοντέλο σε πίνακες πιθανοτήτων είναι αδύνατο. Συνεπώς, αναδιαμορφώνουμε το μοντέλο ώστε να το κάνουμε πιο ευέλικτο και ελεγχόμενο. Χρησιμοποιώντας το θεώρημα Bayes, η δεσμευμένη πιθανότητα μπορεί να αποσυντεθεί ως:

$$p(C_k|\widehat{\boldsymbol{x}}) = \frac{p(C_k)p(\widehat{\boldsymbol{x}}|C_k)}{p(\widehat{\boldsymbol{x}})}$$

Με πιο απλή αναπαράσταση και χρησιμοποιώντας καθιερωμένες έννοιες πιθανοτήτων και αντίστοιχη ορολογία, ο παραπάνω τύπος μπορεί να γραφτεί:

$$posterior = \frac{prior \times likelihood}{evidence}$$

Στην πράξη, το ενδιαφέρον βρίσκεται μόνο στον αριθμητή του κλάσματος, επειδή ο παρονομαστής δεν εξαρτάται από το διάνυσμα κλάσης $\overline{C}$, και οι τιμές των χαρακτηριστικών F_i είναι γνωστές, οπότε ο παρονομαστής είναι τελικά κάποια σταθερά. Ο αριθμητής είναι ισοδύναμος με το μοντέλο της από κοινού πιθανότητας (***joint probability***)

$$p(C_k, \widehat{\boldsymbol{x}}) = p(C_k\ , x_1, \dots, x_n)$$

η οποία μπορεί να γραφτεί ως εξής, χρησιμοποιώντας τον κανόνα της αλυσίδας για επαναλαμβανόμενες εφαρμογές του ορισμού της δεσμευένης πιθανότητας (***conditional probability***):

$$p(C_k, \hat{\boldsymbol{x}}) = p(C_k, x_1, \dots, x_n) = p(x_1, \dots, x_n, C_k) = \cdots$$
$$= p(x_1|x_2, \dots, x_n, C_k)\, p(x_2|x_3, \dots, x_n, C_k)$$
$$\dots p(x_{n-1}|x_n, C_k)\, p(x_n|C_k)\, p(C_k)$$

Τώρα αν συμπεριλάβουμε την "αφελή" υπόθεση σχετικά με την υπό όρους ανεξαρτησία, έστω ότι κάθε χαρακτηριστικό F_i είναι υπό όρους ανεξάρτητο από κάθε άλλο χαρακτηριστικό F_j για $j \neq i$, δεδομένης μια κατηγορίας C.Αυτό πρακτικά σημαίνει ότι:

$$p(x_i|x_{i+1}, \dots, x_n, C_k) = p(x_i|C_k) \;\; για \;\; i = \{1, \dots, n-1\}.$$

Ωστόσο η από κοινού πιθανότητα μπορεί να εκφραστεί ως:

$$p(C_k|x_1, \dots, x_n) \propto p(C_k, x_1, \dots, x_n) \propto p(C_k)\, p(x_1|C_k)\, p(x_2|C_k)\, p(x_3|C_k) \dots p(x_n|C_k)$$
$$\propto p(C_k) \prod_{i=1}^{n} p(x_i|C_k)$$

Αυτό σημαίνει ότι υπό τις παραπάνω προϋποθέσεις, η δεσμευμένη κατανομή πιθανότητας σχετικά με την μεταβλητή κατηγορίας C είναι:

$$p(C_k|x_1, \dots, x_n) = \frac{1}{const}\, p(C_k) \prod_{i=1}^{n} p(x_i|C_k)$$

όπου το στοιχείο $const = p(x)$ είναι ένας κλιμακωτός παράγοντας εξαρτώμενος μόνο από τα χαρακτηριστικά $x_1, \dots, x_n$, ο οποίος θα είναι μια σταθερά αν οι τιμές των μεταβλητών των χαρακτηριστικών είναι γνωστές.

Μέχρι τώρα αναφερθήκαμε στην ανεξαρτησία του μοντέλου των features, δηλαδή στο "αφελές" μοντέλο Bayes. Ο αντίστοιχος ταξινομητής που εφαρμόζει αυτό το μοντέλο, το συνδυάζει με ένα ***κανόνα απόφασης***. Ένας συνήθης κανόνας είναι να επιλέξουμε την υπόθεση που είναι πιο πιθανή, γνωστός και ως "***maximum a posteriori"*** ή ***MAP decision rule.*** Αυτός ο κανόνας ορίζει τον αφελή ταξινομητή και είναι η συνάρτηση που ορίζει την ετικέτα κλάσης $\hat{y} = C_k$ για κάποιο k ως:

$$\hat{y} = \arg \max_{k \in \{1, \dots, K\}} p(C_k) \prod_{i=1}^{n} p(x_i|C_k)$$

Η τεχνική που χρησιμοποιεί ο Naïve Bayes classifier βασίζεται καθώς προ είπαμε στο γνωστό θεώρημα και εφαρμόζεται ιδιαίτερα σε περιπτώσεις που η διάσταση των δεδομένων εισόδου του προβλήματος είναι αρκετά μεγάλη. Παρά την απλότητα του αλγορίθμου, πολλές φορές μπορεί να ξεπεράσει σε απόδοση και ακρίβεια ακόμα και πιο σύγχρονο και πολύπλοκα μοντέλα κατηγοριοποίησης.

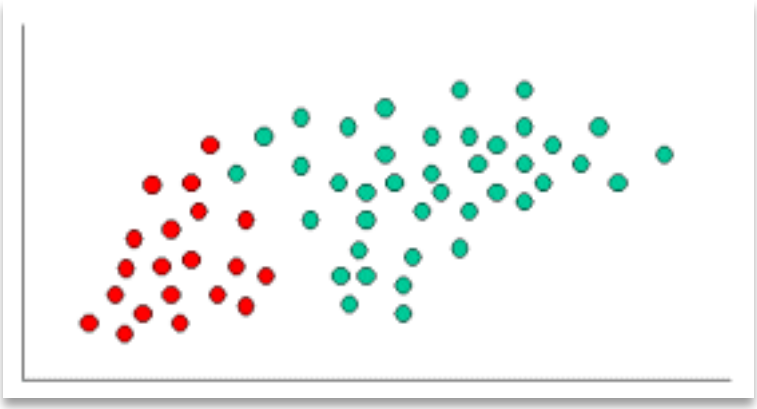

Figure 1. Δεδομένα 2 κατηγοριών (practical naive Bayes)

Για να παρουσιάσουμε τη φιλοσοφία του συγκεκριμένου αλγορίθμου, ας θεωρήσουμε το παράδειγμα που φαίνεται στην παραπάνω εικόνα. Όπως φαίνεται τα αντικείμενα μπορούν να ταξινομηθούν είτε ως *πράσινα**(green)*** ή *κόκκινα**(red)***. Αυτό που θέλουμε να πετύχουμε είναι να μπορούμε να διαχωρίζουμε καινούργια δείγματα όταν αυτά προκύπτουν, δηλαδή να αποφασίζουμε σε ποια από τις 2 αυτές κλάσεις ανήκουν βασιζόμενοι ωστόσο στα αντικείμενα που ήδη υπάρχουν και είναι κατηγοριοποιημένα.

Εφόσον υπάρχουν δύο φορές περισσότερα πράσινα αντικείμενα από κόκκινα, είναι λογικό να πιστεύουμε ότι ένα νέο ενδεχόμενο (το οποίο δεν έχει παρατηρηθεί ακόμα) είναι δύο φορές πιο πιθανό να ανήκει στην κατηγορία με τα πράσινα αντί σε αυτή με τα κόκκινα. Στην ανάλυση με Bayes, αυτή η πεποίθηση είναι γνωστή και ως αρχική πιθανότητα (***prior** probability*). Οι αρχικές πιθανότητες βασίζονται σε προηγούμενη εμπειρία, στην περίπτωση μας, στο ποσοστό των πράσινων και κόκκινων αντικειμένων, και συχνά χρησιμοποιούνται για να προβλέψουν αποτελέσματα πριν καν αυτά προκύψουν.

Συνεπώς μπορούμε να γράψουμε:

$$prior\ probability_{green} \propto \frac{number_{green}}{number_{total}}$$

$$prior\ probability_{red} \propto \frac{number_{red}}{number_{total}}$$

Έστω ότι υπάρχουν συνολικά 60 αντικείμενα, εκ των οποίων 40 είναι πράσινα και 20 είναι κόκκινα. Τότε οι αρχικές πιθανότητες για αυτές τις κλάσεις είναι:

$$\mathcal{P}(green) = \frac{40}{60} = 0.666\ ,\ \mathcal{P}(red) = \frac{20}{60} = 0.333$$

Στην συνέχεια, αφού έχουμε κατασκευάσει τις αρχικές πιθανότητες, είμαστε έτοιμοι να κατηγοριοποιήσουμε ένα νέο αντικείμενο **x** (λευκή μπάλα). Εφόσον τα αντικείμενα είναι πολύ καλά ομαδοποιημένα στο συγκεκριμένο παράδειγμα, μπορεί να θεωρήσει κανείς ότι όσα πιο πολλά πράσινα (ή κόκκινα) υπάρχουν στην περιοχή του **x**, τόσο πιο πιθανό (likely) είναι το νέο ενδεχόμενο να ανήκει στο συγκεκριμένο χρώμα.

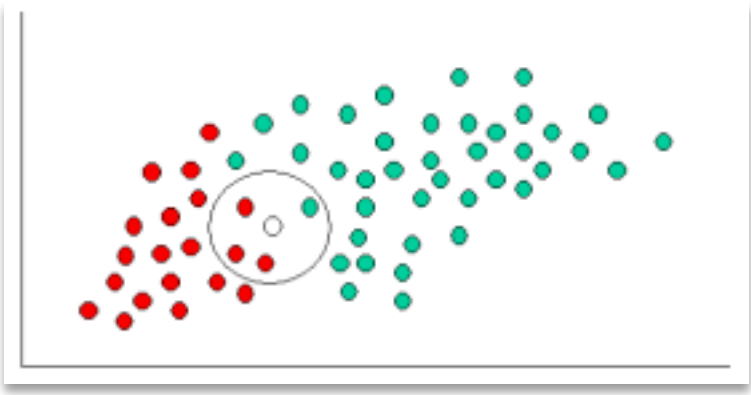

Figure 2. Κατηγοριοποίηση τυχαίου δεδομένου (practical naive Bayes)

Για να ορίσουμε αυτή την πιθανότητα (**likelihood**), σχεδιάζουμε ένα κύκλο γύρω από από το **x**, ο οποίος περιλαμβάνει ένα προεπιλεγμένο αριθμό (επιλεγμένο εκ των προτέρων – **a priori**) από σημεία με τις αντίστοιχες ετικέτες της κλάσης τους. Έπειτα υπολογίζουμε τον αριθμό των σημείων μέσα στον κύκλο που ανήκουν σε κάθε κλάση. Συνεπώς υπολογίζουμε τις πιθανότητες:

$$likelihood\ (x\ given\ green) \propto \frac{number_{green}\ (arround\ x)}{total_{green}}$$

$$likelihood\ (x\ given\ red) \propto \frac{number_{red}\ (arround\ x)}{total_{red}}$$

Από την απεικόνιση παραπάνω, είναι σαφές ότι η πιθανότητα του **x** δεδομένου το πράσινου είναι μικρότερη από την πιθανότητα του **x** δεδομένου το κόκκινο, αφού ο κύκλος περιέχει 1 green αντικείμενο και 3 red. Συνεπώς:

$$\mathcal{P}(x|green) = \frac{1}{40} = 0.025\ ,\ \mathcal{P}(x|red) = \frac{3}{20} = 0.15$$

Αν και η αρχική πιθανότητα δείχνει ότι το **x** θα έπρεπε ίσως να ανήκει στο πράσινο χρώμα καθώς υπάρχουν διπλάσια πράσινα από ότι κόκκινα αντικείμενα, η δεσμευμένη πιθανότητα καταλήγει σε άλλο συμπέρασμα, ότι δηλαδή η κλάση του **x** είναι το κόκκινο χρώμα καθώς βρίσκονται περισσότερα κόκκινα από πράσινα στην περιοχή του **x**.

Στην Bayesian ανάλυση, η τελική κατηγοριοποίηση παράγεται χρησιμοποιώντας και τις 2 πηγές πληροφορίας, δηλαδή και την αρχική και την δεσμευμένη πιθανότητα (prior × likelihood), για να κατασκευάσουμε μια τελική πιθανότητα (***posterior*** *probability*) χρησιμοποιώντας τον γνωστό κανόνα του Bayes:

$$\mathcal{P}(green|x) \propto \mathcal{P}(green) \times \mathcal{P}(x|green) = 0.666 \times 0.02 = 0.016$$
$$\mathcal{P}(red|x) \propto \mathcal{P}(red) \times \mathcal{P}(x|red) = 0.333 \times 0.15 = 0.04$$

Τελικά κατατάσσουμε το **x** στα κόκκινα αφού η μεγαλύτερη posterior πιθανότητα επιτυγχάνεται με ταυτότητα κλάσης το κόκκινο.

$$\mathcal{P}(red|x) > \mathcal{P}(green|x)$$

[**Σημείωση**: Οι παραπάνω πιθανότητες δεν είναι κανονικοποιημένες. Ωστόσο, αυτό δεν επηρεάζει το αποτέλεσμα της κατηγοριοποίησης καθώς οι σταθερές κατηγοριοποίησης τους είναι κοινές.]

3.3.2 Support Vector Machines algorithm (SVM)

Ο αλγόριθμος ***Support Vector Machines*** (**SVM**) είναι ένα σύνολο μεθόδων εκμάθησης που χρησιμοποιούνται για προβλήματα ταξινόμησης και παλινδρομικής ανάλυσης. Η κύρια ιδέα του SVM είναι να κατασκευαστεί ένα υπερεπίπεδο, έτσι ώστε η απόσταση διαχωρισμού μεταξύ των θετικών και αρνητικών παραδειγμάτων να μεγιστοποιείται. Τα διανύσματα των πιο κοντινών στοιχείων στο υπερεπίπεδο αυτό είναι τα υποστηρικτικά διανύσματα (support vectors).

Αυτή η επιθυμητή ιδιότητα επιτυγχάνεται ακολουθώντας την αρχή της ελαχιστοποίησης δομικού ρίσκου (*structural risk minimization*) από τη θεωρία της μηχανικής μάθησης. Η ιδέα της ελαχιστοποίησης του δομικού ρίσκου είναι να βρεθεί μια υπόθεση h για την οποία μπορούμε να εγγυηθούμε το χαμηλότερο πραγματικό σφάλμα. Το πραγματικό σφάλμα της h είναι η πιθανότητα της h να κάνει λάθος σε ένα τυχαία επιλεγμένο παράδειγμα το οποίο δεν έχει εξεταστεί στο παρελθόν. Το πλεονέκτημα της τεχνικής αυτής είναι ότι επιτυγχάνονται καλές επιδόσεις στα προβλήματα ταξινόμησης χωρίς να ενσωματώνεται γνώση από τον τομέα του προβλήματος.

Βλέποντας τα δεδομένα εισόδου σαν δύο σύνολα διανυσμάτων σε ένα ν-διάστατο χώρο, το SVM θα κατασκευάσει ένα διαχωριστικό υπερεπίπεδο σε αυτόν το χώρο, που θα μεγιστοποιεί την απόσταση μεταξύ των δύο συνόλων. Για τον υπολογισμό της απόστασης αυτής, κατασκευάζονται δύο παράλληλα υπερεπίπεδα, ένα σε κάθε πλευρά του διαχωριστικού υπερεπιπέδου, τα οποία "σπρώχνονται" πάνω στα δύο σύνολα δεδομένων. Διαισθητικά, ένας καλός διαχωρισμός επιτυγχάνεται από το υπερεπίπεδο που έχει τη μεγαλύτερη απόσταση από τα γειτονικά σημεία δεδομένων και των δύο συνόλων, δεδομένου ότι σε γενικές γραμμές όσο μεγαλύτερη είναι η απόσταση τόσο καλύτερο είναι το λάθος γενίκευσης του ταξινομητή.

Πιο αναλυτικά, καθώς ο svm είναι ένας αλγόριθμος large-margin και όχι πιθανοτικός, η βασική ιδέα πίσω από την διαδικασία εκπαίδευσης του αλγορίθμου είναι να βρεθεί ένα το υπερεπίπεδο μέγιστου περιθωρίου (maximum margin hyperplane), το οποίο αναπαρίσταται από το διάνυσμα $\underset{w}{\rightarrow}$ και το οποίο όχι μόνο διαχωρίζει τα διανύσματα των εγγράφων της μιας κλάσης από αυτά της άλλης αλλά ταυτάχρονα φροντίζει αυτός ο διαχωρισμός (ή το όριο), να είναι και όσο μεγαλύτερο γίνεται. Αυτό καταλήγει να είναι λοιπόν ένα φραγμένο πρόβλημα βελτιστοποίησης. Έστω ότι $c_j \in \{1, -1\}$, που αντιστοιχούν στα θετικά και αρνητικά

ενδεχόμενα, είναι οι σωστές κατηγορίες για τα κείμενα d_j, τότε η λύση μπορεί να προκύψει όπως φαίνεται στην παρακάτω εξίσωση:

$$\vec{w} = \sum_j a_j c_{j\,d\,j}, a_j \geq 0$$

όπου: $\vec{w}$ είναι ένα διάνυσμα, c_j είναι μια κλάση, και d_j είναι ένα έγγραφο κειμένου. Αυτά τα d_j για τα οποία τα a_j έιναι μεγαλύτερα από το μηδέν, ονομάζονται support vectors (υποστηρικτικά διανύσματα). Η κατηγοριοποίηση των ενδεχομένων ελέγχου γίνεται απλώς εξετάζοντας και καθορίζοντας σε ποιά πλευρά του $\vec{w}$ **hyperplane** αυτά βρίσκονται.

Η ταξινόμηση των δεδομένων είναι μια κοινή ανάγκη στο πεδίο της μηχανικής μάθησης. Ας υποθέσουμε ότι δίνονται κάποια σημεία δεδομένων που ανήκουν στα δύο σύνολα, και ο στόχος είναι να αποφασίσουμε σε ποιο σύνολο θα μπει ένα νέο στιγμιότυπο. Στην περίπτωση των SVM, ένα στιγμιότυπο θεωρείται σαν ένα διάνυσμα p-διαστάσεων, και θέλουμε να ξέρουμε αν μπορούμε να χωρίσουμε αυτά τα σημεία με ένα ***(p–1)-διάστατο υπερεπίπεδο*** που ονομάζεται γραμμικός ταξινομητής.

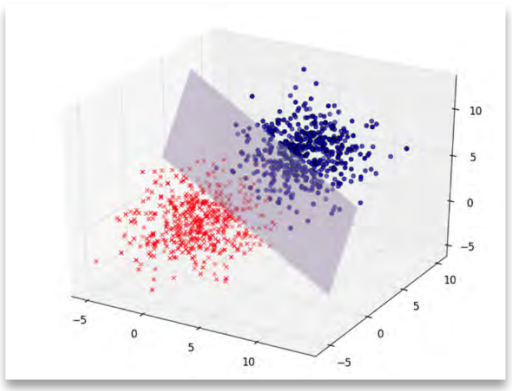

Figure 3. Διαχωρισμός συμβάντων με τον SVM στο χώρο (hyperplane)

Υπάρχουν πολλά υπερεπίπεδα που θα μπορούσαν να ταξινομήσουν τα δεδομένα. Ωστόσο, ενδιαφερόμαστε επιπλέον να διαπιστώσουμε εάν μπορούμε να πετύχουμε το μέγιστο διαχωρισμό (απόσταση) μεταξύ των δύο κλάσεων. Με αυτό εννοούμε ότι διαλέγουμε το υπερεπίπεδο, έτσι ώστε η απόσταση του υπερεπιπέδου από το πλησιέστερο σημείο δεδομένων να μεγιστοποιείται. Αυτό σημαίνει ότι η κοντινότερη απόσταση ανάμεσα σε ένα σημείο στο ένα διαχωρισμένο υπερεπίπεδο και σε ένα σημείο στο άλλο διαχωρισμένο υπερεπίπεδο

μεγιστοποιείται. Αν υπάρχει ένα τέτοιο υπερεπίπεδο, είναι γνωστό ως το υπερεπίπεδο μέγιστου διαχωρισμού, και ένας τέτοιος γραμμικός ταξινομητής είναι γνωστός ως ένας ταξινομητής μέγιστου διαχωρισμού.

Ο SVM ανήκει στην κατηγορία των γενικευμένων γραμμικών ταξινομητών. Μια ειδική ιδιότητά τους είναι ότι ταυτόχρονα ελαχιστοποιούν το εμπειρικό σφάλμα ταξινόμησης και μεγιστοποιούν τη γεωμετρική απόσταση. Ως εκ τούτου, είναι ταξινομητές μέγιστου διαχωρισμού. Μία αξιοσημείωτη ιδιότητα ενός SVM είναι ότι η ικανότητά του να μαθαίνει είναι ανεξάρτητη από τις διαστάσεις του χώρου χαρακτηριστικών. Ο SVM μετράει την πολυπλοκότητα των υποθέσεων με βάση την απόσταση που μπορεί να διαχωρίσουν τα στοιχεία, και όχι με βάση τον αριθμό των χαρακτηριστικών. Αυτό σημαίνει ότι μπορούμε να γενικεύσουμε ακόμη και με την παρουσία πάρα πολλών χαρακτηριστικών, αν τα στοιχεία μας μπορούν να διαχωριστούν με ένα ευρύ περιθώριο χρησιμοποιώντας συναρτήσεις από το χώρο υποθέσεων.

Ο Support Vector Machines έχει πολλά ελκυστικά χαρακτηριστικά. Είναι ένα σπάνιο παράδειγμα μεθοδολογίας όπου συνδυάζονται η γεωμετρική διαίσθηση, τα κομψά μαθηματικά, οι θεωρητικές εγγυήσεις και οι πρακτικοί αλγόριθμοι. Μπορεί να εφαρμοστεί αποτελεσματικά σε ένα ευρύ φάσμα προβλημάτων ταξινόμησης. Κλιμακώνεται σε τεράστια σύνολα δεδομένων και είναι ανεξάρτητος του τομέα του προβλήματος. Επιπλέον, μπορεί να αναπτυχθούν αποτελεσματικές συναρτήσεις πυρήνα για κάθε συγκεκριμένο πρόβλημα, προκειμένου να επιτευχθούν ακόμα καλύτερα αποτελέσματα. Ο SVM έχει πολλές επιτυχημένες εφαρμογές στον τομέα της βιοπληροφορικής (ταξινόμηση δεδομένων μικρό-συστοιχιών), της ανίχνευσης προσώπου και αναγνώρισης χειρογράφου κειμένου. Είναι επίσης πολύ καλός για την κατηγοριοποίηση κειμένων που στην προκειμένη περίπτωση είναι ο τομέας που μας ενδιαφέρει περισσότερο.

Επειδή τα μαθηματικά πίσω από το μοντέλο αυτό είναι αρκετά σύνθετα, ας δούμε ένα απλό παράδειγμα, έστω λοιπόν ότι έχουμε 2 σύνολα από μπάλες διαφορετικού χρώματος (μπλε και κόκκινες) πάνω στο τραπέζι και θέλουμε να τις χωρίσουμε με βάση το χρώμα. Παίρνουμε λοιπόν μια βέργα και την βάζουμε ανάμεσα τους πάνω στο τραπέζι. Αυτό φαίνεται να δουλεύει αρκετά καλά σε αυτήν την περίπτωση. Κάποιοι έρχονται τώρα και αφήνουν κάποιες επιπλέον μπάλες πάνω στο τραπέζι. Ο προηγούμενος διαχωρισμός μας συνεχίζει να δουλεύει σωστά όμως τώρα 1 μπάλα είναι στην λάθος μεριά του τραπεζιού, και συνεπώς υπάρχει μια καλύτερη θέση να τοποθετήσουμε την βέργα διαχωρισμού. Ο SVM διαρκώς προσπαθεί να τοποθετήσει

την βέργα στην καλύτερη δυνατή θέση με γνώμωνα να υπάρχει όσο το δυνατόν μεγαλύτερο κενό σε κάθε μια πλευρά της από τα αντίστοιχα σύνολα με μπάλες. Στο τέλος, η βέργα τοποθετείται σε ένα σημείο που διαχωρίζει απόλυτα τα σύνολα αυτά.

Ωστόσο ο SVM εφαρμόζει και ένα άλλο κόλπο επιπλέον. Έστω τώρα ότι έρχεται κάποιος και τοποθετεί τις μπάλες σε τέτοια σημεία ώστε να μην είναι δυνατό να διαχωριστούνε πλήρως με γραμμική βέργα πάνω στο επίπεδο. Ωστόσο τι κάνουμε σε αυτήν την περίπτωση? Αναποδογυρίζουμε το τραπέζι και πετάμε τις μπάλες στον αέρα. Έτσι, σε μια άλλη διάσταση και με εξαιρετικές τεχνικές, παίρνουμε ένα χαρτί και χωρίζουμε τα 2 σύνολα και πάλι πλήρως με μια επιφάνεια.

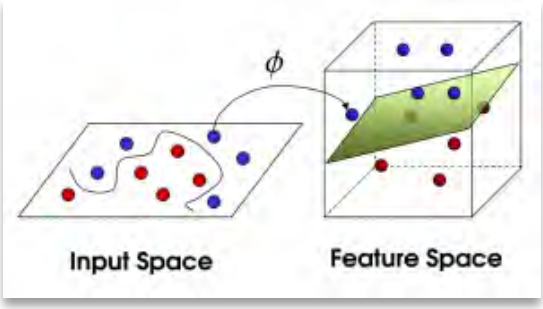

Figure 4. Πρακτική απεικόνιση του kernelling (SVM)

Αν βλέπαμε τις μπάλες προηγουμένως ο μόνος τρόπος να διαχωριστούν ήταν με μια καμπύλη γραμμή, ωστόσο στον αέρα αυτό μπορεί πολύ εύκολα να επιτευχθεί με μια ίσια επιφάνεια. Οι τυπικοί επιστήμονες απλά ονομάζουν τις μπάλες δεδομένα, την ράβδο classifier, τη μέγιστη δυνατή απόσταση trick *optimization*, το αναποδογύρισμα του τραπεζιού *kernelling* και το κομμάτι χαρτί *hyperplane*.

4 ΤΟ TWITTER

4.1 Εισαγωγή στο Twitter

Τα τελευταία χρόνια κοινωνικά δίκτυα όπως το Twitter γίνοτναι όλο και πιο δημοφιλή. Καθώς το Twitter είναι η πιο πολυσύχναστη ιστοσελίδα για *microblogging* με πάνω από 500 εκατομμύρια χρήστες (εκ των οποίων οι 332 εκατομμύρια είναι ενεργοί σύμφωνα με έγκυρα στατιστικά από τον Μάιο του 2015), 340 εκατομμύρια tweets την μέρα, αποτελεί μια ενδιαφέρουσα πηγή πληροφορίας, πράγμα που αποδεικνύεται και από τα 1,6 δις search queries ημερησίως.[12] Τα μηνύματα, ή σε όρους Twitter τα "*tweets*", είναι ένας τρόπος για να μοιράζεται ο κόσμος τα ενδιαφέροντα του δημόσια ή στα πλαίσια κάποιας συγκεκριμένης ομάδας. Το Twitter διαφοροποιείται από τα υπόλοιπα κοινωνικά δίκτυα περιορίζοντας αρκετά το μέγεθος του μηνύματος. Το μέγιστο μέγεθος των **140** χαρακτήρων περιορίζει τους χρήστες στο να εκφράζουν τις απόψεις τους σε μία ή δύο βασικές προτάσεις.

Επειδή το Twitter είναι ευρέως υιοθετημένο από όλους, μπορεί να φανεί σαν μια καλή αντανάκλαση του τι συμβαίνει σε όλο το κόσμο. Ανάμεσα σε όλα όσα συμβαίνουν, οι τελευταίες τάσεις παγκοσμίως έχουν το μεγαλύτερο ενδιαφέρον για τις περισσότερες εταιρίες. Αυτές οι τάσεις μπορούν να αναλυθούν δυναμικά και όταν αναγνωριστούν να υπάρξει η αντίστοιχη ανταπόκριση σε αυτές. Από την πλευρά του marketing, οι τάσεις αυτές μπορούν να χρησιμοποιηθούν ώστε να προκύψουν κατάλληλες ενέργειες όπως διαφημίσεις προϊόντων, λανσάρισμα νέων υπηρεσιών, κτλ. Η ανάλυση αντίστοιχων tweets μπορεί συνεπώς να είναι ένα χρυσορυχείο για εταιρίες και οργανισμούς με σκοπό να δημιουργήσουν ένα πλεονέκτημα έναντι των ανταγωνιστών τους.

4.2 Η δομή ενός tweet

Το tweet αποτελεί τον πυρήνα αυτής της πλατφόρμας, καθώς είναι ο μόνος τρόπος δημοσίευσης και μεταφοράς πληροφορίας ανάμεσα στους χρήστες, αν εξαιρέσουμε τα προσωπικά μηνύματα. Διακρίνεται από μια αρκετά ιδιαίτερη δομή πέραν του περιορισμού στο μέγεθος του. Καταρχήν, συνηθίζεται να περιέχει σχεδόν πάντα τα σύμβολα "**#**" και "**@**". Το #

ονομάζεται «*hashtag*» και με την χρήση του γίνεται μια αναφορά σε κάποιο θέμα (topic). Συνεπώς με αυτόν τον τρόπο προσφέρεται η δυνατότητα ομαδοποίησης πολλών δημοσίων tweet γύρω από ένα συγκεκριμένο θέμα και η αναζήτηση πληροφοριών με την χρήση πχ. #iphone6 που θα φέρει όλα τα σχετικά tweets για καινούριο iPhone. Με την εισαγωγή του @, γίνεται αναφορά σε κάποιον άλλο χρήστη (πχ. @tim_cook).

Ένα άλλο διαδεδομένο σύμβολο που εμφανίζεται συχνά στην αρχή των tweets, είναι το “RT”, το οποίο σημαίνει «retweet», δηλαδή την αναμετάδοση ενός tweet κάποιου χρήστη, από κάποιον άλλο χρήστη. Τέλος, αρκετά συχνή είναι και η χρήση url στα tweets, καθώς εκτός από την επιπρόσθετη αναφορά σε κάτι, δίνει και μια λύση στην περιορισμένη δυνατότητα ανάλυσης ενός θέματος σε τόσο λίγο χώρο.

Παρακάτω βλέπουμε την δομή από ένα sample tweet του "*TheEconomist*":

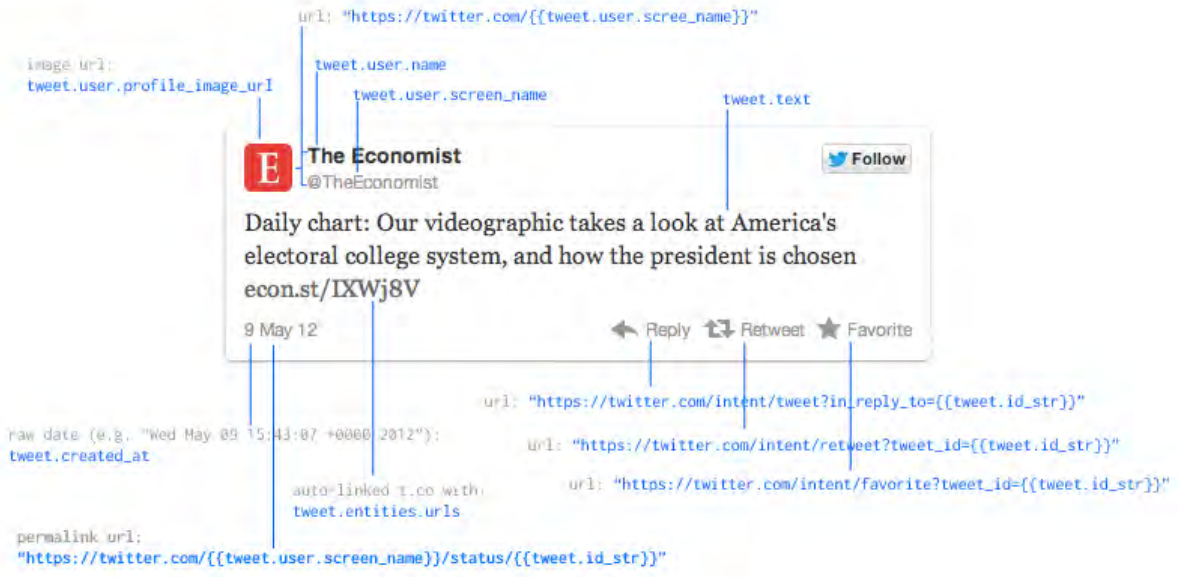

Figure 5. Η δομή ενός tweet (structure)

Το κομμάτι που έχει το μεγαλύτερο ενδιαφέρον για την ανάλυση μας είναι το “**tweet.text**”, το οποίο περιέχει το κείμενο που περιλαμβάνεται σε κάθε tweet.

4.3 Twitter Applications

4.3.1 Δημιουργία μιας Twitter-εφαρμογής

Το Twitter προσφέρει τη δυνατότητα δημιουργίας third-party εφαρμογών με σκοπό την διαρκή και πιο εύκολη εκμετάλλευση του τεράστιου όγκου πληροφορίας που διαθέτει. Συγκεκριμένα, μπορεί κάποιος με μια ασφαλή διαδικασία να πιστοποιήσει μια τέτοια εφαρμογή δίνοντας δικαιώματα πρόσβασης σε APIs από τον υπάρχοντα λογαριασμό του στο Twitter, χωρίς να απαιτείται η χρήση των αντίστοιχων credentials του χρήστη.

Συγκεκριμένα, μέσα από την πλατφόρμα του Twitter για developers, μπορεί κάποιος να επιλέξει τη δημιουργία μιας τέτοιας εφαρμογής, δίνοντας ένα μοναδικό όνομα, μια περιγραφή της εφαρμογής που πρόκειται να φτιάξει, ένα (placeholder) website-url για την πιθανή ιστοσελίδα της συγκεκριμένης εφαρμογής, και ένα callback-url ώστε να ξέρει το Twitter που να στείλει τις απαντήσεις για πιστοποίηση και τα ζητούμενα data από τις εκάστοτε κλήσεις στις διάφορες υπηρεσίες που παρέχει. Έπειτα ρυθμίζεις τα δικαιώματα που έχει αυτή η εφαρμογή στον λογαριασμό σου, και τέλος ολοκληρώνεις την διαδικασία με την ρύθμιση όλων των παραμέτρων σχετικά με την πιστοποίηση της μέσω web.

Για τα πλαίσια αυτής την μελέτης δημιούργησα μια εφαρμογή λοιπόν με το όνομα "***sent2eater***" με στόχο να έχω πρόσβαση από τον κώδικα μου ώστε να καταναλώσω όσα δεδομένα απαιτούνται, κάνοντας κλήσεις σε συγκεκριμένα web-services του Twitter. Το "sent-" προφανώς προέρχεται από το sentiment, καθώς το "-2eater" δηλώνει αφενός την πλατφόρμα twitter, αφετέρου την κατανάλωση των δεδομένων (to eat).

4.3.2 Πιστοποίηση με την χρήση του Twitter OAuth

Για την πιστοποίηση των εφαρμογών, το Twitter χρησιμοποιεί το OAuth, που παρέχει ένα τύπο πιστοποίησης εφαρμογής (application-only), κατά την οποία η εφαρμογή πραγματοποιεί api calls αυτόνομα χωρίς να περιλαμβάνει κάποιο χαρακτηριστικό του χρήστη.

Η διαδικασία είναι ως εξής, η εφαρμογή κωδικοποιεί και στέλνει στο Twitter το *consumer key* & *consumer secret* που έχει παραχθεί κατά την δημιουργία της ίδιας της εφαρμογής, στην συνέχεια στέλνει αυτή την πληροφορία στο OAuth με σκοπό να πάρει μια σκυτάλη πρόσβασης

(access token) και τέλος πραγματοποιεί μια **pin-based** χειραψία (handshake) με το Twitter χρησιμοποιώντας αυτά τα virtual credentials. Ο χρήστης τότε μεταβαίνει αυτόματα στην σελίδα του λογαριασμού του όπου του ζητείται αν θέλει να δεχτεί να δώσει την αντίστοιχη πιστοποίηση στην εφαρμογή που το ζητάει, και με την αποδοχή, προμηθεύεται με ένα pin το οποίο χρησιμοποιεί στην εφαρμογή και ολοκληρώνει την διαδικασία.

Οι κλήσεις που πραγματοποιούνται για την παραγωγή των tokens καθώς και όλα τα api calls στο Twitter απαραιτήτως χρησιμοποιούν SSL για να παρέχεται η απαιτούμενη ασφάλεια. Συνεπώς όλα τα αιτήματα για απόκτηση και χρήση δεδομένων ασφαλείας πρέπει να χρησιμοποιούν *https* endpoints.

4.3.3 Διεπαφές του Twitter – REST vs Streaming API

Συνεπώς χρειαζόμαστε ένα μέσο για να αναζητήσουμε τα επιθυμητά tweets και να δημιουργήσουμε τα αναγκαία datasets για την ανάλυση μας. Για τον σκοπό αυτό το Twitter παρέχει 2 ξεχωριστές διεπαφές, APIs (Application Programmable Interfaces), τα οποία μπορούμε να καλέσουμε δημιουργώντας έναν απλό web service client και καθορίζοντας συγκεκριμένα χαρακτηριστικά στο αίτημα που θα στείλουμε.

Έχουμε τις εξής λοιπόν διεπαφές: την *REST* (ή αλλιώς *search*) API, και την *Streaming* API.[14] Αυτές οι διεπαφές έχουν συγκεκριμένες ιδιότητες και αποσκοπούν στην εξυπηρέτηση συγκεκριμένων αναγκών ανά περίπτωση. Είναι δεδομένο πλέον ότι και οι δύο απαιτούν πιστοποίηση μέσω του OAuth, παρέχουν την δυνατότητα κάποιος να εισάγει κάποιο search-keyword (πχ. κάποιο #hashtag) ώστε να περιορίσει την αναζήτηση του, και ακόμα να εισάγει συγκεκριμένο χρήστη ή λογαριασμό, χρονική περίοδο και κάποια ίσως περιορισμένη γεωγραφική περιοχή.

Συνηθώς, τα δεδομένα επιστρέφουν σε **json** format, και περιέχουν όλη την πληροφορία που σχετίζεται με τα συγκεκριμένα tweets, όπως ποιος τα δημοσίευσε, από ποιο μέρος, πότε, ποιος και αν έγιναν retweeted, κ.α. Φυσικά στην δική μας περίπτωση, το κομμάτι του tweets που έχει το μεγαλύτερο ενδιαφέρον είναι το "**text**", που περιέχει σαφώς και την λέξη κλειδί με βάση την οποία έγινε η αναζήτηση. Η REST δίνει την δυνατότητα για πιο συγκεκριμένα search-queries, με περισσότερες επιλογές για φιλτράρισμα των αποτελεσμάτων και επιστροφή tweets από περισσότερους χρήστες. Η Streaming API, δημιουργεί μια σταθερή σύνδεση με το

twitter και ανοίγει μια ροή που φέρνει διαρκώς δεδομένα για όσο μένει ανοιχτή ή μέχρις ότου ικανοποιηθεί ένα επιθυμητό μέγιστο που μπορεί να οριστεί για την αναζήτηση των tweets.

Πέραν όμως αυτής της βασικής διαφοράς ανάμεσα στις 2 διεπαφές, υπάρχει σαφής διαφοροποίηση και όσο αφορά στα όρια αναζήτησης και την ποσότητα των tweets που μπορούν να αποκτηθούν από την κάθε μια.

Η **streaming** api συνήθως επιστρέφει πολύ μεγαλύτερο αριθμό από tweets. Είναι καταγεγραμμένο πως η ροή της φέρνει δεδομένα από το 1% του συνόλου όλων των tweets που υπάρχουν με μια συγκεκριμένη κατανομή. Έχει ένα μέγιστο περίπου στα **3.000 tweets / λεπτό**, αν δεν έχει αλλάξει κάτι μέχρι πρότινος. Αυτό το όριο μας επιτρέπει να αντλήσουμε προσεγγιστικά ένα maximum των 180.000 tweets την ώρα. Η **rest** api σε αντίθεση μπορεί να μας επιστρέψει μέχρι **100 tweets / αναζήτηση** και επιτρέπει περίπου 720 αιτήματα την ώρα, δίνοντας ένα μέγιστο των 72.000 tweets την ώρα.[12] Αυτό ωστόσο ισχύει ανά χρήστη. Δηλαδή, αν έχουμε μια εφαρμογή στην οποία ο κάθε χρήστης εισέρχεται με τον προσωπικό του λογαριασμό, τότε υπάρχει η δυνατότητα απορρόφησης 72k tweets την ώρα ανά χρήστη.

5 ΑΝΑΛΥΣΗ ΣΥΝΑΙΣΘΗΜΑΤΩΝ ΜΕ ΤΗΝ R

5.1 Πιστοποίηση εφαρμογής και συλλογή δεδομένων

Το πρώτο βήμα για να ξεκινήσουμε την πειραματική ανάλυση είναι να χρησιμοποιήσουμε την Twitter εφαρμογή που δημιουργήσαμε προηγουμένως ώστε να κατεβάσουμε ένα αναγκαίο σύνολο δεδομένων. Καταρχήν για να μπορέσουμε να συνδεθούμε σε αυτήν θα πρέπει να πιστοποιήσουμε την ύπαρξη μας μέσω των αντίστοιχων **consumer-key**, **consumer-secret** που προμηθευτήκαμε (σε *base64 string* αναπαράσταση), και στην συνέχεια με τη χρήση μιας έτοιμης βιβλιοθήκης (ROAuth) να συνάψουμε μια χειραψία (***handshake***) με το Twitter ώστε να αποκτήσουμε πρόσβαση στα ελεύθερα δεδομένα. (*Αυτά τα δεδομένα πρόσβασης ωστόσο πρέπει πάντα να κρατώνται κρυφά διότι αποτελούν κλειδί για άμεση είσοδο στον λογαριασμό του Twitter με τον οποίο αποκτήσαμε πρόσβαση στην εφαρμογή.*)

```
install.packages("ROAuth")
install.packages("RCurl")
install.packages("base")
library(ROAuth)
library(RCurl)
library(base)

# Twitter Application - credentials
consumer_key    <- '**********' #to be hidden
consumer_secret <- '**********'

requestURL  <- "https://api.twitter.com/oauth/request_token"
accessURL   <- "https://api.twitter.com/oauth/access_token"
authURL     <- "https://api.twitter.com/oauth/authorize"

# create oauth
my_oauth <- OAuthFactory$new(consumerKey = consumer_key,
                             consumerSecret = consumer_secret,
                             requestURL = requestURL, accessURL = accessURL,
                             authURL = authURL)
# handshake
my_oauth$handshake(cainfo =
            system.file("CurlSSL", "cacert.pem", package = "RCurl"))
```

Αφού δημιουργήσουμε το αντικείμενο **my_oauth** και πραγματοποιήσουμε το απαιτούμενο handshake, η εφαρμογή θα μας ανακατευθύνει στον προσωπικό μας λογαριασμό

του Twitter ζητώντας άδεια για πιστοποίηση της συγκεκριμένης εφαρμογής. Αν αποδεχτούμε αυτό το αίτημα, θα εμφανιστεί ένας *ψήφιος κωδικός στον browser τον οποίο θα πρέπει να εισάγουμε στην κονσόλα της R για να ολοκληρωθεί επιτυχώς η χειραψία. Βλέπουμε ωστόσο στα *requestURL*, *accessURL* και *authURL* ότι χρησιμοποιείται **SSL** (“*https://…*”).

Εάν θέλουμε αρκετά δεδομένα για την ανάλυση μας, ο πιο εύκολος τρόπος να τα αποκτήσουμε είναι χρησιμοποιώντας την ***streaming API*** του Twitter καθώς μας επιτρέπει να αντλήσουμε ένα αρκετά μεγάλο σύνολο σε σύντομο χρονικό διάστημα και με ευρεία κατανομή για ποικίλες εφαρμογές. Συνεπώς είναι αρκετά καλό μέσο για την προσομοίωση ενός γενικού μοντέλου, εύκολα προσαρμόσιμο σε πολλά προβλήματα. Τυπικά τα μέγιστα που προσφέρει αυτή η διεπαφή είναι αρκετά μεγαλύτερα από το πραγματικά (για τους αντίστοιχους χρόνους), με συνέπεια να χρειάζεται να ορίσουμε ένα αρκετά μεγάλο διάστημα για το timeout της κλήσης μας ώστε να είμαστε σίγουροι πως τα επιστρεφόμενα δεδομένα θα αντιστοιχούν στον ζητούμενο αριθμό.

Η R παρέχει μια βιβλιοθήκη, την ***streamR***, μέσω την οποίας μπορούμε να καλέσουμε την παραπάνω διεπαφή. Η μέθοδος ***filterStream()***, λειτουργεί σαν ένας client μέσω του οποίου μπορούμε να επιλέξουμε την συλλογή συγκεκριμένου αριθμού δεδομένων με κάποια επιθυμητά χαρακτηριστικά. Σαν παραμέτρους πρέπει να περάσουμε για αρχή το my_oauth, ένα όνομα αρχείου στο οποίο θέλουμε να αποθηκευτούν τα δεδομένα που ζητάμε, την γλώσσα που θέλουμε να είναι τα tweets, ένα συγκεκριμένο **keyword** με βάση το οποίο γίνεται η

```
install.packages("streamR")
library(streamR)

# download positive Tweets
filterStream( file.name = "largePos.json", language = "en", track = ":)",
                              timeout = 10000, tweets = 20000,
                              oauth = my_oauth, verbose = TRUE)

# download negative Tweets
filterStream( file.name = "largeNeg.json", language = "en", track = ":(",
                             timeout = 10000, tweets = 20000,
                             oauth = my_oauth, verbose = TRUE)
```

αναζήτηση, ένα σύνολο tweets που θέλουμε να λάβουμε και ένα τυπικό χρόνο που θα διαρκέσει η κλήση μέχρι να λάβει τον απαιτούμενο αριθμό δεδομένων.

Στην συγκεκριμένη περίπτωση, μας ενδιαφέρει να κάνουμε 2 κλήσεις για να συλλέξουμε το σύνολο των δεδομένων που χρειαζόμαστε. Με βάση τους περιορισμούς που επιβάλει στους χρόνους εκτέλεσης το μηχάνημα που θα χρησιμοποιηθεί, θεωρώ πως ένα σύνολο από ***40.000 tweets*** είναι αρκετό για να μπορέσουμε να παραμετροποιήσουμε όλα τα μοντέλα ανάλυσης και ταυτόχρονα να έχουμε αρκετά δεδομένα για την εκπαίδευση του αλγορίθμου.

Συνεπώς, κατεβάζουμε *20.000* tweets με keyword "☺" για να πάρουμε δεδομένα με θετικό περιεχόμενο, και *20.000* tweets με keyword "☹" για να πάρουμε αρνητικά δεδομένα. Τα δεδομένα λαμβάνονται σε *json* format και τα αποθηκεύουμε σε 2 αρχεία με όνομα ***largePos.json*** και ***largeNeg.json*** αντίστοιχα.

5.2 Προεπεξεργασία δεδομένων

5.2.1 Μέθοδος καθαρισμού δεδομένων

Στην συνέχεια πρέπει να φτιάξουμε μια συνάρτηση γενικού σκοπού για όλα τα tweets, η οποία να εφαρμόζεται επαναληπτικά σαν φίλτρο στο περιεχόμενο τους. Δηλαδή, θα πρέπει να παίρνει σαν παράμετρο το κείμενο (**$text**) ενός tweet, εφαρμόζοντας κυρίως μεθόδους αντικατάστασης που βασίζονται σε κοινές εκφράσεις (*regular expressions* ή ***reg exp***), και να επιστρέφει το κείμενο καθαρό από αριθμούς, σημεία στίξης, ειδικούς χαρακτήρες, κτλ.

```
## {textCleaner.R} function to clean Tweets content
clean.text <- function(text) {
  require(base)
  # Remove all kind of common links in Tweets
  text <- gsub("(http|https)\\:\\/\\/(\\w+|(\\.)|\\/)*(\\w+)", "", text)
  # Remove RT/via etc.
  text <- gsub("(RT|via)((?:\\b\\W*@\\w+)+)", "", text)
  # Remove mentioned users tagged with @
  text <- gsub("@\\w+", "", text)
  # Remove hashtags starting with #
  text <- gsub("#\\w+", "", text)
  # Remove ^&amp and &amp
  text <- gsub("(\\^?&amp)", "", text)
  # Remove punctuation (!@#$%^&*()_+-={}|[]\;':",./<>?)
  # and all kind of combinations { + emoticons :) :| :( :/ }
  text <- gsub('[[:punct:]]', " ", text)
  # Remove 1 or more (all) digits
  text <- gsub('\\d+', "", text)
  # Remove >= 2 space characters and replace with single space
  text <- gsub("[ \t]{2,}", " ", text)
  # Convert all characters to Lower case
  text <- tolower(text)
  ## Return clean text!
  return(text)
}
```

Πιο συγκεκριμένα, καθώς φαίνεται και στην παραπάνω αναφορά στον κώδικα που υλοποιεί αυτή την μέθοδο καθαρισμού, την ***textCleaner.R***, ο κύριος στόχος είναι η μετατροπή των tweets σύνθετης δομής κάθε κατηγορίας, με σκοπό να έρθουν σε μια μορφή ικανή ώστε να δομηθεί ένα κοινού τύπου μοντέλο που θα αναπαραστήσει τα δεδομένα για το πείραμα μας.

Απώτερος σκοπός είναι το αποτέλεσμα της μεθόδου να περιέχει όσο το δυνατόν πιο ενιαίου τύπου δεδομένα, περιέχοντας όσο το δυνατόν περισσότερη αξιοποιήσιμη πληροφορία σχετικά με το συναίσθημα που περιλαμβάνεται στο tweet.

Η μέθοδος που καλείται επί το πλείστο για το καθάρισμα του περιεχομένου των tweets είναι η **gsub()** από την βιβλιοθήκη *base* η οποία ορίζεται ως εξής:

```
gsub(pattern, replacement, x, ignore.case = FALSE, perl = FALSE,
     fixed = FALSE, useBytes = FALSE)
```

Αυτή η μέθοδος στην ουσία ψάχνει να ταιριάξει το όρισμα *"**pattern**"* μέσα σε όλα τα στοιχεία ενός διανύσματος χαρακτήρων, και να τα αντικαταστήσει με το αντίστοιχο "***replacement***". Το *pattern* είναι ένα διάνυσμα χαρακτήρων (character vector) το οποίο περιέχει ένα regular expression (*regexp*) ή κάποιους συγκεκριμένους χαρακτήρες (*string*). Το *replacement* είναι αντίστοιχα ένα character vector το οποίο θα αντικαταστήσει όλα τα πιθανά matches. Αυτή η μετατροπή θα εφαρμοστεί στο κείμενο (character vector) που παρέχουμε στο όρισμα "***x***"**.** (Σύμφωνα με την υλοποίηση μας τα υπόλοιπα ορίσματα παραμένουν στις default τιμές τους.)

Το πρώτο πράγμα που θέλουμε να αφαιρέσουμε από τα tweets μας είναι τα διάφορα *link / url* που μπορεί να περιέχονται σε αυτά. Έχουν σύνθετη δομή και βασικά δεν περιέχουν καθόλου χρήσιμη πληροφορία σχετικά με το συναίσθημα του κειμένου που μας ενδιαφέρει. Σε πιο ειδικές αναλύσεις ίσως θα μπορούσαμε να τα χρησιμοποιήσουμε για να περιορίσουμε τα δεδομένα μας κοντά σε κάποιο συγκεκριμένο θέμα (topic), ωστόσο και πάλι δεν θα πρέπει να είναι μέρος του πυρήνα της ανάλυσης σε καμία περίπτωση.

Έπειτα πρέπει να αφαιρέσουμε τα *retweets* **RT/via** και όποια λέξη ενδεχομένως είναι ακριβώς μετά από αυτά. Όπως επίσης και τα "**@username**" καθώς στην συγκεκριμένη ανάλυση το όνομα του χρήστη που δημιούργησε το tweet έχει μηδενική επιρροή στο συναίσθημα του. Ωστόσο, το ίδιο πρέπει να κάνουμε και για τα **#hashtags,** καθώς η αναφορά σε κάποιο συγκεκριμένο topic, ένα topic σύμφωνα με το οποίο τις πιο πολλές φορές σε πραγματικές αναλύσεις θα χρησιμοποιούσαμε για να συλλέξουμε τα δεδομένα, δεν συνεισφέρει καθόλου στην πρόβλεψη και εξαγωγή του συνολικού συναισθήματος του tweet.

Στην συνέχεια κοινή πρακτική είναι να αφαιρούμε όλα τα σημεία στίξης καθώς δεν πρέπει να χρησιμοποιηθούν στον αλγόριθμο μας, και όλες τις μορφές **punctuation** που μπορεί

να υπάρχουν στο κείμενο. Έτσι, θα αφαιρεθούν στην ουσία και όλα τα emoticons που υπάρχουν στα tweets, και στην περίπτωση μας σίγουρα υπάρχουν σε κάθε ένα από αυτά καθώς έτσι έγινε η αναζήτηση μας. Τα emoticons σίγουρα φανερώνουν άμεσα το συναίσθημα του περιεχομένου, εκτός και αν έχουμε να κάνουμε με φαινόμενα "*ειρωνείας*", ωστόσο στόχος είναι η πρόβλεψη μας να βασίζεται καθαρά σε γλωσσολογικές ιδιαιτερότητες και κανόνες του εναπομείναντος κειμένου. Στα ίδια πλαίσια, αφαιρούμε αριθμούς (***numbers***) και κενά (***spaces***) μεγαλύτερα από 2 θέσεις ώστε τα διανύσματα κειμένου που προκύπτουν από τα επεξεργασμένα tweets να είναι απολύτως ωμά (*raw text*).

Τέλος είναι αναγκαίο να μετατρέψουμε το περιεχόμενο όλων των δεδομένων μας σε πεζά γράμματα (***lower text***) ώστε να υπάρχει μια ενιαία και κοινή δομή για την κατασκευή των features. Ακριβώς η ίδια επεξεργασία γίνεται τόσο στα δεδομένα που χρησιμοποιούνται για την εκπαίδευση του αλγορίθμου που κάνει την κατηγοριοποίηση, όσο και στα δεδομένα που χρησιμοποιούμε για να ελέγξουμε την ορθή λειτουργία και την ακρίβεια της μηχανής πρόβλεψης που υλοποιούμε.

5.2.2 Επεξεργασία δεδομένων και ανάθεση ετικετών

Σημαντικό βήμα στην όλη διαδικασία μας είναι να αυτοματοποιήσουμε την προ-επεξεργασία των δεδομένων με μια συνάρτηση που θα φορτώνει τα δεδομένα μας, θα τα καθαρίζει από την περιττή πληροφορία και θα εφαρμόζει οποιαδήποτε αναγκαία μορφοποίηση. Η ακόλουθη συνάρτηση ***tweetParser.R*** αναλαμβάνει την υλοποίηση όλων αυτών των διαδικασιών που περιγράψαμε:

```
# {tweetParser.R} function used to load tweets from a json file,
# clean the text content and assign a class label to each of them
parse.tweets <- function(filename, label, ch.threshold) {
  require(streamR)
  require(rlist)
  require(stringr)
  #read positive tweets from json file (e.g. "tweets.json")
  tweetsDataFrame <- readTweets(filename)
  #keep only text from tweet list
  textOnly <- unlist(lapply(tweetsDataFrame, '[[', 'text'))
  #create a matrix with the text of the tweets
  tweetsTextMatrix <- as.matrix(as.character(textOnly))
  #remove odd charactes and create a character matrix
  tweetsTextMatrix <- iconv(tweetsTextMatrix, from = "ASCII",
            to = "UTF-8",  sub = "", mark = TRUE, toRaw = FALSE)
  #check matrix dimension
  numberOfTweets <- dim(tweetsTextMatrix)[1]
  #create a single column matrix with the given
  #class label  ('positive'/'negative')
  labelVector <- matrix(label, nrow = numberOfTweets , ncol = 1)
  #combine matrices columns
  datasetMatrix <- cbind(tweetsTextMatrix, labelVector)
  #copy to cleanLabeledDataset to keep both dirty/clean datasets
  cleanLabeledDataset <-  datasetMatrix
  clean.text <- source("textCleaner.R")
  #loop through each row of the data matrix to clean text
  for (i in 1:numberOfTweets) {
    temporalText <- cleanLabeledDataset[i ,1]
    temporalText <- clean.text(temporalText)
    cleanLabeledDataset[i ,1] <- temporalText
  }
  #keep Tweets of > $(ch.threshold) chars long
  cleanLabeledDataset <- subset(m <- cleanLabeledDataset,
            str_length(m[,1]) > ch.threshold)
  return(cleanLabeledDataset)
}
```

Σύμφωνα με τον ορισμό της παραπάνω συνάρτησης, σαν είσοδο παίρνει 3 παραμέτρους. Καταρχήν, πρέπει να δώσουμε το όνομα του αρχείου (***filename***) στο οποίο έχουμε αποθηκευμένα τα tweets (δεδομένα προς επεξεργασία). Το αρχείο αυτό θα πρέπει να είναι της μορφής filename = "*dataset.json*", και να περιλαμβάνει ένα σύνολο από tweets. Με την συνάρτηση *readTweets()* από το package *streamR* που χρησιμοποιήσαμε και στην αρχή για την συλλογή των tweets, περνάμε ολόκληρο το περιεχόμενο των δεδομένων σε ένα data frame, το οποίο είναι στην ουσία αντικείμενο σαν 2-διάστατος πίνακας με κάποια παραπάνω χαρακτηριστικά.

Στην συνέχεια, δεδομένου το αντικειμένου που δημιουργήσαμε από τα tweets του αρχείου, πρέπει να εφαρμόσουμε δύο μεθόδους από το package *base*, την ***lapply()*** ώστε να μετατρέψουμε τα δεδομένα σε μια λίστα ίδιου μεγέθους με διανύσματα, όπου κάθε διάνυσμα περιέχει το κείμενο μόνο (**$text**) από κάθε tweet, και μετά να περάσουμε την λίστα στην ***unlist()*** ώστε να παράγουμε ένα μοναδικό διάνυσμα, τα στοιχεία του οποίου θα είναι τα αντίστοιχα κείμενα των tweets. Αφού ολοκληρωθεί αυτή η διαδικασία, είναι σημαντικό να μετατρέψουμε το διάνυσμα αυτό σε διάνυσμα χαρακτήρων (character vector), και στην συνέχεια να το μετατρέψουμε σε αντικείμενο πίνακα (*matrix*). Αυτές οι μετατροπές είναι εξαιρετικής σημασίας καθώς πρέπει να επεξεργαστούμε τα δεδομένα με μεθόδους που έχουν συγκεκριμένα ορίσματα, και επιπλέον δεν πρέπει να χάσουμε καθόλου πληροφορία από τα αρχικά αντικείμενα μέχρι να διαμορφώσουμε τις τελικές δομές που θα χρησιμοποιήσουμε για την εκπαίδευση και τον έλεγχο των μοντέλων κατηγοριοποίησης που εξετάζουμε.

Ένα πρόβλημα που πρέπει να χειριστούμε είναι αυτό με τους "παράξενους" χαρακτήρες που εμφανίζονται πολλές φορές στα tweets. Αυτοί οφείλονται συνήθως σε *emojis* ή άλλα σπάνια σύμβολα που εισάγουν οι χρήστες κατά την δημιουργία των tweets. Πρέπει λοιπόν να μετατρέψουμε το περιεχόμενο του κειμένου από ***ASCII*** σε ***UTF-8***. Στην ουσία αυτό που κάνουμε είναι να αναζητούμε τέτοιους χαρακτήρες σε κάθε tweets, και αν βρούμε να τους αφαιρούμε. Μετατρέποντας λοιπόν το κείμενο στην επιθυμητή κωδικοποίηση, αν υπάρχουν χαρακτήρες που δεν μπορούν να μετατραπούν, αφαιρούνται και εισάγεται ένα κενό στην θέση τους, καθώς και γενικότερα αν υπάρχουν tweets που για κάποιο λόγο δεν μπορούν να μετατραπούν στην επιθυμητή μορφή, αφαιρούνται ολόκληρα από το σύνολο. Κάτι τέτοιο έχει συνήθως σαν αποτέλεσμα την μείωση του συνολικού αριθμού των δεδομένων που χρησιμοποιούμε, αλλά παρατηρήθηκε ότι δεν είναι σε καμία περίπτωση πάνω από *2%* του συνολικού αριθμού tweets.

Αφού κάνουμε μια πρώτη επεξεργασία στο σύνολο των tweets, βρίσκουμε την τελική διάσταση του ολικού πίνακα με τα δεδομένα, και ο αριθμός των γραμμών είναι σαφώς ο τελικός αριθμός των tweets που έχουν απομείνει. Δημιουργούμε έναν πίνακα στήλη με το δοσμένο όρισμα label, και το συγχωνεύουμε σε ένα κοινό πίνακα με τα tweets. Είναι η κατάλληλη στιγμή να καλέσουμε την *cleanText.R*, και να την εφαρμόσουμε σε κάθε tweet χωριστά. Έτσι λοιπόν θα έχουμε πλήρως αξιόπιστα δεδομένα, και στην κατάλληλη κοινή επιθυμητή μορφή.

Ένα επιπλέον χαρακτηριστικό που έχω εισάγει στην συνάρτηση, είναι η δυνατότητα να φιλτράρουμε το σύνολο των tweets με βάση τους συνολικούς χαρακτήρες που έχουν απομείνει μετά από όλες τις διάφορες επεξεργασίες που έχουν υποστεί τα δεδομένα μας. Είναι σχετικά γνωστό ότι μεγαλύτερη ποσότητα κειμένου συνήθως περιγράφει καλύτερα το συναίσθημα που περιέχεται σε αυτό, καθώς και είναι μπορεί να περιέχει περισσότερη χρήσιμη πληροφορία για την εκπαίδευση του αλγορίθμου ώστε να εφαρμόζει καλά σε νέα άγνωστα δεδομένα. Συνεπώς, θα αναλύσουμε και περιπτώσεις με διαφορετικά μεγέθη στο περιεχόμενο των tweets. Φυσικά, όσο αυξάνεται αυτός ο αριθμός περιορισμού (*character threshold*), τόσο το σύνολο μικραίνει, με αποτέλεσμα να μένουν όλο και λιγότερα tweets για εκπαίδευση(*train data*) δεδομένου ότι κρατάμε ένα σταθερό αριθμό για τα δεδομένα ελέγχου(*test data*).

Ας δούμε ένα παράδειγμα της εισόδου σε αυτή την συνάρτηση και της πιθανής εξόδου, για ένα κομμάτι από το θετικό σύνολο δεδομένων που χρησιμοποιούμε. Τα δεδομένα στο αρχείο .json είναι της παρακάτω μορφής (πιο συγκεκριμένα το κομμάτι εκείνο που αφορά στο κείμενο του tweet):

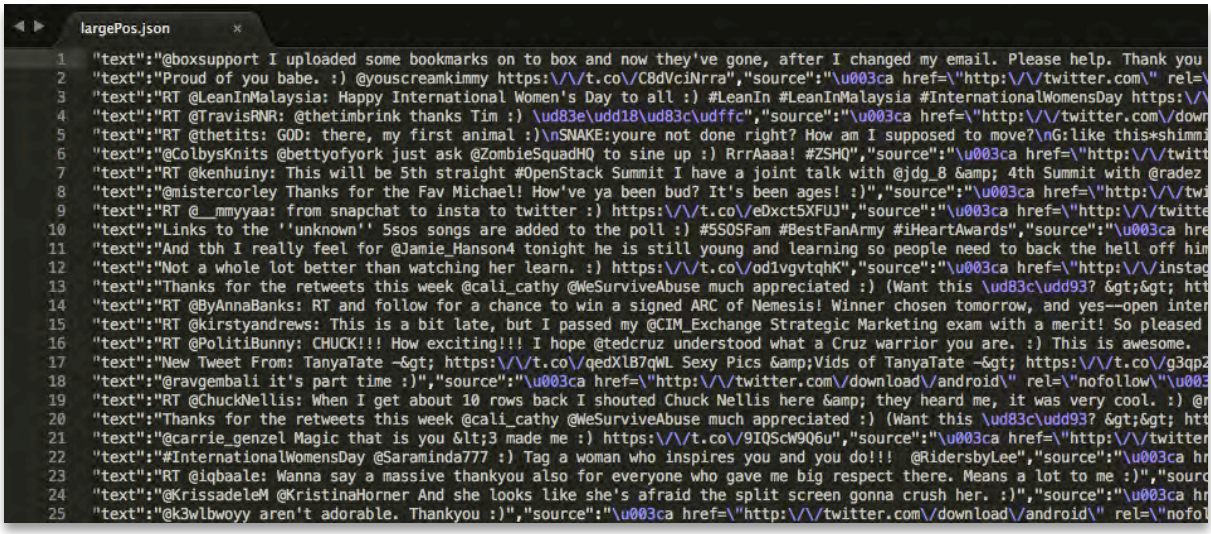

Figure 6. Πολλαπλά raw tweets σε json αρχείο

Μια πιθανή έξοδος μετά από την επεξεργασία αυτών των δεδομένων είναι ένα data frame με 2 στήλες, όπου στην 1η βρίσκονται τα καθαρισμένα tweets και στην 2η το class-label, που στην προκειμένη περίπτωση είναι "*positive*" καθώς αφορά στο θετικό σύνολο. Ένα κομμάτι αυτού του αντικειμένου φαίνεται παρακάτω:

	V1	V2
1	i uploaded some bookmarks on to box and now they ...	positive
2	happy international women s day to all	positive
3	god there my first animal snake youre not done right ...	positive
4	just ask to sine up rrraaaa	positive
5	this will be th straight summit i have a joint talk with ...	positive
6	thanks for the fav michael how ve ya been bud it s be...	positive
7	from snapchat to insta to twitter	positive
8	links to the unknown sos songs are added to the poll	positive
9	and tbh i really feel for tonight he is still young and le...	positive
10	not a whole lot better than watching her learn	positive
11	thanks for the retweets this week much appreciated w...	positive
12	rt and follow for a chance to win a signed arc of neme...	positive
13	this is a bit late but i passed my strategic marketing e...	positive
14	chuck how exciting i hope understood what a cruz wa...	positive
15	new tweet from tanyatate gt sexy pics vids of tanyatat...	positive
16	when i get about rows back i shouted chuck nellis her...	positive
17	thanks for the retweets this week much appreciated w...	positive
18	magic that is you It made me	positive

Figure 7. Ταξινομημένα καθαρά tweets έπειτα από επεξεργασία

5.2.3 Δημιουργία συνόλου δεδομένων προς ανάλυση

Αρχικά θα πρέπει να φορτώσουμε την συνάρτηση που δημιουργήσαμε για να καθαρίσουμε και να φορτώσουμε τα tweets με τα αντίστοιχα class labels τους. Στην συνέχεια αναθέτουμε σε 2 μεταβλητές όπως φαίνεται στην παρακάτω αναφορά, τα θετικά και αρνητικά σύνολα δεδομένων χρησιμοποιώντας σαν κατώφλι τους ***20 χαρακτήρες***. Έτσι, μετά από όλη την επεξεργασία δεν θα υπάρχει κανένα tweet με πάνω από τόσους χαρακτήρες.

```
source("tweetParser.R")

## Load Train Set from JSON files
positiveDataSet <- parse.tweets("largePos.json", "positive", 20)
dim(positiveDataSet)[1] # actual number of positive parsed tweets

negativeDataSet <- parse.tweets("largeNeg.json", "negative", 20)
dim(negativeDataSet)[1] # actual number of negative parsed tweets

datasetDimension <- dim(positiveDataSet)[1] + dim(negativeDataSet)[1]
datasetDimension
```

Αναθέτουμε στην μεταβλητή *datasetDimension* τον συνολικό αριθμό της κάθετης διάστασης του αθροίσματος των δεδομένων μας. Αυτό σημαίνει πως χρησιμοποιώντας σαν σημείο αναφοράς ένα σταθερό αριθμό για το σύνολο των δεδομένων ελέγχου (**test set**), μπορούμε να ορίσουμε δυναμικά, συναρτήσει αυτής της διάστασης, το επιθυμητό σύνολο εκπαίδευσης (**train set**).

```
> positiveDataSet <- parse.tweets("largePos.json", "positive", 20)
12459 tweets have been parsed.
> dim(positiveDataSet)[1]
[1] 10162
> negativeDataSet <- parse.tweets("largeNeg.json", "negative", 20)
14880 tweets have been parsed.
> dim(negativeDataSet)[1]
[1] 11592
> datasetDimension <- dim(positiveDataSet)[1] + dim(negativeDataSet)[1]
> datasetDimension
[1] 21754
>
```

Figure 8. Φόρτωση και επεξεργασία των tweets με την tweetsParser μέθοδο

Όπως βλέπουμε στο στιγμιότυπο [Figure 8] της εκτέλεσης του παραπάνω κώδικα, από το αρχείο με τα θετικά tweets, έγιναν parse τα *12.459* από τα *20.000*, το πιθανότερο είναι τα υπόλοιπα είτε να μην ήταν πλήρη ήταν να είχαν προβλήματα στα δεδομένα τους, και στην συνέχεια απέμειναν μόνο τα *10.162* καθώς τα υπόλοιπα φιλτραρίστηκαν μέσα από τις μεθόδους επεργασίας. Για τα αρνητικά tweets, από τα *14.880* (*parsed*) έμειναν τελικά τα *11.592*. Έτσι συνολικά μας απομένουν ***21.754*** ανάμεικτα tweets ώστε να χρησιμοποιήσουμε για το πείραμα μας. Από αυτά, θα χρησιμοποιήσουμε ένα σταθερό αριθμό ίσο με ***1.000*** για τα δεδομένα ελέγχου (test-set) της απόδοσης των μοντέλων, και τα υπόλοιπα (~ *20k*) , με διάφορους συνδιασμούς, για την εκπαίδευση τους.

Τα δεδομένα αυτά φυσικά, δεν είναι τόσα πολλά ώστε να θεωρηθεί ότι το πρόβλημα μας ανάγεται στην τάξη των big-data προβλημάτων, συνεπώς και η ανάλυση που θα ακολουθήσουμε θα είναι τυπική. Περισσότερα data θα μπορούσαν να συλλεχθούν για ακόμα καλύτερη απόδοση των αλγορίθμων, όμως μετά θα ήταν αρκετά χρονοβόρα διαδικασία η εκπαίδευση των μοντέλων, καθώς και η λειτουργία πρόβλεψης. Αυτό συμβαίνει γιατί η μηχανή παραγωγής του μοντέλου με τα features (document term matrix) θα ανέβαινε πολύ σε διάσταση και μάλιστα με πάρα πολλούς sparse terms (αραιά μήτρα, πολλά μηδενικά στοιχεία).

*(Με βάση τις δυνατότητες του μηχανήματος που χρησιμοποιήθηκε για την ανάλυση των δεδομένων, συγκεκριμένα ένας mac με επεξεργαστή Intel Core i5 (1.4 GHz) και μνήμη 4 GB (1600 MHz DDR3), είναι αναγκαίο να επιλεχθεί ένα data set τέτοιο ώστε να είναι εύκολη η προσομοίωση πολλών δομικών ώστε σύμφωνα και με την θεωρία να μπορούν να προσεγγιστούν όσο το δυνατόν καλύτερα οι απαιτούμενες παράμετροι του προβλήματος.)

5.3 Κατηγοριοποίηση με τον αλγόριθμο Naïve Bayes

Η πρώτη ανάλυση θα γίνει με την βοήθεια του naïve Bayes μοντέλου (*package* ***e1071***). Θα εκπαιδεύσουμε ένα τέτοιο αλγόριθμο με το σύνολο των δεδομένων που δημιουργήσαμε στο προηγούμενο βήμα, και θα ελέγξουμε τα αποτελέσματα και την απόδοση του με βάση ένα συγκεκριμένο test-set ίσο με 1000 tweets, όπως άλλωστε φαίνεται στην αναφορά που υπάρχει παρακάτω.

Αφού φορτώσουμε όλα τα δεδομένα που διαθέτουμε στο **nbTotalDataSet**, πρέπει να φροντίσουμε να ανακατέψουμε με τυχαίο τρόπο (sample) τα tweets, ώστε όταν διαχωρίσουμε το *train-set* με το *test-set*, να υπάρχουν εξίσου θετικά και αρνητικά tweets και στα 2 σύνολα δεδομένων. (Αρχικά είναι μοιρασμένα όπως τα βάλαμε κατά την επεξεργασία τους, δηλαδή πρώτα όλα τα θετικά και μετά όλα τα αρνητικά.)

```
nbTestSetSize <- 1000
nbTotalDataSetSize <- datasetDimension

nbTotalDataSet <- rbind2(positiveDataSet[, ], negativeDataSet[, ])

tempNbTrainDf1 <- as.data.frame(nbTotalDataSet[1:nbTotalDataSetSize,])
## sample data
tempNbTrainDf2 <- tempNbTrainDf1[sample(nrow(tempNbTrainDf1)),]
nbTotalDataSet <- as.matrix(tempNbTrainDf2)

nbTrainDocumentTermMatrix <- create_matrix(
        nbTotalDataSet[nbTestSetSize:nbTotalDataSetSize ,1],
        language = 'english',
        removeStopwords = TRUE,
        removeNumbers = TRUE,
        stemWords = TRUE,
        weighting = tm::weightTfIdf,
        removeSparseTerms = 0.998)

nbTrainDTMcharMatrix <- apply(as.matrix(
        nbTrainDocumentTermMatrix), 2, as.character)
```

Στην συνέχεια, πρέπει να δημιουργήσουμε το feature object που θα δώσουμε στο μοντέλο για εκπαίδευση. Αυτό είναι ένα *document term matrix* (**dtm**) από το *package* ***tm***, ένας δισδιάστατος πίνακας δηλαδή όπου οι γραμμές D_m είναι τα tweets (documents) και οι στήλες είναι οι λέξεις (terms) t_n (στην περίπτωση μας απλές λέξεις, unigrams), και σε κάθε κελί a_{mn}

που αντιστοιχεί σε ένα σετ ***(D_m , t_n)*** υπολογίζεται η συχνότητα εμφάνισης της λέξης μέσα στο κάθε έγγραφο. Αυτή είναι η κλασσική προσέγγιση, ωστόσο εμείς εφαρμόζουμε μια πιο περίτεχνη μέθοδο, και δημιουργούμε τον πίνακα αυτό με την τεχνική της ***tf-idf*** που αναλύσαμε σε προηγουμένως.

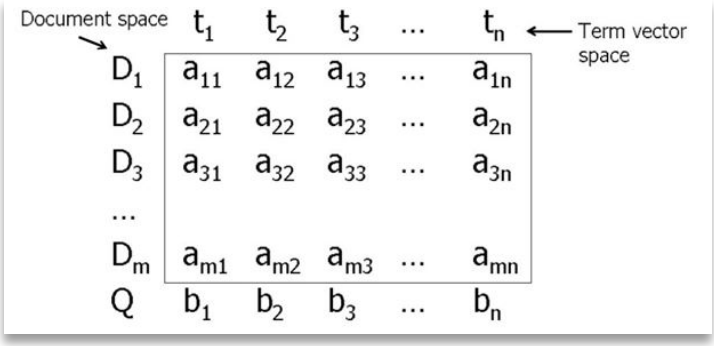

Figure 9. Απεικόνιση ενός Document Term Matrix (dtm)

Σημαντικό είναι να επιλέξουμε φίλτρα όπως η γλώσσα να είναι μόνο “english”, αν και αντίστοιχο περιορισμό έχουμε εισάγει και όταν κατεβάζουμε τα δεδομένα μας, καθώς έχουμε επίσης προβλέψει την αφαίρεση stop words και αριθμών. Χρησιμοποιούμε επιπλέον μια παράμετρο για την αφαίρεση κάπου όρων ώστε να κάνουμε τον πίνακα λιγότερο αραιό. Με το *removeSparseTerms = 0.998*, με πιθανά ορίσματα από το 0 έως το 1, με το 1 να είναι ανεκτικό σε απόλυτο sparsity, καταφέρνουμε να μειώσουμε λίγο την διάσταση του αντικειμένου, και να το κατεβάσουμε από 100% sparsity σε **99%**, που όμως επιφέρει σημαντική βελτίωση στην απόδοση του μοντέλου.

```
> nbTrainDocumentTermMatrix
<<DocumentTermMatrix (documents: 20755, terms: 498)>>
Non-/sparse entries: 85447/10250543
Sparsity           : 99%
Maximal term length: 20
Weighting          : term frequency - inverse document frequency (normalized) (tf-idf)
>
```

Figure 10. DTM του trainSet για τον naive Bayes

Όπως βλέπουμε στο στιγμιότυπο [Figure 10] εκτέλεσης ακριβώς από πάνω, τα συνολικά documents είναι 20755, δηλαδή 1000 λιγότερα από το συνολικό data set. Παρόλο που τα συνολικά non-sparse terms αποτελούν μόλις το ~1% του συνολικού dtm, είναι αρκετά για να μας παρέχουν μια καλής τάξης ακρίβεια όπως θα δούμε παρακάτω.

Την αντίστοιχη διαδικασία ακριβώς πρέπει να ακολουθήσουμε και για το test-set, δηλαδή τα δεδομένα που θα ελέγξουν τον αλγόριθμο μας, τα οποία αποτελούνται από ακριβώς **1000** tweets. Τα features στο σύνολο του train-set πρέπει να είναι ίδιου τύπου με αυτά στο test-set, ώστε να μπορέσουν να γίνουν οι κατάλληλοι υπολογισμοί, συνεπώς πρέπει να δημιουργήσουμε ένα document term matrix με 1000 documents (tweets).

Είναι αναγκαίο τα 2 αυτά σύνολα δεδομένων να είναι ξεχωριστά, και να μην υπάρχει επικάλυψη καθώς υπάρχει περίπτωση χρησιμοποιώντας κοινά δεδομένα, να έχουν τέλεια προσαρμογή του αλγορίθμου στα δεδομένα ελέγχου, και να βλέπουμε αρκετά μεγάλες αποδόσεις, που όμως δεν είναι τίποτα περισσότερο από εικονικές/ιδανικές, καθώς σε ένα καινούριο σετ από δεδομένα, μπορεί να έχει πολύ μικρότερη απόδοση και ακολούθως καθίσταται πλήρως αναξιόπιστος.

Παρακάτω απεικονίζεται ο κώδικας που δημιουργεί το test-dtm για τα 1000 πρώτα tweets που χρησιμοποιούνται για έλεγχο του naïve Bayes model:

```
tempNbTestDf1 <- as.data.frame(nbTotalDataSet[1:nbTestSetSize,])
## sample data
tempNbTestDf2 <- tempNbTestDf1[sample(nrow(tempNbTestDf1)),]
nbTestSet <- as.matrix(tempNbTestDf2)

nbTestDocumentTermMatrix <- create_matrix(nbTestSet[ ,1],
        language = 'english', removeStopwords = TRUE,
        removeNumbers = TRUE, stemWords = TRUE,
        weighting = tm::weightTfIdf,
        removeSparseTerms = 0.998)

nbTestDTMcharMatrix <- apply(as.matrix(
        nbTestDocumentTermMatrix), 2, as.character)
```

```
> nbTestDocumentTermMatrix
<<DocumentTermMatrix (documents: 1000, terms: 470)>>
Non-/sparse entries: 4152/465848
Sparsity           : 99%
Maximal term length: 20
Weighting          : term frequency - inverse document frequency (normalized) (tf-idf)
>
```

Figure 11. DTM του testSet για τον naive Bayes

Είναι σημαντικό επίσης να αναφερθεί εδώ ότι κατά την δημιουργία του dtm περνάμε μόνο την πρώτη στήλη από το data-frame, αυτή δηλαδή που περιέχει το κείμενο των tweets,

και όχι τα class labels, καθώς αυτά θα χρησιμοποιούν αυτοτελή κατά την εκπαίδευση του αλγορίθμου και μάλιστα με αντιστοιχία 1-1 και με απόλυτη ισότητα μήκους διανυσμάτων.

Έπειτα από την προαναφερθείσα διαδικασία, ήρθε η ώρα για την εκπαίδευση του αλγορίθμου naïve Bayes με το train-set που κατασκευάσαμε.

```
# naive bayes model
naiveBayes.classifier <- naiveBayes(nbTrainDTMcharMatrix[,] ,
        as.factor( nbTotalDataSet[nbTestSetSize:nbTotalDataSetSize, 2]),
        laplace = 1)

# predict
nbPredicted.results <- predict(naiveBayes.classifier,
                               nbTestDTMcharMatrix[,])
```

Το πρώτο όρισμα που περνάμε στην συνάρτηση κατασκευής του μοντέλου είναι όλα τα στοιχεία του τελικού *train-dtm,* το οποίο έχει μετατραπεί σε απλό matrix το οποίο περιέχει τις τιμές των κελιών του dtm για κάθε όρο ανά tweet. Το δεύτερο όρισμα, είναι τα class-labels που αντιστοιχούν σε αυτά τα tweets, τα οποία τα παίρνουμε από την $2^{η}$ στήλη του αρχικού train set που δημιουργήθηκε κατά την διαδικασία καθαρισμού των δεδομένων. Τέλος το τρίτο όρισμα "*laplace=1*" είναι μια τεχνική που χρησιμοποιείται ευρέως στο μοντέλο naïve Bayes, και ονομάζεται laplace smoothing.

$$p(C_k|x_1, \dots, x_n) \cong p(C_k) \prod_{i=1}^{n} p(x_i|C_k)$$

$$p(x_i|C_k) = \frac{1 + \sum_{d_j \in C_k} tfidf(x_i, d_j)}{|V| + \sum_{d_j \in C_k} \sum_{t=1}^{|V|} tfidf(x_t, d_j)}$$

Αν έχουμε πολλούς μηδενικούς όρους στο dtm, τότε το γινόμενο των πιθανοτήτων για αυτούς τους όρους δεδομένης κάποιας κλάσης θα υπολογιζόταν εξίσου μηδέν, με αποτέλεσμα να χάναμε αρκετή από την απόδοση μας στην πρόβλεψη. Έτσι, προσθέτωντας +1 σε όλους τους όρους, απαλοίφουμε τους μηδενικούς και αυξάνουμε τα βάρη για τους υπόλοιπους μη-μηδενικούς όρους.

Τέλος, υπάρχει μια μέθοδος που αναλμβάνει να κάνει την πρόβλεψη με βάση ένα δεδομένο test-set, σύμφωνα με το μοντέλο που περνάμε σε αυτή. Η *predict()* function λοιπόν, μας δίνει σαν αποτέλεμσα μια λίστα με τιμές positive/negative, τα factor levels των class labels δηλαδή, σε πλήρη αντιστοιχεία με τα tweets του test-set. Αυτές οι τιμές, αποτελούν τις ετικέτες που ανέθεσε ο αλγόριθμος μας στα tweets ελέγχου κατά την διαδικασία πρόβλεψης, και στόχος μας είναι επομένως να δούμε κατά πόσο αυτές οι τιμές ανταποκρίνονται στα πραγματικά labels που έχουν ανατεθεί αρχικά στα δεδομένα αυτά κατά την φάση της επεξεργασίας, καθώς μην ξεχνάμε ότι και αυτά τα data έχουν συλλεχθεί με γνώμωνα να περιέχουν είτε ☺ ή ☹ ως κλειδιά αναζήτησης.

Στην συνέχεια, θα επεξεργαστούμε αυτά τα αποτελέσματα (**nbPredicted.results**) , και θα προσπαθήσουμε να υπολογίσουμε μια συνολική απόδοση για αυτό αλλά και για το επόμενο μοντέλο που θα αναλύσουμε (svm), ώστε να βγάλουμε κάποια χρήσιμα συμπεράσματα.

5.4 Κατηγοριοποίηση με το μοντέλο SVM

Παρόλο που σε αρκετές μελέτες έχει αποδειχθεί ότι ο naïve Bayes έχει καλά αποτελέσματα με αξιόπιστη απόδοση για κατηγοριοποίηση κειμένου, σε αρκετές άλλες υποστηρίζεται το αντίθετο, και φαίνεται να τον ξεπερνά ο SVM και μάλιστα σε σημαντικό βαθμό. Ένα σημαντικό κριτήριο για την επιλογή του μοντέλου εξ αρχής, είναι το πόσες κλάσεις έχει το classification πρόβλημα στο οποίο εργαζόμαστε. Για παράδειγμα, στο *binary* πρόβλημα της κατηγοριοποίησης κειμένου σε απλώς positive/negative, και οι 2 αλγόριθμοι μπορεί να θεωρηθεί ότι είναι αρκετά αποδοτικοί. Ωστόσο, σε πολυδιάστατα προβλήματα, και στην περίπτωση που στην πράξη δεν ισχύει απόλυτα η συνθήκη ανεξαρτησίας των δεδομένων που έχουμε υποθέσει για τον naïve Bayes, τότε ο SVM επιφέρει καλύτερα αποτελέσματα.

Ας δούμε λοιπόν πως θα χειριστούμε την ανάλυση του ίδιου προβλήματος με το μοντέλο support vector machines, ακολουθώντας παρόμοια βήματα, και κοινές προϋποθέσεις όσο αφορά στα δεδομένα εκπαίδευσης αλλά και ελέγχου. Στο τέλος θα μπορέσουμε να αναλύσουμε τα αποτελέσματα και από τους 2 αλγόριθμους και να συγκρίνουμε την απόδοση τους.

```
svmDataSet <- rbind2(positiveDataSet, negativeDataSet)

## sample data
tempSvmDataFrame1 <- as.data.frame(svmDataSet)
tempSvmDataFrame2 <- tempSvmDataFrame1[sample(nrow(tempSvmDataFrame1)),]
svmDataSet <- as.matrix(tempSvmDataFrame2)

# DTM - tfIdf (train)
svmDocumentTermMatrix <- create_matrix(svmDataSet[, 1],
                 language = "english", removeStopwords = TRUE,
                       removeNumbers = TRUE, stemWords = TRUE,
                        weighting = tm::weightTfIdf,
                        removeSparseTerms = 0.998)
```

Έτσι φορτώνουμε όλα τα δεδομένα από τα θετικά και αρνητικά (clean) tweets, και φτιάχνουμε πάλι ένα document term matrix (με όλα τα data). Εδώ θα χρησιμοποιήσουμε ένα άλλο package, που λέγεται *RTextTools*, και θα ακολουθήσουμε μια διαφορετική προσεγγίσει ως προς τον σχηματισμό των αντικειμένων train/test set. Στην ουσία, με αυτή την βιβλιοθήκη, έχουμε την δυνατότητα να κατασκευάσουμε ένα ***container*** με όλο το σύνολο των δεδομένων,

και να καθορίσουμε σε αυτό το σημείο τον ακριβή αριθμό των tweets που θα χρησιμοποιηθούν ως train data αλλά και ως test data. Έτσι, η ανάλυση γίνεται λίγο πιο εύκολη, καθώς με τον container αυτό, μπορούμε να παραμετροποιήσουμε τα χαρακτηριστικά του προβλήματος πολύ ευκολότερα και δυναμικά από ότι στην προηγούμενη περίπτωση.

```
svmTestSize <- 1000
svmVariableTotalDataSize <- 1.00*datasetDimension  #100% of data-set

## Create a CONTAINER (specifing train/test data)
svmContainer <- create_container(svmDocumentTermMatrix,
                    as.numeric(as.factor(svmDataSet[, 2])),
                    trainSize =svmTestSize:svmVariableTotalDataSize,
                    testSize = 1:svmTestSize,
                    virgin = FALSE)
```

Εδώ έχουμε ορίσει σαν *test-set* 1000 tweets, και μια μεταβλητή που να καθορίζει δυναμικά το σύνολο των δεδομένων με βάση ένα ποσοστό του αρχικού *datasetDimension*. Συνεπώς μπορούμε να διαμορφώσουμε εύκολα τον αριθμό των tweets που παίρνουν μέρος στο *train-set*. Αρχικά σε αυτή την μεταβλητή έχει τεθεί ολόκληρο το σύνολο των δεδομένων.

Στον container επίσης θα πρέπει να καθορίσουμε και τα class labels, εισάγοντας σαν όρισμα την $2^{η}$ στήλη του svm data-set που είναι οι ετικέτες που έχουν ανατεθεί κατά την φάση επεξεργασίας στο κάθε tweets. Συνεπώς, αυτό το αντικείμενο περιέχει όλη την πληροφορία που απαιτείται για την εκπαίδευση, την πρόβλεψη αλλά και την αξιολόγηση του μοντέλου που κατασκευάζουμε.

Το μόνο που μένει τώρα είναι να δημιουργήσουμε ένα μοντέλο svm, εκπαιδεύοντας το με τα δεδομένα που έχουμε ορίσει σαν train-set στον container. Δίνοντας λοιπόν σαν ορίσματα απλά τον container και καθορίζοντας τον αλγόριθμο να είναι "SVM", ολοκληρώνουμε την διαδικασία.

```
## Train a Support Vector Machine (SVM) model
## (default) kernel = "radial"
svmModel = train_models(svmContainer, algorithms = "SVM")

## Get the classification results
svmResults = classify_models(svmContainer, svmModel)
```

Η μέθοδος **train_models()** (*RTextTools*), μπορεί να πάρει ένα διάνυσμα από πιθανούς αλγορίθμους όπως οι {"BAGGING", "BOOSTING", "GLMNET", "MAXENT", "NNET", "RF", "SLDA" και "TREE" }, ωστόσο εμείς θα περιοριστούμε στην σύγκριση των 2 μοντέλων που αναλύσαμε προηγουμένως. Δυστυχώς, δεν υπάρχει δυνατότητα για χρήση του NB με αυτό το εργαλείο, κάτι το οποίο θα διευκόλυνε αρκετά την ανάλυση μας.

Με την κλήση της ***classify_models(container, model)*** μπορούμε τελικά να δοκιμάσουμε να προβλέψουμε την κλάση σε όλα τα tweets που έχουμε καθορίσει να αποτελούν το test-set στον αρχικό container. Έτσι, παίρνουμε μια αντίστοιχη λίστα με αποτελέσματα κατηγοριοποίησης, δηλαδή ετικέτες κλάσης, την ***svmResult,*** την οποία θα χρησιμοποιήσουμε παρακάτω για να αναλύσουμε τα αποτελέσματα του συγκεκριμένου μοντέλου αλλά και να το συγκρίνουμε με τον naïve Bayes.

6 ΑΠΟΤΕΛΕΣΜΑΤΑ & ΣΥΜΠΕΡΑΣΜΑΤΑ

6.1 Αποτελέσματα και σύγκριση μοντέλων

6.1.1 Αποτελέσματα και ακρίβεια του naïve Bayes

Εφόσον εκτελέσαμε το παραπάνω πείραμα (κοινά δεδομένα και προϋποθέσεις) με τις δύο προαναφερθείσες προσεγγίσεις, πρέπει να αναλύσουμε τα αποτελέσματα και να δούμε αν μπορούμε να καταλήξουμε σε κάποια αξιόλογα συμπεράσματα.

Ξεκινώντας από τον naïve Bayes, ας δούμε ποια είναι τα αποτελέσματα κατηγοριοποίησης για το σύνολο των 1000 test tweets.

```
table(nbTestSet[ ,2], nbPredicted.results)

nb.accuracy <- recall_accuracy(nbTestSet[,2], nbPredicted.results)
nb.accuracy
```

Η 1η εντολή τυπώνει ένα πίνακα 2x2 (γνωστό και ως *accuracy-table ή confusion-matrix*) ο οποίος περιλαμβάνει τα αποτελέσματα της κατηγοριοποίησης σε συνάρτηση με τα πραγματικά αποτελέσματα που έπρεπε να προβλέψει, καθώς αυτά υπάρχουν στο πρώτο όρισμα αυτής.

```
> table(nbTestSet[ ,2], nbPredicted.results)
          nbPredicted.results
           negative positive
  negative      442       83
  positive      262      213
>
```

Figure 12. Αποτελέσματα κατηγοριοποίησης με τον naiveBayes

Εδώ βλέπουμε για αρχή ότι στα 1000 tweets συνολικά, αυτός ο αλγόριθμος προέβλεψε ότι τα 704 {*442+262*} είναι αρνητικά (*negative*) και τα 296 {*83+213*} είναι θετικά (*positive*). Οι κατηγορίες κάθετα μας δείχνουν την πρόβλεψη, ενώ οι οριζόντιες μας δείχνουν την πραγματική κλάση των tweets.

Συνεπώς, μια κοινή προσέγγιση για την αξιολόγηση των αποτελεσμάτων είναι να μετρήσουμε τα true/false positive (tp / fp) και τα true/false negative (tn / fn). Αυτό σημαίνει

ότι από τα 704 που παρατηρήθηκαν ως αρνητικά, στην πραγματικότητα ήταν μόνο τα 442, και από τα 296 που παρατηρήθηκαν θετικά, ήταν μόνο τα 213. Άρα έχουμε:

RESULTS	predicted	
actual	**tn** = 442	**fp** = 83
	fn = 262	**tp** = 213

Η 2η εντολή θα μας τυπώσει την ακρίβεια του αλγορίθμου με βάση αυτά τα αποτελέσματα και παρόλο που υπάρχουν και άλλα μέτρα για την απόδοση τέτοιου είδους μοντέλων, αυτό είναι το πιο καθοριστικό.

```
> nb.accuracy <- recall_accuracy(nbTestSet[,2], nbPredicted.results)
> nb.accuracy
[1] 0.655
>
```

Figure 13. Ακρίβεια κατηγοριοποίησης με τον naiveBayes

Βλέπουμε λοιπόν, ότι μια τυχαία εκτέλεση του πειράματος με τον naïve Bayes, μας επιστρέφει αποτελέσματα με ακρίβεια **0.655** (ή **65.5%)**, ένα νούμερο που είναι γενικά αποδεκτό από τέτοια μοντέλα, ωστόσο επιφέρει προσπάθεια για βελτίωση. Με τον όρο τυχαία εκτέλεση περιγράφουμε το γεγονός ότι το πείραμα εισάγει μια τυχαιότητα στα δεδομένα καθώς όταν δημιουργεί τα dataset, πραγματοποιεί ένα sampling στα δεδομένα. Συνεπώς για να μπορέσουμε να αξιολογήσουμε πλήρως τον αλγόριθμο αυτό αλλά και να τον συγκρίνουμε με άλλες προσεγγίσεις όπως τον svm, θα πρέπει να κάνουμε ένα *cross-validation*, δηλαδή να τρέξουμε το πείραμα κάποιες (n) φορές, και να υπολογίσουμε μια μέση ακρίβεια (***mean accuracy***), την οποία θα χρησιμοποιήσουμε για την τελική σύγκριση.

```
> nb.accuracyVector <- c(0.677, 0.673, 0.687, 0.655, 0.692)
> mean(nb.accuracyVector)
[1] 0.6768
>
```

Figure 14. Μέση ακρίβεια κατηγοριοποίησης με τον naiveBayes

Επιλέγοντας λοιπόν να τρέξουμε το πείραμα **5** συνεχόμενες φορές, δημιούργησα ένα διάνυσμα ακρίβειας ώστε να υπολογίσουμε την τελική μέση ακρίβεια του naïve Bayes classification model, η οποία προκύπτει ίση με **0.6768** (67.68% ~ **68%**).

6.1.2 Αποτελέσματα και ακρίβεια του SVM

Ας δοκιμάσουμε τώρα να τρέξουμε το ίδιο πείραμα με το μοντέλου SVM, με ίδιες παραμέτρους και ίδιο μέγεθος train/test sets.

```
table(as.factor(svmDataSet[1:svmTestSize, 2]),
                svmResults[, "SVM_LABEL"])

svm.accuracy <- recall_accuracy(as.numeric
        (as.factor(svmDataSet[1:svmTestSize, 2])),
                svmResults[, "SVM_LABEL"])
svm.accuracy
```

Σε αυτήν την περίπτωση επειδή στον *container* θα μπορούσαμε να είχαμε περισσότερα από ένα μοντέλο, και κατά συνέπεια, στην μεταβλητή *svmResults* αποτελέσματα για περισσότερους από έναν αλγορίθμους, πρέπει για τα αποτελέσματα και την ακρίβεια να θέσουμε το αντίστοιχο *label* του μοντέλου, στην προκειμένη περίπτωση το "*SVM_LABEL*".

```
> table(as.factor(svmDataSet[1:svmTestSize, 2]),
+       svmResults[, "SVM_LABEL"])

            1   2
  negative 448  86
  positive 152 314
> svm.accuracy <- recall_accuracy(as.numeric(
+       as.factor(svmDataSet[1:svmTestSize, 2])),
+       svmResults[, "SVM_LABEL"])
> svm.accuracy
[1] 0.762
>
```

Figure 15. Αποτελέσματα και ακρίβεια με τον SVM

Με μια αντιστοίχως τυχαία εκτέλεση του πειράματος με αυτό το μοντέλο, έχουμε τα παρακάτω αποτελέσματα:

RESULTS	predicted	
actual	**tn** = 448	**fp** = 86
	fn = 152	**tp** = 314

Σε πρώτη ανάλυση βλέπουμε ότι τα tn & tp είναι περισσότερα από τα προηγούμενα αποτελέσματα, όπως επίσης και η ακρίβεια η οποία υπολογίστηκε στο **0.762** (ή **76.2%**).

Επίσης βλέπουμε πως η βελτίωση έγινε στην σωστή πρόβλεψη των θετικών tweets, ένα σημείο που φάνηκε να υστερεί ο *nb* στην εκτέλεση που προηγήθηκε.

Ωστόσο και πάλι για να μπορέσουμε να αξιολογήσουμε πλήρως την απόδοση αυτού του μοντέλου μέσω της μέτρησης της ακρίβειας του στην κατηγοριοποίηση συγκεκριμένου αριθμού από tweets, θα πρέπει να υπολογίσουμε μια μέση ακρίβεια, και στην προκειμένη περίπτωση αυτό είναι εύκολο καθώς το package *RTextTools* παρέχει αυτή την δυνατότητα με απλή υλοποίηση.

```
N <- 5
cross_validate(svmContainer, N, "SVM")
```

Έτσι πραγματοποιούμε μια **N-fold** cross-validation, με $N = 5$, και παίρνουμε τα παρακάτω αποτελέσματα:

```
> N <- 5
> cross_validate(container, N, "SVM")
Fold 1 Out of Sample Accuracy = 0.7865497
Fold 2 Out of Sample Accuracy = 0.7710367
Fold 3 Out of Sample Accuracy = 0.7789037
Fold 4 Out of Sample Accuracy = 0.7775966
Fold 5 Out of Sample Accuracy = 0.7822009
[[1]]
[1] 0.7865497 0.7710367 0.7789037 0.7775966 0.7822009

$meanAccuracy
[1] 0.7792575
```

Figure 16. Cross-validation και μέση ακρίβεια για τον SVM

Η μέση ακρίβεια που προκύπτει λοιπόν από το *Support Vector Machines* (***SVM***) model είναι ίση με **0.7792** (77.92% ~ **78%**) και είναι κατά 10% καλύτερη από την μέση ακρίβεια του **NB** model. Αυτό το ποσοστό είναι παραπάνω από ικανό για να καταλήξουμε στο σίγουρο συμπέρασμα ότι ο SVM αποδίδει καλύτερα στην κατηγοριοποίηση κειμένου, και συγκεκριμένα tweets με περιορισμένο μήκος διαθέσιμου κειμένου που να περιέχει αξιοποιήσιμο συναίσθημα. Αυτή η βελτίωση της απόδοσης ενός *classifier* που απλά χρησιμοποιεί **svm** έναντι του **nb**, προκύπτει πιθανώς από το γεγονός ότι μάλλον η συνθήκη για ανεξαρτησία που υποθέσαμε στο πρώτο μοντέλο, και είναι αναγκαία συνθήκη για την ορθή λειτουργεία του, δεν ικανοποιείται πλήρως και εισάγει λανθασμένες προβλέψεις.

6.2 Ειδικές περιπτώσεις βελτίωσης

Υποθέτοντας πως το μοντέλο SVM αποδίδει καλύτερα στο συγκεκριμένο πρόβλημα, θα πρέπει να εξετάσουμε κάποιες ειδικές περιπτώσεις γύρω από αυτή την τεχνική, και κάποια ειδικά απλά τεχνάσματα με το οποία ίσως μπορέσουμε να αυξήσουμε την ακρίβεια του μοντέλου. Θα δούμε στην συνέχεια ωστόσο, ότι πέρα από την καθαρή ακρίβεια των σωστά προβλέψιμων δεδομένων, υπάρχουν και κάποιες άλλες μετρήσεις που χαρακτηρίζουν την ποιότητα του αλγορίθμου μας.

Για αρχή, θα ήταν πολύ χρήσιμο να δούμε την επίδραση του μεγέθους του train-set στην απόδοση του μοντέλου. Το συνολικό μας σετ από δεδομένα είναι ~21k tweets, το οποίο συνεπάγεται ότι θα μπορούσαμε να κρατήσουμε απόλυτα σταθερό ένα κομμάτι από 1000 tweets όπως χρησιμοποιήθηκε σε όλη την διαδικασία ανάλυσης, και να κάνουμε πολλαπλές εκτελέσεις με διαφορετικά train-set στο εύρος των 1.000 - 20.000 tweets.

Ας δούμε λοιπόν τα αποτελέσματα αυτού του πειράματος με την βοήθεια μιας καμπύλης ακρίβειας, η οποία προκύπτει από την χρήση του πακέτου ***ggplot2*** με τις παρακάτω γραμμές κώδικα δεδομένων των αποτελεσμάτων ακρίβειας φυσικά:

```
#install.packages("ggplot2")
library(ggplot2)
#dev.off()

SVM.TRAINSET <- c(1000, 2500, 5000, 7500, 10000, 12500, 15000,
                  17500, 20000)

SVM.ACCURACY <- c(0.717, 0.727, 0.735, 0.750, 0.755, 0.761, 0.762,
                  0.772, 0.776)

acc.results <- data.frame(SVM.TRAINSET, SVM.ACCURACY)

## Plot the results
ggplot(acc.results, aes(x = SVM.TRAINSET, y = SVM.ACCURACY,
            col = "accuracy")) + geom_line() + geom_point()
```

Η καμπύλη ακρίβειας παρουσιάζεται εδώ και μας δείχνει την ακριβή σχέση ανάμεσα στο μέγεθος του συνόλου ελέγχου και στην προκύπτουσα ακρίβεια πρόβλεψης επί του σταθερού συνόλου ελέγχου των 1.000 tweets:

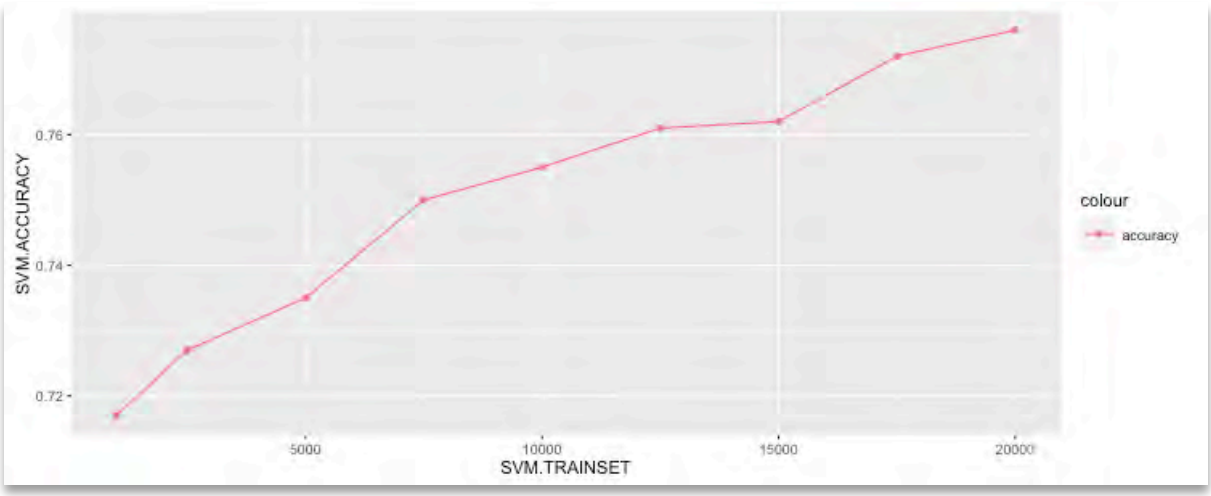

Figure 17. Ακρίβεια του svm συναρτήση του trainSet

Είναι λοιπόν φανερό πως με την αύξηση του train-set επέρχεται και αντίστοιχη βελτίωση της ακρίβειας, και αυτό είναι ένα πολύ βασικό συμπέρασμα καθώς υποδηλώνει ότι όσο πιο πολλά διαθέσιμα δεδομένα έχει ο αλγόριθμος στην διάθεση του για εκπαίδευση, τόσο πιο καλά λειτουργεί στην φάση της πρόβλεψης. Ωστόσο ακόμα και με λίγα tweets, δηλαδή με την "βασική γνώση", καταφέρνει να προβλέπει το δεδομένα ελέγχου σε ικανοποιητικό βαθμό, όμως μια αύξηση στην απόδοση της τάξης του 8% δεν μπορεί να θεωρηθεί αμελητέα.

Επιπλέον, σε δοκιμές εκτός καταγραφής, βγήκε το συμπέρασμα πως ακόμα και σε αυτό υπάρχει ένα ανώτατο όριο βελτίωσης, όμως αυτό καθορίζεται και από το αντίστοιχο test-set το οποίο διαθέτουμε για την έλεγχο. Η αύξηση των δεδομένων εκπαίδευσης βέβαια, έχει και ένα αρνητικό στην όλη διαδικασία. Αυτό φυσικά είναι η αύξηση του χρόνου εκπαίδευσης αλλά παράλληλα και του χρόνου πρόβλεψης με βάση το επαυξημένο μοντέλο. Ένα δείγμα αυτών των χρόνων που προέκυψαν κατά την προαναφερθείσα διαδικασία, είναι αρχικά για δεδομένα <5.000 περίπου 1sec και φτάνει για >15.000 κοντά στα 1-2min. Είναι ξεκάθαρο λοιπόν ότι κάτι τέτοιο δεν είναι επιθυμητό σε real-time εφαρμογές, για αυτό θα πρέπει η εκπαίδευση να γίνεται σε αρχικά στάδια αν πρόκειται να περιλαμβάνει πολλά data (1million tweets +).

Φυσικά, αν θέλουμε ακόμα καλύτερα και ρεαλιστικά αποτελέσματα, ο καλύτερος τρόπος να τα πετύχουμε είναι δημιουργώντας ένα train-set από "manually-labeled data", δηλαδή δεδομένα που έχουν κατηγοριοποιηθεί με το χέρι από ανθρώπους (και συγκεκριμένα γλωσσολόγους), καθώς έτσι είναι σίγουρο ότι θα έχουν ληφθεί υπόψη αρκετές περιπτώσεις που δεν είναι ξεκάθαρες. Για παράδειγμα, περιπτώσεις διπλής άρνησης, ειρωνείας, έμφασης, σαρκασμού, και κάθε είδους ιδιωματισμός της αγγλικής γλώσσας που δεν μπορεί να

καθοριστεί επακριβώς μέσα ενός προγράμματος. Τέτοια σύνολα δεδομένων υπάρχουν σε διάφορα websites που ασχολούνται γύρω από τον χώρο των linguistics, και προσφέρουν ελεύθερη και δημόσια χρήση για πειραματικούς σκοπούς. (Βέβαια θα πρέπει συνυπολογίσουμε, ότι σε πολύ απαιτητικές περιπτώσεις ανάλυσης, ίσως χρειαζόμαστε κάποιο σύνολο δεδομένων με πολύ περιορισμένο θέμα (topic) ώστε να αντανακλά επακριβώς τις ιδιαιτερότητες του πεδίου που μελετάμε και να δίνει άριστα ποσοστά ακρίβειας.)

Επιπροσθέτως, είναι εξαρχής γνωστό ότι η ανάλυση συναισθημάτων έχει μια επιπλέον δυσκολία στο twitter επειδή τα tweets είναι επί το πλείστων μικρά σε μέγεθος και συνεπώς είναι πιο δύσκολο για τους χρήστες να εισάγουν αρκετό συναίσθημα σε αυτά, και ακολούθως ακόμα πιο δύσκολο για ένα πρόγραμμα να αντιληφθεί αυτό το λιγοστό συναίσθημα που εμπεριέχεται. Έτσι θα είχε αρκετό ενδιαφέρον να δούμε στο σύνολο των δεδομένων που έχουμε, αν φιλτράρουμε με βάση τους συνολικούς χαρακτήρες των tweets, ποια θα είναι η προκύπτουσα ακρίβεια αλλά και πόσο μικραίνει το dataset με βάση αυτόν τον περιορισμό. Με την δοκιμή αυτή παίρνουμε τα παρακάτω αποτελέσματα:

char(threshold)	10	20	30	40	50	60
dataset	25743	21754	17774	14758	12129	9556
accuracy	0.779	0.786	0.779	0.813	0.804	0.794

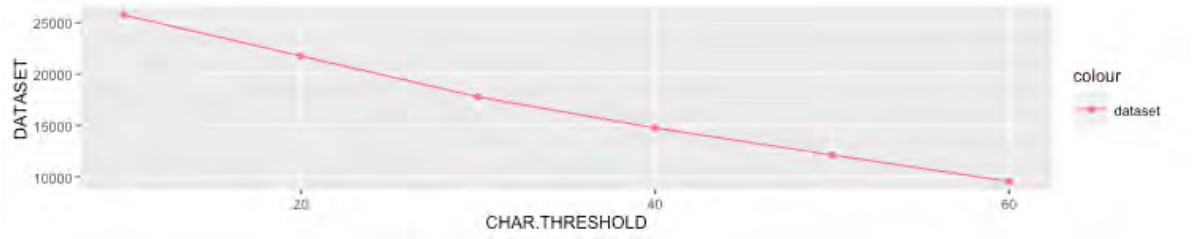

Figure 18. Δεδομένα εκπαίδευσης (svm) συναρτήσει του character threshold

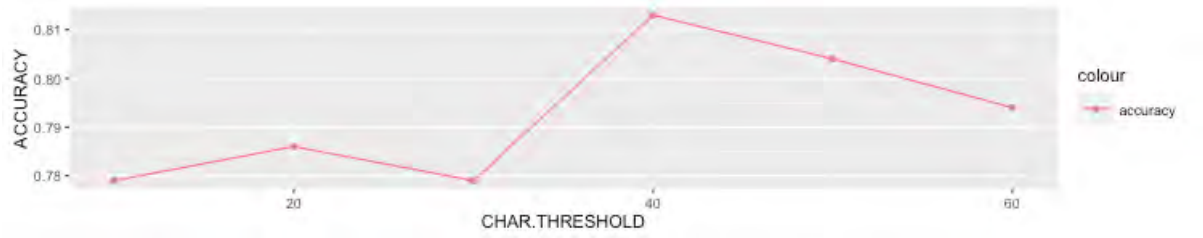

Figure 19. Ακρίβεια του svm συναρτήσει του character threshold

Συμπεραίνουμε λοιπόν ότι το βέλτιστο κατώφλι χαρακτήρων για το συγκεκριμένο σύνολο δεδομένων δοκιμάζοντας το στο σταθερό test-set των 1000 tweets, είναι οι 40 χαρακτήρες, καθώς σε εκείνο το σημείο μεγιστοποιείται η ακρίβεια, παρά την μείωση κατά >25% του train-set. Βλέπουμε ωστόσο, ότι για μεγαλύτερες τιμές αυτού του κατωφλίου, η απόδοση μειώνεται αρκετά, και μειώνεται σημαντικά και το σύνολο των εναπομεινάντων δεδομένων.

Έτσι, αυτή η παραμετροποίηση θα πρέπει να δίνεται με κάποιο τρόπο δυναμικά σε κάποιο πρόβλημα, ώστε να επιλέγονται κάθε φορά οι καλύτερες δυνατές τιμές. Η διαφορά φυσικά είναι της τάξης του 2%, ό μως για μια επιχείρηση ή έρευνα που επαφίεται στην λεπτομέρεια και έχει να κάνει με απόλυτες συγκρίσεις μεταξύ διαφόρων θεμάτων, αυτή η διαφορά μπορεί να είναι καθοριστικής σημασίας.

6.3 Τελικά Συμπεράσματα

Σκοπός αυτής της διπλωματικής ήταν καταρχήν να γίνει μια εισαγωγή στον κλάδο της ανάλυσης δεδομένων με τεχνικές μηχανικής μάθησης. Αυτός ο χώρος είναι εξ ορισμού χαοτικός και δεν θα μπορούσαμε να εμβαθύνουμε αν δεν περιορίζαμε τον τομέα στην ανάλυση συναισθημάτων. Πολλές γλώσσες προγραμματισμού μας δίνουν την δυνατότητα να εργαστούμε πάνω σε αυτό το κομμάτι, κυρίως η python, η R και η java. Επιλέχθηκε η R, καθώς ενδείκνυται συγκεκριμένα για εφαρμογές που περιέχουν statistical analysis, ωστόσο αν θέλαμε να δημιουργήσουμε ένα κομμάτι άλλης εφαρμογής η java θα ήταν προτιμότερη καθώς εξασφαλίζει και συμβατότητα με οποιοδήποτε σύστημα. Το twitter κατ΄επέκταση μας δίνει την δυνατότητα να πειραματιστούμε και να δοκιμάσουμε οποιαδήποτε προσέγγιση σε ρεαλιστικά δεδομένα καθώς παρέχει ελεύθερη πρόσβαση σε πολλά από αυτά.

Το μεγαλύτερο ενδιαφέρον πέραν της θεωρητικής ανάλυσης των μοντέλων έχει η σύγκριση αυτών σε ένα μικρό πείραμα με πραγματικά δεδομένα. Έτσι παρέχεται η δυνατότητα για σύγκριση των μοντέλων στα πλαίσια της εργασίας, και η αξιολόγηση των αποτελεσμάτων σε σύγκριση με θεωρητικές προσεγγίσεις από άλλα papers για την εξαγωγή ενός τελικού πορίσματος. Κύριο μέλημα μου είναι η αναλυτική περιγραφή όλων των βημάτων προς την δημιουργία ενός συνολικού εργαλείου, από την θεωρία στην πράξη, το οποίο θα είναι δυναμικό και παραμετροποιήσιμο, και συνεπώς θα μπορεί να χρησιμοποιηθεί και από κάποιον άλλο ενδιαφερόμενο για επέκταση του πειράματος ή κάποιο πιθανό future-work που θα προταθεί στην συνέχεια. Η δομή της εργασίας εισάγει επιστημονικούς όρους οι οποίοι απαιτούν κάποιο βασικό υπόβαθρο στα μαθηματικά την στατιστική και τον προγραμματισμό, ωστόσο δεν παύει να είναι απλή ώστε να μπορεί ο κάθε πιθανός αναγνώστης να αναπαράγει την ανάλυση που παρουσιάστηκε και να κατανοήσει βασικές έννοιες αλλά και δομές της R.

Βασικό συμπέρασμα για αρχή αποτελεί το γεγονός ότι ένα από τα πιο σημαντικά βήματα στην όλη διαδικασία είναι η συλλογή και η προ επεξεργασία των δεδομένων. Όσο πιο κοντά σε κάποιο συσχετιζόμενο topic είναι τα δεδομένα που παίρνουμε για την ανάλυση, τόσο μεγαλύτερη πιθανότητα έχουμε να φτιάξουμε έναν classification αλγόριθμο με υψηλή ακρίβεια. Πέρα όμως από τα δεδομένα αυτά καθ' αυτά θα πρέπει να δοθεί πολλή μεγάλη έμφαση στο στάδιο καθαρισμού και φιλτραρίσματος των δεδομένων καθώς αυτά θα είναι τα τελικά data που θα αποτελούν τα features σύμφωνα με τα οποία θα γίνει η εκπαίδευση και κατά συνέπεια η πρόβλεψη των ζητούμενων instances.

Αφού καθορίσουμε τις δυνατότητες του συστήματος που σκοπεύουμε να τοποθετήσουμε την εφαρμογή μας αλλά και τους απαιτούμενους χρόνους υπολογισμού που θέτει το πρόβλημα που θέλουμε να λύσουμε αλλά και ο ενδεχόμενος πελάτης που θα το χρησιμοποιήσει, πρέπει να πάρουμε μια απόφαση για το μέγεθος του συνόλου των δεδομένων εκπαίδευσης καθώς και αν αυτά θα είναι στατικά ή δυναμικά (real-time). Αποδείχτηκε προηγουμένως, ότι ένα σαφώς μεγαλύτερο train-set δίνει αρκετά καλύτερη ακρίβεια στα αποτελέσματα μας. Αυτό θα μπορούσε να είναι κρίσιμης σημασίας σε ευαίσθητες επιχειρηματικές εφαρμογές.

Αρκετά papers δηλώνουν πως καλύτερα αποτελέσματα λήφθηκαν με το μοντέλο svm σε αντίθεση του naïve Bayes, ωστόσο υπάρχουν και άλλοι που υποστηρίζουν το αντίθετο. Αυτό μπορεί να προκύψει αν εφαρμόζονται στην πράξη όλες οι προϋποθέσεις της χρήσης του $2^{ου}$ μοντέλου, και όχι απλά να υποτεθούν ως δεδομένες, ή στην περίπτωση που ο svm δεν έχει παραμετροποιηθεί σωστά με το αντίστοιχο kernel ώστε να μπορεί να υπολογιστεί σωστά ένα βέλτιστο hyperplane. Σύμφωνα με την δική μας προσέγγιση ωστόσο, είναι εξαιρετικά σαφές ότι μια βελτίωση στην απόδοση της τάξης του 10% δεν θα μπορούσε παρά να καταστήσει αυτό το μοντέλο ως το επικρατέστερο για text classification και συγκεκριμένα sentiment analysis.

Σημαντική λεπτομέρεια επίσης είναι ότι σε κάθε ανάλυση τέτοιων μοντέλων, ακόμα και “off-the-record”, πρέπει να γίνεται κάποιας μορφής cross validation των αποτελεσμάτων σε 3 επίπεδα:

1. Θα πρέπει να διερευνηθεί το κατά πόσο ο αλγόριθμος έχει την ίδια απόδοση και συμπεριφορά σε ένα συγκεκριμένο κομμάτι από δεδομένα στα πλαίσια ενός μεμονωμένου συνόλου από χρησιμοποιούμενα data.
2. Επιπλέον, θα πρέπει να ελεγχθεί το κατά πόσο συγκλίνουν τα αποτελέσματα ακρίβειας σε ένα τυχαίο εντελώς καινούριο dataset, ακόμα και αν αυτό περιέχει εντελώς διαφορετικό περιεχόμενο ως προς το topic.
3. Τέλος, είναι ιδιαίτερα σημαντικό να γίνεται διασταύρωση των αποτελεσμάτων κατηγοριοποίησης με τα πραγματικά class labels σε αντιστοιχία 1-1. Αυτό στην πράξη σημαίνει ότι δεν θα πρέπει απλά να υπολογίζουμε την ακρίβεια μετρώντας και συγκρίνοντας απλώς τα θετικά και αρνητικά instances μεταξύ labeled και predicted δεδομένων, αλλά αυστηρά ως τον λόγο των $\frac{tp+tn}{total}$, καθώς σε αντίθετη περίπτωση μπορεί αριθμητικά τα αποτελέσματα να συμπίπτουν (σε μετρήσιμα δείγματα), αλλά να μην ανταποκρίνονται στα σωστά δεδομένα προς πρόβλεψη.

Μην ξεχνάμε ωστόσο ότι το 68% σαν μέση ακρίβεια για το μοντέλο του naïve Bayes είναι πολύ κοντά στο επιθυμητό ποσοστό ακρίβειας ενός μέσου αλγορίθμου, και το 78% που μας δίνει ο svm τείνει να αγγίξει το μέγιστο επιτρεπτό όριο ακρίβειας που επιβάλει η κρίση του ανθρώπου, καθώς όπως προαναφέρθηκε, πάνω από ~80% ακρίβεια έχει ως αποτέλεσμα την διαφωνία τους για το υπολειπόμενο 20%, ένα φαινόμενο που εισάγει κάποιο παράγοντα "αμφισβήτησης" για το ευρύτερο σύνολο του μοντέλου κατηγοριοποίησης.[13] Συνεπώς, μπορούμε να θεωρήσουμε με ασφάλεια έναν classification algorithm ως βέλτιστο, αν η απόδοση του είναι στο όρια του 75-80%, εκτός και αν έχουμε μεγαλύτερες απαιτήσεις ή μικρότερο κατώφλι ευαισθησίας και συνεπώς η προσπάθεια για αύξηση της απόδοσης θα πρέπει να συνοδευτεί από την εκπαίδευση με ένα test-set από κοινώς αποδεκτά κατηγοριοποιημένα δεδομένα για το τελικό validation.

6.4 Future Work – Δημιουργία Shiny app

Σαν πρόταση για μελλοντική εργασία είναι η δημιουργία μιας web based εφαρμογής ανάλυσης δεδομένων σε real time με σκοπό την εξαγωγή την κοινής γνώμης πάνω σε κάποιο ζήτημα, εκμεταλλευόμενοι το συναίσθημα που εμπεριέχεται στα tweets σχετικών με το χώρο ανθρώπων όσο αφορά σε κάποιο συγκεκριμένο θέμα (topic).

Για τον σκοπό αυτό θα μπορούσε να χρησιμοποιηθεί αυτή η εφαρμογή ως πυρήνας και με κάποια σύγχρονα εργαλεία που παρέχει το framework της R, να δημιουργηθεί κάποιο online tool με ένα απλό user interface (UI) ώστε να μπορεί οποιοσδήποτε χρήστης να αναζητά δεδομένα για οποιοδήποτε topic, και να πραγματοποιεί το αντίστοιχο opinion mining με διάφορα διαθέσιμα charts ή export options.

Αυτή την δυνατότητα μας την παρέχει το framework **RShiny** (http://shiny.rstudio.com) που παρέχεται από το RStudio, και διευκολύνει την δημιουργία τόσο μιας *server.R* εφαρμογής όσο και μιας *ui.R*, που μαζί μπορούν πολύ εύκολα να εγκατασταθούν σε κάποιο server μέσω του *shinyapps* (https://www.shinyapps.io) και να είναι προσβάσιμες από το διαδίκτυο.

7 References

[1] A. Ortony, G. Clore, and A. Collins, "*The Cognitive Structure of Emotions in Language*", Cambridge Univ. Press, (1988).

[2] Ellen Riloff and Janyce Wiebe, "*Learning extraction patterns for subjective expressions*", Proceedings of the Conference on Empirical Methods in Natural Language Processing, (2003).

[3] George Forman, "*An Extensive Empirical study of feature selection Metrics for Text Classification*", Journal of Machine Learning Research, vol.3, (2004).

[4] Michael Wiegand and Alexandra Balahur, "*A Survey on th Role of Negation in Sentiment Analysis*", Proceedings of the Workshop on Negation and Speculation in Natural Language Processing, (2010).

[5] A. Neviarouska, H. Prendinger and M. Ishizuka, "*SentiFul: A Lexicon for Sentiment Analysis*", IEEE Transactions on Affective Computing, vol.2, (2011).

[6] S. ChandraKala and C. Sindhu, "*Opinion Mining and Sentiment Classification: A Survey"*, Department of Computer Science and Engineering, VEC, India, (2012).

[7] Richa Sharma, Shweta Nigam and Rekha Jain, "*Supervised Opinion Mining Techniques: A Survey*", Department of Computer Science, Banasthali University, Jaipur, India, (2013).

[8] Ayushi Dalmia, Manish Gupta and Vasudeva Varma "*Twitter Sentiment Analysis: The good, the bad, and the neutral!*", IIIT, Hyderabad, (2015).

[9] Erik Cambria(NUS), Björn Schuller(TUM), Yunqing Xia(TU), Catherine Havasi(MIT), "*New Avenues in Opinion Mining and Sentiment Analysis*", Knowledge-based approaches to Concept-level Sentiment Analysis, IEEE (2013).

[10] Allison Chang, "*R for Machine Learning*", Prediction: Machine Learning and Statistics(Spring 2012), MIT OpenCourseWare (http://ocw.mit.edu)

[11] Adam Westerski, "*Sentiment Analysis: Introduction and the State of the Art overview*", Universidad Politechnica de Madrid, Spain, (2009).

[12] Twitter Wikipedia, (https://en.wikipedia.org/wiki/Twitter)

[13] Sentiment Analysis Wikipedia, (https://en.wikipedia.org/wiki/Sentiment_analysis)

[14] Twitter Developer Blog, (https://dev.twitter.com)

[15] The R Project for Statistical Computing, (https://www.r-project.org)

[16] The Comprehensive R Archive Network, (https://cran.r-project.org)

[17] Text Classification and Naive Bayes, (https://web.stanford.edu/class/cs124/lec/naivebayes.pdf)

[18] Support Vector Machines and Machine Learning on documents, (http://nlp.stanford.edu/IR-book/html/htmledition/support-vector-machines-and-machine-learning-on-documents-1.html)