ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΕΦΑΡΜΟΣΜΕΝΩΝ ΜΑΘΗΜΑΤΙΚΩΝ ΚΑΙ ΦΥΣΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΟΜΕΑΣ ΜΗΧΑΝΙΚΗΣ

ΤΙΤΛΟΣ ΕΡΓΑΣΙΑΣ

Ανάπτυξη ενός Συστήματος Διεπαφής Προγραμματισμού Εφαρμογών για την Ανάκτηση Διαδικτυακών Δεδομένων με Τεχνικές Αναγνώρισης Προτύπων για την Κατηγοριοποίηση της Πολικότητας Κειμένων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΔΗΜΗΤΡΗΣ ΣΧΙΖΑΣ

Τριμελής Επιτροπή:

Κ. Σιέττος Αν. Καθηγητής (Επιβλέπων)
Δ. Γκούσης, Καθηγητής
Γ. Ματσόπουλος, Αναπλ. Καθηγητής ΣΗΜΜΥ

Αθήνα, Οκτώβριος 2014

ΠΕΡΙΕΧΟΜΕΝΑ

ΠΕΡΙΛΗΨΗ

Το μέγεθος του διαδικτύου αυξάνεται συνεχώς και η διαχείριση της πληροφορίας που το περιγράφει έχει οδηγήσει στη δημιουργία νέων επιστημών. Η εξόρυξη δεδομένων (data mining και αργότερα web mining) και η μηχανική μάθηση (machine learning) είναι δύο από αυτές, που μαζί με το internet αναπτύσσονται ραγδαία. Ο τρόπος με τον οποίο μπορεί ένα υπολογιστικό σύστημα να «τραβήξει» την πληροφορία μέσω του διαδικτύου πλέον ποικίλει. Σε αυτή την εργασία εξορύξαμε την πληροφορία του διαδικτύου (web content mining) μέσω ενός crawler καθώς και με την "κατανάλωση" (consume - API client) διαφόρων web services που παρέχουν πακέτα πληροφορίας με στόχο την ανάλυση του συναισθήματος (sentiment analysis) και ειδικότερα της πολικότητας (polarity) κειμένων από το διαδίκτυο. Αυτός είναι ο βασικός τρόπος που ακολουθούν και μεγάλες εταιρείες για να έχουν πρόσβαση στη πληροφορία, μαζί φυσικά και με πρόσβαση σε ιδιωτικές βάσεις δεδομένων (private database).

Abstract

The size of the internet is growing very fast and managing the information that describes it has led to the creation of new fields in science. Data and machine learning are two of them that along with the internet are growing rapidly. The way in which a computer system can "pull" the information through the Internet varies. In the current work, the information is mined from the internet (web content mining) though a crawler and by consuming various web services that provide information packets to perform a systematic sentiment analysis with a focus on the identification of the polarity of the contents.

ΕΙΣΑΓΩΓΗ

Είναι γεγονός ότι το διαδίκτυο (internet) πρωταγωνιστεί στη ζωή μας επηρεάζοντας σημαντικά την ενημέρωση μας. Η ταχεία ανάπτυξη του διαδικτύου καθώς και η εισαγωγή των πληροφοριακών συστημάτων σε υπηρεσίες και οργανισμούς τόσο για την εσωτερική τους λειτουργία όσο και για την εξυπηρέτηση του κοινού, έχουν ως αποτέλεσμα τη συνεχή παραγωγή, διακίνηση και αποθήκευση τεράστιου όγκου πληροφορίας. Αναφορικά το 2012 είχαν δημιουργηθει στο διαδίκτυο δεδομένα που ξεπερνούσαν τα 2.7 billion terabytes και καθημερινά δημιουργούνται 2500 terabytes [1]. Μεταξύ αυτών είναι και τα κείμενα που γράφονται και διαδίδονται καθημερινά μέσω του διαδικτύου τα οποία πολλές φορές δεν ειναι δομημένα και μπορεί να έχουν γραφθεί μέσω email, σε μορφή κώδικα HTML, σε tweet και με άλλους πολλούς τρόπους. Από τη στιγμή που το διαδίκτυο έδειξε τάσεις για ραγδαία ανάπτυξη έγινε εύκολα αντιληπτή η ανάγκη για ανάπτυξη αυτοματοποιημένων τεχνικών για την ανακάλυψη, ανάλυση και επεξεργασία πληροφοριών από κειμενικά δεδομένα. Πόσο όμως μπορεί το internet να επηρεάσει τόσο πολύ τη ζωή ένος ανθρώπου; Αναφορικά, στο twitter γίνονται 600 εκατομμύρια tweets την ημέρα και το twitter αποτελεί βασικό εργαλείο ενημέρωσης για πολλούς, επηρεάζοντας σημαντικά την καθημερινοτητά τους. Όλη αυτή η πληροφορία η οποία δημιουργείται και διανέμεται μέσω των ιστοσελίδων και των εφαρμογών στους χρήστες, περιέχει την ξεχωριστή αυτή πληροφορία που πολλές φορές παίζει καθοριστικό ρόλο σε θέματα λήψης αποφάσεων. Όπως για παράδειγμα συμβαίνει με το χρηματιστήριο, όπου γι'αυτό γράφονται συνεχώς άρθρα τα οποία ενημερώνουν τους ανθρώπους και οι αυτοί με τη σειρά τους λαμβάνουν την απόφαση για το αν θα αγοράσουν ή θα πουλήσουν κάποια μετοχή ή ίσως ένα συνάλλαγμα. Θα ήταν λοιπόν θετικό αν μπορέσουμε να αναλύσουμε αυτοματοποιημένα μέσω μια μηχανής και αναλύσουμετην πληροφορία ενός κειμένου (π.χ. να αναγνωρίσουμε το υποκείμενο συναίσθημα-μύνημα (καλά νέα-κακά νέα) χωρίς να το διαβάσουμε και όλα αυτά σε ελάχιστο χρόνο.

Αναλύσαμε τα δεδομένα αυτά και με τις διαδικασίες που περιγράφονται στα κεφάλαια 1, 2 και 3, και μπορέσαμε να παράγουμεένα τελικό λεξιλόγιο. Το λεξιλόγιο αυτό αποτελείται από ένα σύνολο θετικών και αρνητικών λέξεων και είναι σε θέση να δείξει, αναλύοντας κείμενα που αφορούν οικονομικά νέα αν το κείμενο αναφέρει κάτι θετικό ή αρνητικό πάνω στο θέμα του.

Στο πρώτο κεφάλαιο περιγράφονται τα βασικά στοιχεία της τεχνητής νοημοσύνης, μιας επιστήμης που γίνεται ολοένα και πιο γνωστή όσο μεγαλώνει το διαδίκτυο, οι τεχνολογίες και η οικονομία που στηρίζεται πάνω σε αυτό.

Στο δεύτερο κεφάλαιο αναλύεται ένας κομμάτι της τεχνητής νοημοσύνης, η εξόρυξη δεδομένων που μαζί με την μηχανική μάθηση θεωρούνται από τα πιο σημαντικά πεδία της.

Στο τρίτο κεφάλαιο γίνεται αναφορά σε μια ειδική κατηγορία της εξόρυξης δεδομένων, την εξόρυξη κειμένου (text mining). Πάνω σε αυτό το κεφάλαιο καθώς και στα άλλα δύο βασίστηκε ένα μεγάλο τμήμα της εργασίας αφού έπρεπε εν τέλει να αναλυθεί ο τεράστιος όγκος πληροφορίας που παράγεται και αναμεταδίδεται καθημερινά στο διαδίκτυο.

Στο τέταρτο κεφάλαιο αναλύεται το πρόβλημα που αντιμετωπίσαμε με στόχο την ανγνώριση της πολυκότητας από ειδησιογραφικά κείμενα διαδικτύου καθώς και την διαδικασία επίλυσης με τα αντίστοιχα αποτελέσματα.

ΚΕΦΑΛΑΙΟ 1: ΤΕΧΝΗΤΗ ΝΟΗΜΟΣΥΝΗ

1.1 Εισαγωγή στη Τεχνητή Νοημοσύνη

Ο τελευταίος αιώνας έχει χαρακτηριστεί από πολλούς ως ο αιώνας της πληροφορίας. Βασικό χαρακτηριστικό επιτυχημένων εταιρειών σε διάφορους κλάδους είναι η συγκέντρωση και η αξιοποίηση της πληροφορίας και των δεδομένων. Εταιρείες οπώς η IBM και η Google το αντιλήφθηκαν γρήγορα και εδραιώθηκαν στο κλάδο αυτό από το 1911 και το 1998 αντίστοιχα. Αυτή η ικανότητα της διαχείρισης και ανάλυσης της πληροφορίας, σε θέματα που την αφορούν πολλές φορές είναι κρίσιμη για τη βιοσιμότητα της εταιρείας. Η Τεχνητή Νοημοσύνη (Artificial Intelligence) και ειδικότερα η χρήση των τεχνικών Εξόρυξης Δεδομένων (Data Mining) βοηθά σε τέτοιου είδους αναλύσεις πάνω σε ποιοτικά και αριθμητικά δεδομένα. Οι τελευταίες δίνουν τη δυνατότητα εξαγωγής σχέσεων και κανόνων μέσω της χρήσης ηλεκτρονικών υπολογιστών.

1.2 Ορισμός

Το να δοθεί ένας ακριβής ορισμός γύρω από τον όρο Τεχνητή Νοημοσύνη, είναι κάτι πολύπλοκο και σύνθετο καθώς η έννοια της λέξης "νοημοσύνη", μπορεί να προσεγγιστεί με διάφορους τρόπους. Ο καθηγητής του MIT στον τομέα της Τεχνητής Νοημοσύνη, Marvin Minsky, υποστηρίζει ότι η τεχνητή νοημοσύνη είναι η επιστήμη του να κάνεις τις μηχανές να κάνουν πράγματα που θα απαιτούσαν τον ανθρώπινο παράγοντα για να πραγματοποιηθούν [2]. Τεχνητή Νοημοσύνη είναι ο τομέας της επιστήμης των υπολογιστών, που ασχολείται με τη σχεδίαση ευφυών υπολογιστικών συστημάτων, δηλαδή συστημάτων που επιδεικνύουν χαρακτηριστικά που σχετίζονται με τη νοημοσύνη στην ανθρώπινη συμπεριφορά κατά τον Barr και Feigenbaum [3]. Ένας άλλος ορισμός από τον Elaine Rich αναφέρει πως Τεχνητή Νοημοσύνη είναι η μελέτη του πως να κάνουμε τους ηλεκτρονικούς υπολογιστές να κάνουν πράγματα για τα οποία, προς το παρόν, οι άνθρωποι είναι καλύτεροι [4]. Επίσης διάσταση απόψεων υπάρχει και στο εάν η Τεχνητή Νοημοσύνη αποτελεί επιστήμη ξεχωριστή, ή απλά είναι ένας κλάδος της επιστήμης των υπολογιστών. Ωστόσο κοινά αποδεκτό από την επιστημονική κοινότητα είναι ότι η Τεχνητή Νοημοσύνη σε μια μηχανή είναι κάτι πολύ περισσότερο από απλά την εισαγωγή δεδομένων σε αυτή, καθώς συμπεριλαμβάνει επίσης την δυνατότητα αξιολόγησης και εξαγωγής πληροφορίας από αυτά και τέλος την εκμάθηση και βελτίωση μέσα από τις εμπειρίες της επαφής της με το περιβάλλον.
Η Τεχνητή Νοημοσύνη αποτελεί σημείο τομής μεταξύ πολλών επιστημών όπως της επιστήμης υπολογιστών, της ψυχολογίας, της φιλοσοφίας, της νευρολογίας, της

γλωσσολογίας και της επιστήμης των μηχανικών, με στόχο τη σύνθεση ευφυούς συμπεριφοράς μια μηχανής με βασικά χαρακτηριστικά συλλογιστικής, μάθησης και προσαρμογής σε ένα πληροφοριακό περιβάλλον.
Μια από τις κατηγοριοποιήσεις είναι η :
συμβολική (symbolic) τεχνητή νοημοσύνη, η οποία επιχειρεί να εξομοιώσει την ανθρώπινη νοημοσύνη αλγοριθμικά χρησιμοποιώντας σύμβολα και λογικούς κανόνες υψηλού επιπέδου και θεωρήθηκε από πολλούς ερευνητές ως η λύση για την δημιουργία ενός μηχανήματος με γενική τεχνητή νοημοσύνη κατά τις δεκαετίες του 1960 και 1970,
και η **υπό συμβολική** (sub-symbolic) τεχνητή νοημοσύνη, η οποία προσπαθεί να αναπαράγει την ανθρώπινη ευφυΐα χρησιμοποιώντας στοιχειώδη αριθμητικά μοντέλα που συνθέτουν επαγωγικά νοήμονες συμπεριφορές με τη διαδοχική αυτοοργάνωση απλούστερων δομικών συστατικών (συμπεριφορική τεχνητή νοημοσύνη), προσομοιώνουν πραγματικές βιολογικές διαδικασίες όπως η εξέλιξη των ειδών και η λειτουργία του εγκεφάλου (υπολογιστική νοημοσύνη), ή αποτελούν εφαρμογή στατιστικών μεθοδολογιών σε προβλήματα τεχνητής νοημοσύνης.

Η διάκριση σε συμβολικές και υπό συμβολικές προσεγγίσεις αφορά τον χαρακτήρα των χρησιμοποιούμενων εργαλείων, ενώ δεν είναι σπάνια η σύζευξη πολλαπλών προσεγγίσεων (διαφορετικών συμβολικών, υπό συμβολικών, ή ακόμα συμβολικών και υπό συμβολικών μεθόδων) κατά την προσπάθεια αντιμετώπισης ενός προβλήματος με βάση τον επιθυμητό επιστημονικό στόχο η τεχνητή νοημοσύνη κατηγοριοποιείται σε άλλου τύπου ευρείς τομείς, όπως επίλυση προβλημάτων, μηχανική μάθηση, ανακάλυψη γνώσης, συστήματα γνώσης κλπ. Επίσης υπάρχει επικάλυψη με συναφή επιστημονικά πεδία όπως η μηχανική όραση, η επεξεργασία φυσικής γλώσσας, η ρομποτική κλπ.

1.3 Στόχος της Τεχνητής Νοημοσύνης

Το γενικό πρόβλημα της προσομοίωσης (ή δημιουργίας) των πληροφοριών έχει αναλυθεί σε μια σειρά από συγκεκριμένα υπό-προβλήματα. Αυτά αποτελούνται από συγκεκριμένα χαρακτηριστικά ή δυνατότητες που οι ερευνητές θα ήθελαν ένα ευφυές σύστημα να διαθέτει. Τα χαρακτηριστικά που περιγράφονται παρακάτω έχουν λάβει την μεγαλύτερη προσοχή [5].

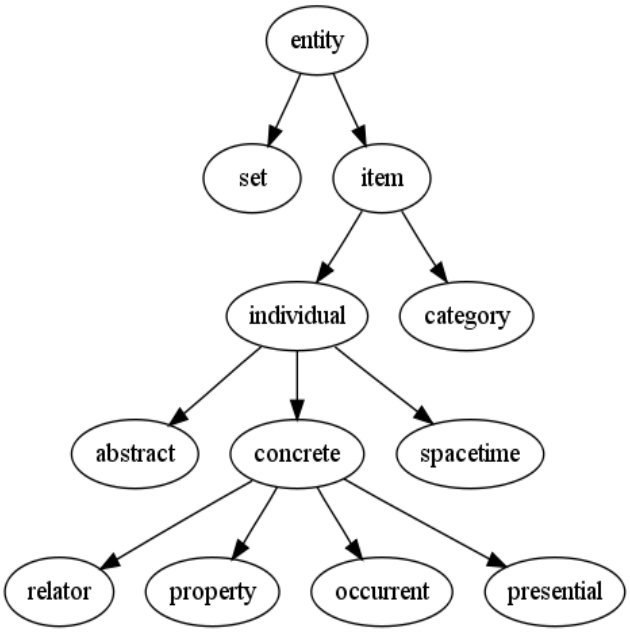
1.3.1 Μείωση, αξιολόγηση, επίλυση

Ερευνητές της τεχνητής νοημοσύνης ανέπτυξαν αλγόριθμους, που μιμούνταν την αιτιολόγηση βήμα προς βήμα όπως αυτή που ακολουθούν οι άνθρωποι για να λύσουν σπαζοκεφαλιές. Από τα τέλη της δεκαετίας του 1980 και του 1990, η έρευνα πάνω στη τεχνητή νοημοσύνη είχε σαν αποτέλεσμα την ανάπτυξη επιτυχημένων μεθόδων για την

αντιμετώπιση της αβέβαιης ή ατελούς πληροφορίας, χρησιμοποιώντας έννοιες από τις πιθανότητες και την οικονομία.
Για τα δύσκολα προβλήματα, οι περισσότεροι από αυτούς τους αλγορίθμους απαιτούν τεράστια υπολογιστική δύναμη. Το ποσό της μνήμης ή της ώρας που χρειάζεται ένας υπολογιστής για να λύσει ένα πρόβλημα γίνεται πολύ μεγάλο όταν το πρόβλημα πηγαίνει πέρα από ένα ορισμένο μέγεθος. Η αναζήτηση για πιο αποτελεσματική επίλυση των προβλημάτων των αλγορίθμων αποτελείυψηλή προτεραιότητα για την Τεχνητή Νοημοσύνη.
Η Τεχνητή Νοημοσύνη έχει σημειώσει κάποια πρόοδο στη μίμηση της υπο-συμβολικής επίλυσης προβλημάτων (δηλαδή τη γρήγορη, διαισθητική λήψη αποφάσεων αντί για την συνειδητή, βήμα-προς-βήμα): ενσωματώνοντας προσπάθειες για δημιουργία νευρωνικών δικτύων που σαν στόχο έχουν την προσομοιώση της δομής στο εσωτερικό του εγκεφάλου που δημιουργούν αυτήν την επιδεξιότητα και στατιστικές προσεγγίσεις ώστε η τεχνιτή νοημοσύνη να μιμηθεί την πιθανολογική φύση της ανθρώπινης ικανότητας να μαντέψει.

1.3.2 Αναπαράσταση γνώσης

Η αναπαράσταση της γνώσης και της μηχανικής γνώσης είναι κεντρικής σημασίας για την Τεχνητή Νοημοσύνη. Ανάμεσα στα πράγματα που η Τεχνητή Νοημοσύνη χρειάζεται να αναπαραστήσει είναι: αντικείμενα, τις ιδιότητες, τις κατηγορίες και τις σχέσεις μεταξύ των αντικειμένων, καταστάσεις, γεγονότα, καταστάσεις και το χρόνο, τα αίτια και τις επιπτώσεις, η γνώση για τη γνώση (ό, τι γνωρίζουμε για το τι άλλοι άνθρωποι γνωρίζουν) και πολλούς άλλους τομείς που έχουν ερευνηθεί λιγότερο. Μια αναπαράσταση του "τι υπάρχει" είναι μια οντολογία: το σύνολο των αντικειμένων, τις σχέσεις, τις έννοιες και ούτω καθεξής που η μηχανή γνωρίζει για αυτά.



Σχήμα 1.1 Η οντολογία αναπαριστά τη γνώση σαν ένα σύνολο εννοιών (πηγή: isquared.wordpress.com)

1.3.3 Μηχανική Μάθηση - Machine Learning

Μηχανική μάθηση είναι η μελέτη των αλγορίθμων που βελτιώνουν αυτόματα μέσα από τις εμπειρίες και έχει κεντρική σημασία για την έρευνα της Τεχνητής Νοημοσύνης.
Ανεπίβλεπτη (unsupervised) μάθηση είναι η ικανότητα να ανακαλύπτονται αυτόματα μοτίβα σε ένα σύνολο δεδομένων που εισήχθηκαν στο πρόγραμμα. Στην επιβλεπόμενη (supervised) μάθηση ή μάθηση με επίβλεψη, ο αλγόριθμος κατασκευάζει μια συνάρτηση που απεικονίζει δεδομένες εισόδους σε γνωστές, επιθυμητές εξόδους (σύνολο εκπαίδευσης), με απώτερο στόχο τη γενίκευση της συνάρτησης αυτής και για εισόδους με άγνωστη έξοδο (σύνολο ελέγχου). Η επιβλεπόμενη μάθηση περιλαμβάνει την ταξινόμηση (classification) και την αριθμητική παλινδρόμησης (regression).
Η ταξινόμηση χρησιμοποιείται για να καθοριστει σε ποια κατηγορία ανήκει κάτι. Αυτό γίνεται αφού παρατηρηθούν ορισμένα παραδείγματα απ;ο διάφορες άλλες κατηγορίες.
Παλινδρόμηση είναι η προσπάθεια να παραχθεί μια συνάρτηση που περιγράφει τη σχέση μεταξύ των δεδομένων που εισάγονται και εξάγονται αντίστοιχα και να προβλέψει πως αλλάζουν τα δεδομένα που εξάγονται σε σχέση με τις αλλάγες αυτών που εισάγονται.

1.3.4 Επεξεργασία φυσικής γλώσσας - Natural Language Processing

Η επεξεργασία της φυσικής γλώσσας δίνει στα μηχανήματα την ικανότητα να διαβάζουν και να κατανοούν τις γλώσσες που μιλούν οι άνθρωποι. Αυτή η επιστημονική περιοχή καλύπτει τον σχεδιασμό και την υλοποίηση υπολογιστικών μοντέλων της φυσικής γλώσσας. Ένα αρκετά ισχυρό σύστημα επεξεργασίας φυσικής γλώσσας θα λειτουργούσε από διεπαφές φυσικής γλώσσας και απόκτηση της γνώσης κατευθείαν από πηγές γραμμένες από ανθρώπους, όπως για παράδειγμα τα κείμενα των ειδήσεων. Μερικές απλές τεχνικές επεξεργασίας φυσικής γλώσσας περιλαμβάνει την ανάκτηση πληροφοριών (ή εξόρυξη κειμένου) και αυτόματη μετάφραση.
Μια κοινή μέθοδος επεξεργασίας και εξόρυξης της πληροφορίας από τη φυσική γλώσσα είναι μέσω σημασιολογικής ευρετηρίασης (semantic indexing) [6].

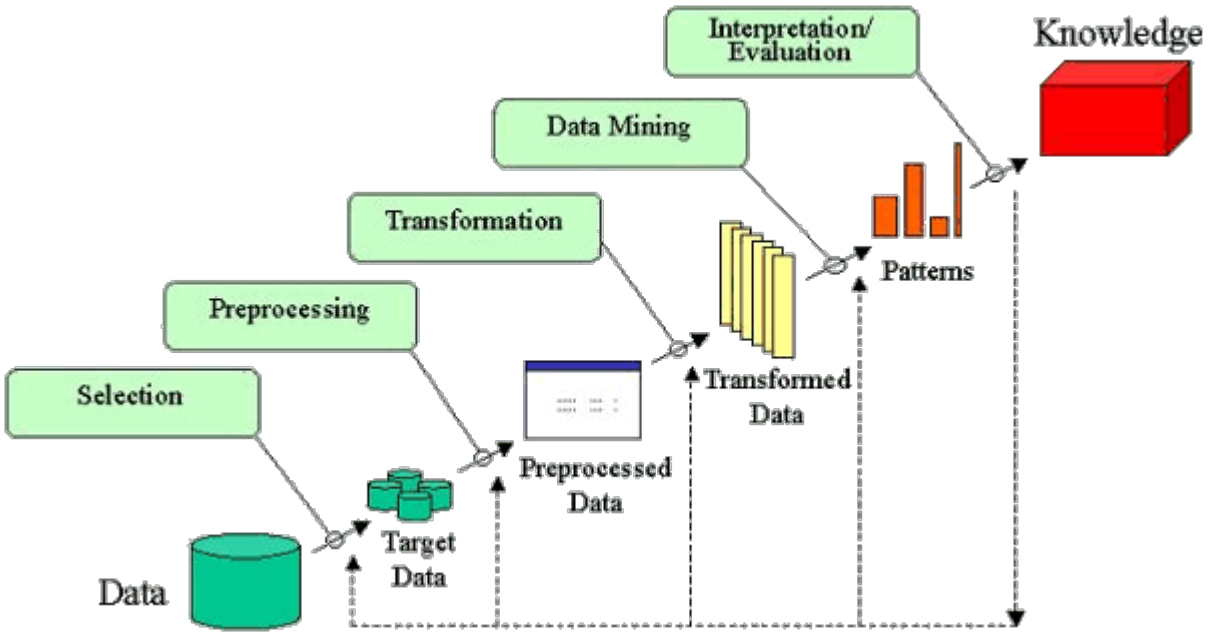
ΚΕΦΑΛΑΙΟ 2 : Εξόρυξη δεδομένων - Data Mining

2.1 Εισαγωγή

Εξόρυξη δεδομένων ορίζεται η αυτόματη ή ημιαυτόματη μη τετριμμένη διαδικασία εξαγωγής σημαντικών πληροφοριών και προτύπων από βάσεις δεδομένων με τη χρήση μιας αυτοματοποιημένης διαδικασίας.
Η διαδικασία βασίζεται στη χρησιμοποίηση αλγορίθμων οι οποίοι ανακαλύπτουν τους κανόνες μεταξύ των μεταβλητών των δεδομένων. Ο απώτερος στόχος της διαδικασίας εξόρυξης δεδομένων είναι να αποσπαστεί η πληροφορία από ένα σύνολο δεδομένων και να μετατραπεί σε μια κατανοητή δομή για περαιτέρω χρήση.
Η εξόρυξη δεδομένων είναι το σημείο της ανάλυσης κατα τη διαδικασία της Ανάκτησης της γνώσης μέσω Βάσεων Δεδομένων (Knowledge Discovery in Databases - KDD) [7].

2.2 Διαδικασία KDD

Για να εξορύξουμε τα δεδομένα πρέπει να υπάρχει η απαιτούμενη γνώση τους, ώστε να απαλειφθούν επιτυχώς οι αρνητικοί παράγοντες. Η διαδικασία KDD απεικονίζεται στο σχήμα 2.1 και αποτελείται απο τα εξής:



Σχήμα 2.1 Η διαδικασία KDD (πηγή: www2.cs.uregina.ca)

- Προεπεξεργασία των δεδομένων: περιλαμβάνει τον καθορισμό, την ολοκλήρωση, την επιλογή δεδομένων που θα εξορυχθούν και το μετασχηματισμό τους σε μορφή κατάλληλη προς εξόρυξη.
- Εξόρυξη των δεδομένων: αποτελεί το κύριο βήμα της διαδικασίας, όπου εφαρμόζονται οι αλγόριθμοι εξόρυξης και βρίσκεται η πληροφορία.
- Αποτίμηση και αναπαράσταση των αποτελεσμάτων: γίνεται η αποτίμηση της εξορυχθείσας πληροφορίας, ώστε να διατηρηθεί μόνο η νέα και χρήσιμη.
- Τέλος γίνεται η αναπαράσταση της πληροφορίας μέσω οπτικών μέσων και ενδεχομένως γίνεται η αποθήκευση της ως γνώση υποβάθρου.

2.3 Μέθοδοι εξόρυξης

Υπάρχουν πολλές μεθόδοι εξόρυξης δεδομένων (data mining functionalities). Αναφορικά μερικές είναι:

- Ταξινόμηση (Classification)
- Ομαδοποίηση (Clustering)
- Συσχέτιση (Association)
- Χαρακτηρισμός (Characterization)
- Στατιστική Ανάλυση (Statistical Analysis)
- Αριθμητική Πρόβλεψη (Numerical Prediction)
- Παλινδρόμηση (Regression)
- Εντοπισμός ψευδών αντικειμένων (Outlier detection)

Από αυτές οι πρώτες τρεις είναι και οι πιο σημαντικές στο κομμάτι της εξόρυξης της πληροφορίας.

2.3.1 Ταξινόμηση

Για ένα σύνολο δεδομένων, το οποίο καλείται σύνολο εκμάθησης (training set) κάθε δεδομένο περιέχει ιδιότητες, από τις οποίες, μια ονομάζεται ιδιότητα κλάσης. Η ιδιότητα κλάσης ορίζει και τα δεδομένα ως ταξινομημένα. Σκοπός είναι η εύρεση ενός μοντέλου πρόβλεψης της τιμής της ιδιότητας κλάσης από τις τιμές των υπολοίπων ιδιοτήτων για δεδομένα που δεν ανήκουν στο σύνολο εκμάθησης και άρα η τιμής της ιδιότητας κλάσης δεν είναι γνωστή. Αυτή τη διαδικασία μπορούμε να την υλοποιήσουμε με τα ακόλουθα βήματα:

- Εισάγουμε το σύνολο εκμάθησης (σύνολο δεδομένων) σε ένα αλγόριθμο ταξινόμησης.
- Τα δεδομένα ταξινομούνται και ο αλγόριθμος κατανοεί τον τρόπο με τον οποίο έγινε η ταξινόμηση.
- Ο αλγόριθμος είναι πλέον σε θέση να ταξονομίσει νεα δεδομένα.

Το σύνολο των κανόνων ονομάζεταιι ταξινομητής (classifier). Οι αλγόριθμοι ταξινόμησης μπορούν να δημιουργήσουν λίστες αποφάσεων ή δένδρα αποφάσεων αναλόγος με το είδος της ταξινόμησης.

2.3.2 Ομαδοποίηση

Οι αλγόριθμοι της ομαδοποίησης μοιάζουν με αυτούς της ταξινόμησης αλλά δεν είναι ίδιοι. Πάλι υπάρχει ένα σύνολο δεδομένων με κάποιες ιδιότητες αλλά δεν υπάρχει η ιδιότητα κλάσης. Άρα τα δεδομένα δεν είναι προταξινομημένα όπως στη ταξινόμηση. Ο σκοπός είναι ο εντοπισμός ομάδων δεδομένων με μεγάλη ομοιότητα, ενώ δεδομένα διαφορετικών ομάδων διαφέρουν σημαντικά. Σε κάποιους αλγορίθμους ομαδοποίησης ορισμένα δεδομένα μπορούν να ανήκουν ταυτόχρονα σε διαφορετικές ομάδες δεδομένων.

2.3.3 Συσχέτιση

Δίνεται ένα σύνολο δεδομένων, όπου στο κάθε ένα υπάρχουν αντικείμενα στα οποία έχει συμβεί απο κοινού μια ενέργεια, σκοπός είναι η εύρεση των σημαντικότερων αλληλεξαρτήσεων μεταξύ των διάφορων αντικειμένων στο σύνολο εκπαίδευσης. Κλασσικό παράδειγμα η ανάλυση του καλαθιού της αγοράς (market basket analysis).

2.4 Αλγόριθμοι Εξόρυξης Δεδομένων

Οι αλγόριθμοι εξόρυξης δεδομένων είναι πολλοί, παρακάτω θα αποτυπωθούν ορισμένοι που θεωρήθηκαν ως οι πιο σημαντικοί στο πεδίο αυτό καθώς και άλλοι αλγόριθμοι [8].

2.4.1 Αλγόριθμος C4.5

Συστήματα που κατασκευάζουν ταξινομητές είναι από τα πιο συχνά εργαλεία που χρησιμοποιούνται συνήθως στον τομέα της εξόρυξης δεδομένων. Τέτοια συστήματα λαμβάνουν ως είσοδο ένα σύνολο από δεδομένα, όπου το κάθε ένα αποτελείται από

υποδείγματα που έχουν κάποιες ιδιότητες, τα οποία μπορούν να παράγουν ένα ταξινομητή.
Ο αλγόριθμος C4.5 είναι ο διάδοχος αλγόριθος του ID3 ο οποίος αναπτύχθηκε το 1980 από τον J. Ross Quinlan [9] και αφορούσε τα δέντρα αποφάσεων (Iterative Dichotomiser). Ο C4.5 παράγει ταξινομητές που εκφράζονται σαν δέντρα απόφασης, αλλά μπορεί να κατασκευάσει ταξινομητές σε πιο κατανοητή μορφή. Ο αλγόριθμος έχει την ακόλουθη δομή:

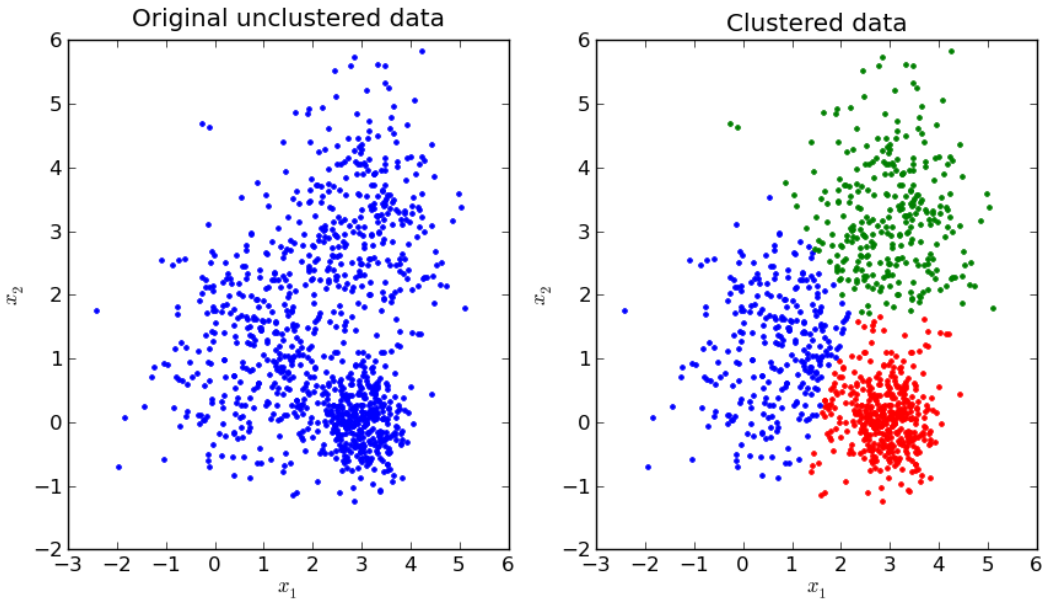
- Αν όλα τα υποδείγματα στο σύνολο S ανήκουν στην ίδια τάξη ή το σύνολο υποδειγμάτων S είναι πολύ μικρό, το δέντρο αποτελείται από ένα φύλλο με όνομα την τάξη με τη μεγαλύτερη συχνότητα εμφάνισης στο σύνολο S.
- Αλλιώς γίνεται ένας έλεγχος με βάση μια μεταβλητή με δύο ή περισσότερες πιθανές τιμές. Ο έλεγχος αυτός γίνεται η βάση του δέντρου και από αυτή ξεκινούν κλαδιά για κάθε πιθανό αποτέλεσμα από τον έλεγχο. Έτσι το σύνολο των υποδειγμάτων διαιρείται σε μικρότερα σύνολα ανάλογα με το αποτέλεσμα του ελέγχου και για κάθε ένα από τα καινούργια σύνολα ακολουθείται η ίδια αναδρομική διαδικασία.

Συνήθως υπάρχουν πολλοί έλεγχοι που μπορούν να επιλεγούν. Συγκεκριμένα ο C4.5 χρησιμοποιεί δύο ευρετικά (heuristic) κριτήρια για να βαθμολογήσει αυτούς τους ελέγχους και να επιλέξει τον προτιμότερο. Το πρώτο κριτήριο είναι το **κέρδος** πληροφορίας, το οποίο ελαχιστοποιεί την συνολική εντροπία των παραγόμενων υποσυνόλων. Το δεύτερο κριτήριο είναι ο **λόγος κέρδους**, ο οποίος προκύπτει από τη διαίρεση του κέρδους πληροφορίας προς την πληροφορία που προκύπτει από τις πιθανές τιμές του ελέγχου.
Οι μεταβλητές που χρησιμοποιούνται μπορεί να είναι ονομαστικές ή αριθμητικές και αυτό καθορίζει την μορφή του ελέγχου. Για τις αριθμητικές μεταβλητές οι πιθανές μορφές είναι $(A \leq h, A \geq h)$, όπου το A είναι η μεταβλητή και το h είναι ένα όριο που προκύπτει ταξινομώντας το σύνολο των υποδειγμάτων με βάση τις τιμές του A και επιλέγοντας το όριο μεταξύ δύο διαδοχικών τιμών που μεγιστοποιεί τα παραπάνω κριτήρια. Για τις μεταβλητές με διακριτές τιμές, υπάρχει εξ΄ορισμού ένα αποτέλεσμα από τον έλεγχο για κάθε διαφορετικής τιμής του A σε υποσύνολα, επιτρέποντας τον έλεγχο σε κάθε υποσύνολο.
Το αρχικό δέντρο που δημιουργείται στη συνέχεια υπόκειται σε μείωση/κλάδεμα (prune) για να αποφευχθεί η υπερπροσαρμογή. Το κλάδεμα γίνεται από τα «φύλλα» προς τη «ρίζα». Για ένα τμήμα του δέντρου ο αλγόριθμος βρίσκει το σταθμισμένο άθροισμα του εκτιμώμενου σφάλματος των κλαδιών και το συγκρίνει με το εκτιμώμενο σφάλμα που θα προέκυπτε αν το τμήμα αυτό το αντικαθιστούσε ένα φύλλο. Αν το φύλλο δεν είναι μεγαλύτερο απο το πρώτο τμήμα του δέντρου τότε το τμήμα αυτό «κλαδεύεται». Με ανάλογο τρόπο γίνεται ο έλεγχος αν ένα τμήμα του δέντρου μπορεί να αντικατασταθεί

με ένα κλαδί του τμήματος αυτού συκγρίνοντας το εκτιμώμενο σφάλμα. Με την προσπέλαση από όλο το δέντρο ολοκληρώνεται και η διαδικασία του κλαδέματος. Υπάρχουν δύο είδη κλαδέματος το pre-prune όπου σταματάει η ανάπτυξη του δέντρου όταν η πληροφορία θεωρείται αβάσιμη και το post-prune όταν το δέντρο έχει ολοκληρωθεί και έχει πάρει το τελικό του ύψος και έπειτα ξεκινάει το κλάδεμα.

2.4.2 Αλγόριθμος k-means

Ο αλγόριθμος k-means (k-μέσων) είναι μια επαναληπτική μέθοδος για το διαχωρισμό ενός συνόλου δεδομένων που ορίζεται από το χρήστη σε ένα αριθμό ομάδων k. Στην αρχή του αλγορίθμου επιλέγονται τα κεντροειδή σημεία τα οποία αποτελούν και τα αρχικά σημεία αναφοράς της κάθε ομάδας. Ο τρόπος με τον οποίο επιλέγονται αυτά τα σημεία μπορεί να είναι είτε με την τυχαία δειγματοληψία είτε με μικρή μεταβολή του ολικού μέσου των δεδομένων k φορές. Στη συνέχεια γίνεται η ακόλουθη επανάληψη μέχρι να υπάρξει σύκλιση. Κάθε σημείο αντιστοιχίζεται στο πλησιέστερο κεντροειδές σημείο. Αν η απόσταση από κάποιο κεντροειδές είναι ίση από κάποιο άλλο τότε διαλέγεται ένα τυχαία. Έτσι διαχωρίζονται τα δεδομένα σε k ομάδες. Μετά από αυτή τη διαδικασία τα κεντροειδή σημεία της κάθε ομάδας επανατοποθετούνται στο κέντρο των σημείων που την αποτελούν. Στη περίπτωση που τα σημεία έχουν σταθμικά βάρη τότε η επανατοποθέτηση γίνεται με τη χρήση του σταθμισμένου μέσου της ομάδας. Ο αλγόριθμος συγκλίνει όταν σταματούν να μεταβάλλονται οι αναθέσεις των δεδομένων και επομένως και των κεντροειδών. Για κάθε επανάληψη απαιτούνται k επί το σύνολο συγκρίσεις και είναι ο παράγοντας που καθορίζει την πολυπλοκότητα του αλγορίθμου. Ο αριθμός των επαναλήψεων που χρειάζεται ο αλγόριθμος ώστε να συγκλίνει για τα εκάστοτε δεδομένα είναι σχεδόν γραμμικός σε σχέση με το μέγεθος των δεδομένων.
Η Ευκλείδια απόσταση μπορεί να είναι το βασικό μέτρο κατά την ανάθεση του κάθε σημείου σε ένα σύνολο. Για ένα αρκετά μεγάλο d η Ευκλείδια απόσταση μεταξύ δύο τυχαίων σημείων $[x_1, x_2, \ldots, x_d]$ και $[y_1, y_2, \ldots, y_d]$ είναι ίση με $\sqrt{\sum_{i=1}^{d} (x_i - y_i)^2}$. Ένα σημαντικό ζήτημα με τον αλγόριθμο είναι η ευαισθησία που έχει στην αρχική επιλογή κεντροειδών και αυτό γιατί παγιδεύεται συχνά σε τοπικά ακρότατα.



Σχήμα 2.2 Μη ομαδοποιημένα δεδομένα (αριστερά) και k-means ομαδοποιημένα δεδομένα (δεξια) (πηγη: http://pypr.sourceforge.net/kmeans.html)

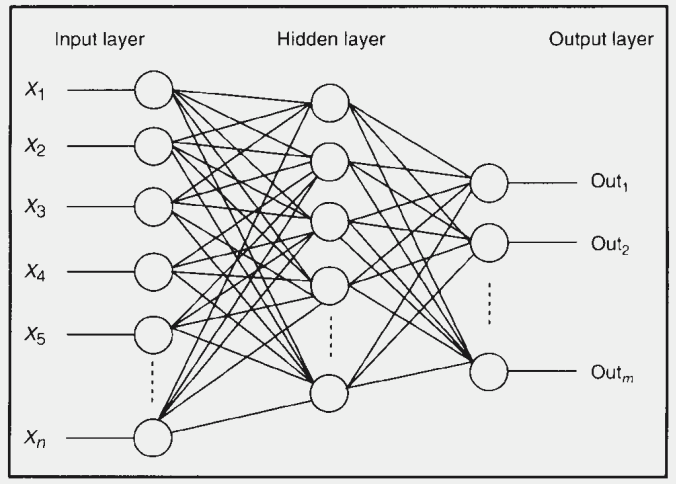
2.4.3 Νευρωνικό Δίκτυο - Neural Network

Τα νευρωνικά δίκτυα αποτελούν μια πολύ δυνατή τεχνική για την πρόβλεψη, ταξινόμηση και τμηματοποίηση δεδομένων. Τον τρόπο με τον οποίο οι άνθρωποι μαθαίνουν από τις εμπειρίες τους μιμούνται και τα νευρωνικά δίκτυα ενώ παράλληλα αποτελούν μία προσέγγιση ανάπτυξης και εκτίμησης μαθηματικών δομών με δυνατότητα μάθησης και στοχεύουν στην εξαγωγή προτύπων και τον προσδιορισμό τάσεων οι οποίες είναι πολύ πολύπλοκες για να προσδιοριστούν από ανθρώπους ή από άλλες υπολογιστικές τεχνικές. Ένα εκπαιδευμένο νευρωνικό δίκτυο μπορεί να κάνει έγκυρες προβλέψεις για τα νέα στιγμιότυπα του προβλήματος στο οποίο έχει εκπαιδευτεί.

Τα νευρωνικά δίκτυα χρησιμοποιούν ένα σύνολο από στοιχεία επεξεργασίας (κόμβους) ανάλογους με τους νευρώνες στο ανθρώπινο μυαλό. Όλοι οι κόμβοι ενώνονται σε ένα δίκτυο που μπορεί να καταλάβει τα πρότυπα μόλις αυτά παρουσιαστούν μέσα σε ένα σύνολο δεδομένων. Δηλαδή, το δίκτυο μπορεί να μαθαίνει από την εμπειρία όπως ακριβώς κάνουν και οι άνθρωποι. Το σημείο αυτό διακρίνει τα νευρωνικά δίκτυα από τα παραδοσιακά προγράμματα υπολογιστών, τα οποία απλά ακολουθούν οδηγίες σύμφωνα με μια καλά ορισμένη σειρά.

Η βασική μονάδα ενός νευρωνικού δικτύου είναι ο τεχνητός νευρώνας (κόμβος), το οποίο παίρνει ως είσοδο ένα διάνυσμα πραγματικών τιμών, υπολογίζει ένα γραμμικό

συνδυασμό των εισόδων και δίνει ως έξοδο 1 αν το αποτέλεσμα είναι μεγαλύτερο από κάποιο κατώφλι θ ή μηδέν διαφορετικά.
Τα νευρωνικά δίκτυα αποτελούνται από επιμέρους μονάδες που λειτουργούν παράλληλα (Σχήμα 3.1). Η συνάρτηση του δικτύου καθορίζεται ως επί το πλείστον από τις συνδέσεις μεταξύ των νευρώνων. Μπορούμε να εκπαιδεύσουμε το νευρωνικό ώστε να εκτελεί μια συγκεκριμένη συνάρτηση ρυθμίζοντας τα βάρη μεταξύ των συνδέσεων.



Σχήμα 2.3 Η δομή ενός νευρωνικού δικτύου (πηγή: http://mechanicalforex.com)

Μια συγκεκριμένη είσοδος οδηγείται σε μια συγκεκριμένη έξοδο και με βάση αυτή την αρχή εκπαιδεύονται τα νευρωνικά δίκτυα. Στη συνέχεια το νευρωνικό δίκτυο ρυθμίζεται βάσει μιας σύγκρισης της τρέχουσας εξόδου με την επιθυμητή έξοδο, μέχρι να ταιριάξουν.
Ο πιο δημοφιλής αλγόριθμος των νευρωνικών δικτύων είναι ο «back propagation». Τα νευρωνικά δίκτυα είναι πολύ ισχυρά εργαλεία, με πολύ ικανοποιητική απόδοση ακόμα και στα προβλήματα εξόρυξης δεδομένων. Επίσης, έχουν πολύ μεγάλη ανοχή σε ελλιπή δεδομένα ή δεδομένα με θόρυβο. Για τους παραπάνω λόγους χρησιμοποιούνται πολύ ακόμα και αν η εκπαίδευσή τους απαιτεί πολύ χρόνο και η ερμηνεία τους είναι δύσκολη.

2.5 Εξόρυξη στο διαδίκτυο – Web Mining

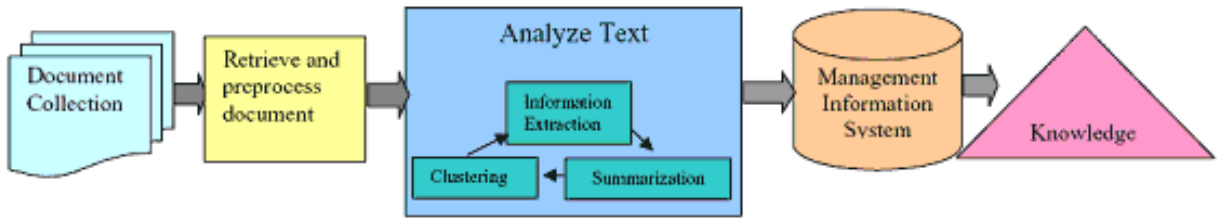
Εξόρυξη στο διαδικτύο (Web Mining) είναι μια απο τις τεχνικές ευφυούς υπολογισμού γύρω απο το πλαίσιο της διαχείρισης δεδομένων του διαδικτύου (Web Data Management). Σε γενικές γραμμές, ως εξόρυξη στο διαδίκτυο ορίζεται η εφαρμογή των

μεθόδων της εξόρυξης δεδομένων (data mining) για να γίνει η εξαγωγή χρήσιμων πληροφοριών από τα δεδομένα του διαδικτύου [10].
Η εξόρυξη διαδικτύου μπορεί να ταξινομηθεί σε τρεις κατηγορίες με βάση τους στόχους που γίνεται η εκάστοτε εξόρυξη και το μέρος του διαδικτύου που γίνεται. Οι κατηγορίες αυτές είναι η εξόρυξη περιεχομένου, δομής και χρήσης του διαδίκτυου (web content mining, web structure mining, web usage mining).
Η εξόρυξη περιεχομένου προσπαθεί να ανακαλύψει τη σημαντική πληροφορία από το διαδίκτυο και αυτό το επιτυγχάνει κυρίως από γραπτά κείμενα που αναρτώνται καθημερινά σε αυτό. Ο όρος εξόρυξης περιεχομένου από το διαδίκτυο καλείται ορισμένες φορές και ως εξόρυξη κειμένου (text mining).
Η εξόρυξη δομής [11] του διαδίκτυου περιλαμβάνει την μοντελοποίηση διαφόρων ιστοσελίδων σε συνδετικές δομές, όπου η κάθε σύνδεση βασίζεται στο σύνδεσμο (link) από τη μια σε σελίδα στην άλλη. Αυτή η συνδετική δομή μεταξύ διαφόρων ιστοσελίδων μπορεί να χρησιμοποιηθεί στη δημιουργία ομάδων, από σελίδες που έχουν τουλάχιστον ένα σύνδεσμο η μια στην άλλη. Αυτές οι ομάδες μπορούν να αποδείξουν ότι οι σελίδες που τις απαρτίζουν έχουν κάποιες ομοιότητες για την κάθε ομάδα αντίστοιχα.
Η εξόρυξη χρήσης του διαδίκτυου προσπαθεί να αποκαλύψει τα υποκείμενα πρότυπα πρόσβασης από συναλλαγές web ή από τις συνεδρίες των χρηστών στο διαδίκτυο. Ως συνεδρία (session) ορίζεται οι διάφορες κινήσεις και επισκέψεις σε διάφορες ιστοσελίδες ενός χρήστη από τη στιγμή που θα ανοίξει έναν browser μέχρι τη στιγμή που θα τον κλείσει.

ΚΕΦΑΛΑΙΟ 3 : Εξόρυξη Κειμένου - Text Mining

3.1.1 Εισαγωγή στην εξόρυξη κειμένου

Η ανακάλυψη γνώσης σε κείμενο (Knowledge Discovery in Text - KDT) καθώς και η εξόρυξη κειμένου (Text Mining – TM) περιλαμβάνουν αυτοματοποιημένες τεχνικές για την ανάλυση πολύ μεγάλων συλλογών από δεδομένα αλλά και την εξαγωγή χρήσιμων πληροφοριών από αυτά, οι οποίες βρίσκονται σήμερα στο επίκεντρο του ενδιαφέροντος τόσο από εμπορική όσο και από επιστημονική πλευρά. Χρησιμοποιώντας τεχνικές από την εξόρυξη δεδομένων (text mining), την μηχανική μάθηση (machine learning), τη στατιστική (statistics) την επεξεργασία φυσικής γλώσσας (natural language processing), την ανάκτηση πληροφορίας (information retrieval), την εξαγωγή πληροφορίας (information extraction) και τη διαχείριση γνώσης (knowledge management), οι τεχνικές αυτές προσπαθούν να επιλύσουν το πρόβλημα της μετατροπής των τεραστίων ποσοτήτων από δεδομένα, σε χρήσιμη γνώση. Καθώς δεν υπάρχει καθιερωμένο λεξιλόγιο για αυτό τον αναπτυσσόμενο ερευνητικό τομέα, συχνά απαντώνται διαφορετικοί όροι για να δηλώσουν το ίδιο πράγμα: Ανακάλυψη γνώσης σε κείμενο (Knowledge Discovery in Text), Κειμενική Εξόρυξη Δεδομένων (Text Data Mining), Εξόρυξη Κειμένου ή Εξόρυξη Γνώσης από Κείμενα (Text Mining).



Σχήμα 3.1 Διαδικασία εξόρυξης κειμένου (πηγή: gophoto.us)

Διαχωρίζοντας τον όρο knowledge discovery in text από τον όρο text mining, μπορούμε να πούμε ότι η εξόρυξη κειμένου αποτελεί ένα στάδιο της ανακάλυψης γνώσης σε κείμενο, η οποία είναι μια διαδικασία που περιλαμβάνει πολλά βήματα για την ανεύρεση χρήσιμης πληροφορίας από κείμενα, από την συλλογή των εγγράφων, την προ-επεξεργασία τους (ώστε να μετατραπούν σε κάποια επιθυμητή αναπαράσταση όπως XML, SGML κλπ), την εξαγωγή λεκτικών πληροφοριών σχετικών με το περιεχόμενο κάθε εγγράφου, την εξόρυξη κειμένου μέσω της δημιουργίας μεταδεδομένων (metadata creation) και της αναγνώρισης προτύπων και συσχετίσεων μεταξύ των δεδομένων, μέχρι και την απεικόνιση (οπτικοποίηση- visualization) της γνώσης που προκύπτει.

Η διαδικασία της εξόρυξης γνώσης από κείμενο (text mining) χρησιμοποιεί πολύ μεγάλα σύνολα κειμένων (γνωστά και ως corpora) που είναι αποθηκευμένα είτε στο διαδίκτυο είτε συμβατικά, και περιλαμβάνει την ανακάλυψη (discovery) προτύπων (patterns) ανάμεσα στα σύνολα δεδομένων (data sets) που περιλαμβάνονται στα κείμενα, που πριν δεν ήταν γνωστά, ισχύουν, είναι κατανοητά και πιθανώς χρήσιμα, καθώς και την ανάλυσή τους για να βρούμε μη αναμενόμενες συσχετίσεις ανάμεσα στα δεδομένα και να τα συνοψίσουμε με νέους τρόπους που είναι κατανοητοί και χρήσιμοι στους χρήστες.
Για να επεξηγήσουμε τον όρο «πρότυπο» που προαναφέρθηκε, μπορούμε να θεωρήσουμε τα δεδομένα μας ως ένα σύνολο γεγονότων F (π.χ. περιπτώσεις σε μια βάση δεδομένων). Το πρότυπο είναι ένας κανόνας E ο οποίος περιγράφει γεγονότα σε ένα υποσύνολο FE του F. Μπορεί να έχουμε είτε πρότυπα πρόβλεψης (predictive pattern), με σκοπό την πρόβλεψη ενός ή περισσότερων γνωρισμάτων (attributes) από αυτά που υπάρχουν στη βάση, είτε πρότυπα ενημέρωσης (informative pattern) τα οποία δεν επιλύουν κάποιο συγκεκριμένο πρόβλημα αλλά παρουσιάζουν στο χρήστη ενδιαφέροντα πρότυπα που θα έπρεπε να γνωρίζει.
Έτσι, το text mining εξετάζει μεγάλες συλλογές από έγγραφα (documents) μη δομημένων κειμένων προκειμένου να ανακαλύψει τη δομή καθώς και αυτονόητα «νοήματα» που κρύβονται μέσα στο κείμενο. Έτσι, όπως η εξόρυξη δεδομένων εντοπίζει συνδέσεις και συσχετίσεις που δεν ήταν προηγουμένως γνωστές ανάμεσα σε δομημένα δεδομένα, έτσι και η εξόρυξη δεδομένων βρίσκει συνδέσεις ανάμεσα σε κείμενα, τα οποία όμως αποτελούν μη δομημένα δεδομένα.

3.1.2 Στόχοι και Τεχνικές Εξόρυξης Κειμένου

Η εξόρυξη κειμένου στοχεύει στην εξαγωγή πληροφοριών από μεγάλο όγκο κειμένων οι οποίες μπορεί να φανούν χρήσιμες προς το χρήστη, μέσω της ανακάλυψης προτύπων ανάμεσα στις πληροφορίες και τα μεταδεδομένα που έχουν προκύψει από αυτά ύστερα από επεξεργασία των μη δομημένων δεδομένων τους.

Οι κυριότερες κατηγορίες των μεθόδων που χειρίζεται η εξόρυξη κειμένου είναι:

- Εξαγωγή χαρακτηριστικών γνωρισμάτων (Feature Extraction).
- Πλοήγηση με βάση το κείμενο (Text Based Navigation).
- Κατηγοριοποίηση, κατάταξη με επίβλεψη (Categorization, Supervised Classification).
- Ομαδοποίηση, μη επιβλεπόμενη κατάταξη (Clustering, Unsupervised Classification).
- Περιληπτική Παρουσίαση της Πληροφορίας (Summarization).

- Γλωσσικός προσδιορισμός (Language Identification) και απόδοση κειμένου στο συγγραφέα.
- Συσχετίσεις (Associations).
- Απεικόνιση – Οπτικοποίηση (Visualization).

3.1.2.1 Εξαγωγή χαρακτηριστικών γνωρισμάτων (Feature Extraction)

Έχει ως στόχο τον προσδιορισμό γεγονότων και σχέσεων στο κείμενο, διακρίνοντας συχνά εάν κάποια ονομαστική φράση είναι πρόσωπο, θέση, οργανισμός ή άλλο διακριτό αντικείμενο. Οι αλγόριθμοι εξαγωγής χαρακτηριστικών περιλαμβάνουν την εξαγωγή ονόματος (εντοπίζονται εμφανίσεις ονομάτων στο κείμενο και καθορίζεται σε ποιο τύπο οντότητας αναφέρεται το όνομα), την εξαγωγή όρου μιας περιοχής (προσδιορισμός τεχνικών όρων σε ένα κείμενο) αναγνώριση συντμήσεων (προσδιορίζονται συντμήσεις και αρκτικόλεξα και αντιστοιχούνται στην πλήρη μορφή τους). Φυσικά αυτό περιλαμβάνει έπειτα και την επιλογή σημαντικών όρων και απόρριψη άλλων μη σημαντικών, καθώς και τον υπολογισμό της συχνότητας εμφάνισης των όρων, ενώ οι όροι πρέπει να βρίσκονται σε κανονική ή καθιερωμένη μορφή (πχ χωρίς επιπλέον καταλήξεις λόγω κλίσης της λέξης). Η διαδικασία αυτή μπορεί να χρησιμοποιεί λεξικά για τον προσδιορισμό μερικών όρων καθώς και γλωσσικά υποδείγματα για την ανίχνευση άλλων.

3.1.2.2 Αναζήτηση και Ανάκτηση (Search and Retrieval)

Περιλαμβάνει την αναζήτηση σε εσωτερικές συλλογές εγγράφων ή σε συλλογές που βρίσκονται στον Παγκόσμιο Ιστό. Κύριο χαρακτηριστικό αποτελεί η δυνατότητα, αφού αρχικά συνταχθεί ένα ευρετήριο, να προσφέρεται ένα αρκετά ευρύ φάσμα επιλογών αναζήτησης κειμένου, στις οποίες συμπεριλαμβάνονται οι βασικές επιλογές αναζήτησης (όπως η Boolean, η index-based, η βασισμένη σε οντολογίες, ή στην αριθμητική σειρά, η τμηματική αναζήτηση,) αλλά και πιο σύνθετες επιλογές αναζήτησης (όπως relevancy, έρευνα φυσικής γλώσσας, η αναζήτηση έννοιας, η ασαφής αναζήτηση, κα).

3.1.2.3 Κατηγοριοποίηση, κατάταξη με επίβλεψη (Categorization, Supervised Classification)

Η κατηγοριοποίηση είναι η διαδικασία της κατάταξης εγγράφων σε προκαθορισμένες κατηγορίες. Μας βοηθάει λοιπόν στο να προσδιορίσουμε ποια είναι τα κύρια θέματα μιας συλλογής εγγράφων. Οι κατηγορίες είτε έχουν διαμορφωθεί εξαρχής από τον

προγραμματιστή είτε μπορούν να προσδιοριστούν από το χρήστη. Υπάρχουν δύο τρόποι για την κατηγοριοποίηση: ο πρώτος περιλαμβάνει τη δημιουργία ενός θησαυρού (thesaurus), δηλαδή ενός συνόλου που περιλαμβάνει όρους σχετικούς με το θέμα κάθε κατηγορίας καθώς και συσχετίσεις μεταξύ αυτών των όρων (πχ διευρυμένους όρους, κοντινότερους όρους, συνώνυμα, σχετικούς όρους) και τελικά τον ορισμό του θέματος του κειμένου με βάση τη συχνότητα των όρων σχετικών με το θέμα που υπάρχουν στο έγγραφο. Ο δεύτερος τρόπος περιλαμβάνει την εκπαίδευση (training) του εργαλείου κατηγοριοποίησης με κάποια δείγματα από τα έγγραφα, τη στατιστική ανάλυση λεκτικών προτύπων (linguistic patterns) όπως είναι οι λεξικολογικές συγγένειες, οι συχνότητες λέξεων των εγγράφων προς εκπαίδευση, το χωρισμό αυτών των προτύπων σε κατηγορίες (με στατιστικό τρόπο), και τέλος την ταξινόμηση των υπόλοιπων εγγράφων. Η δεύτερη προσέγγιση είναι προτιμότερη όταν έχουμε να κάνουμε με μεγάλους τομείς, καθώς τότε είναι αρκετά δύσκολο να δημιουργηθεί κάποιος θησαυρός εννοιών.

3.1.2.4 Ομαδοποίηση, μη επιβλεπόμενη κατάταξη (Clustering, Unsupervised Classification)

Μία ομάδα (cluster) είναι μια συλλογή από σχετικά έγγραφα, και η ομαδοποίηση (clustering) είναι η διαδικασία της δημιουργίας ομάδων εγγράφων βάσει κάποιου κριτηρίου ομοιότητας, αυτόματα χωρίς να έχουμε προσδιορίσει από πριν τις κατηγορίες. Η ομαδοποίηση κειμένων είναι χρήσιμη για τον προσδιορισμό κρυμμένων ομοιοτήτων, για να διευκολύνει τη διαδικασία του να βρούμε παρόμοιες ή σχετικές πληροφορίες, ενώ επιπλέον μπορούμε όταν εξερευνούμε μια καινούρια συλλογή δεδομένων να έχουμε μια γενική επισκόπηση της συλλογής. Οι πιο γνωστοί αλγόριθμοι που χρησιμοποιούνται είναι ιεραρχικοί (hierarchical), διαχωριστικοί (partitional), δυαδικοί σχεσιακοί (binary relational) και ασαφείς (fuzzy). Ο πιο σημαντικός παράγοντας στη λειτουργία της ομαδοποίησης είναι το μέτρο ομοιότητας που χρησιμοποιεί ο εκάστοτε αλγόριθμος , καθώς υπάρχουν διάφοροι τύποι μέτρων όπως η θεώρηση λέξεων οι οποίες εμφανίζονται συχνά μαζί ως κοινά χαρακτηριστικά, ενώ ένας άλλος τύπος μπορεί να περιλαμβάνει χαρακτηριστικά γνωρίσματα που έχουν εξαχθεί (πχ το όνομα ενός προσώπου).

3.1.2.5 Περιληπτική Παρουσίαση της Πληροφορίας (**Summarization**)

Αποτελεί την εξαγωγή της περίληψης ενός κειμένου, δηλαδή τη μείωση του μεγέθους του κειμένου διατηρώντας όμως τα βασικά στοιχεία του περιεχομένου του. Σε αυτή τη λειτουργία ο χρήστης έχει συνήθως τη δυνατότητα να καθορίσει διάφορες παραμέτρους, όπως το πλήθος των λέξεων που θα εξαχθούν ή το ποσοστό επί του συνολικού κειμένου που θα αποτελεί την περίληψη.

3.1.2.6 Γλωσσικός προσδιορισμός (Language Identification) και απόδοση κειμένου στο συγγραφέα

Ένα εργαλείο language identification μπορεί να προσδιορίσει σε ποια γλώσσα είναι γραμμένο ένα κείμενο, ή και τι ποσοστό του κειμένου είναι γραμμένο σε κάθε γλώσσα, εάν αυτό είναι γραμμένο σε περισσότερες. Επιπλέον, υπάρχει η δυνατότητα προσδιορισμού του συγγραφέα στον οποίο ανήκει το κείμενο, χρησιμοποιώντας τεχνικές data mining.

3.1.2.7 Συσχετίσεις (Associations)

Στην ανάλυση συσχετίσεων αναγνωρίζονται σχέσεις μεταξύ χαρακτηριστικών γνωρισμάτων που έχουν εξαχθεί από τη συλλογή εγγράφων, και ορίζεται ένα πρότυπο με τη χρήση μιας αντικειμενικής συσχέτισης. Το πρότυπο αυτό, εκφράζει έναν κανόνα που αναφέρει ότι αν βρεθεί η υπο-λέξη που περιέχεται στο πρότυπο, ακολουθούμενη από μία άλλη δεδομένη υπο-λέξη, σε συγκεκριμένη απόσταση μεταξύ τους, τότε η αντικειμενική συνθήκη θα διατηρηθεί με μεγάλη πιθανότητα. Οι κανόνες αυτοί είναι πολύ ευέλικτοι για την περιγραφή των τοπικών ομοιοτήτων που περιέχονται στα δεδομένα του κειμένου.

3.1.2.8 Απεικόνιση – Οπτικοποίηση (Visualization)

Το visualization χρησιμοποιεί την εξαγωγή χαρακτηριστικών γνωρισμάτων και το ευρετήριο βασικών όρων για να κατασκευάσει μια γραφική αναπαράσταση μιας συλλογής εγγράφων. Η προσέγγιση αυτή βοηθάει το χρήστη να αναγνωρίζει πολύ γρήγορα τα κύρια θέματα και τις βασικές έννοιες των κειμένων, με βάση τη σπουδαιότητα (π.χ. μέγεθος) αυτών στην αναπαράσταση.

3.1.3 Μέθοδοι text mining

Ενώ γενικά η εξόρυξη κειμένου περιέχει μεθόδους από διάφορα τεχνολογικά πεδία, θα μπορούσαμε να χωρίσουμε αυτές τις μεθόδους σε δύο κατηγορίες:
Η πρώτη κατηγορία περιλαμβάνει μεθόδους βασισμένες στην απόδοση. Ενδιαφέρει δηλαδή περισσότερο η αποτελεσματική συμπεριφορά του συστήματος και όχι απαραίτητα τα μέσα με τα οποία λαμβάνεται αυτή η συμπεριφορά. Στην κατηγορία αυτή περιλαμβάνονται διάφορες στατιστικές μέθοδοι (που στηρίζονται συνήθως σε ένα σαφές θεμελιώδες πρότυπο πιθανότητας) καθώς και τα νευρωνικά δίκτυα (neural networks: συστήματα στα οποία οι διασυνδέσεις διαμορφώνονται όπως οι νευρώνες του εγκεφάλου, και οι οποίες μπορούν να αλλάξουν δυναμικά).

Η δεύτερη κατηγορία περιλαμβάνει μεθόδους βασισμένες στη γνώση. Χρησιμοποιούν δηλαδή σαφείς αντιπροσωπεύσεις της γνώσης όπως οι έννοιες των λέξεων, οι σχέσεις μεταξύ των γεγονότων και των κανόνων για τα συμπεράσματα στις ιδιαίτερες περιοχές. Τέτοια συστήματα περιλαμβάνουν τους κανόνες διεξαγωγής συμπεράσματος, τις λογικές προτάσεις, τα σημασιολογικά δίκτυα (π.χ. ταξινομήσεις, οντολογίες), κανόνες ταιριάσματος των patterns κα.

Η επιλογή μεταξύ ενός στατιστικά προσανατολισμένου ή ενός βασισμένου στη γνώση εργαλείου εξαρτάται από την περιοχή στην οποία ενδιαφερόμαστε να κάνουμε εξόρυξη δεδομένων. Για παράδειγμα, για περιοχές που δεν αλλάζουν συχνά έννοιες και κανόνες, όπως είναι για παράδειγμα τα οικονομικά και η πολιτική, θα προτιμούσαμε κάποιον αλγόριθμο βασισμένο στη γνώση. Από την άλλη, σε μια περιοχή όπως η γενετική, η οποία αλλάζει συνεχώς έννοιες λόγω της ταχείας εξέλιξης του ερευνητικού αυτού τομέα, είναι προτιμότερο να χρησιμοποιηθεί κάποιο εργαλείο βασισμένο στην απόδοση.

3.1.4 Αναπαράσταση Κειμένου στην Εξόρυξη Κειμένου

Αφού μελετήσαμε τεχνικές και μεθόδους της εξόρυξης κειμένου, στη συνέχεια θα ασχοληθούμε με τον τρόπο αναπαράστασης κειμένου στη διαδικασία του text mining. Λόγω της συχνής έλλειψης κάποιας δομής στα αρχεία κειμένων, είναι προφανής η ανάγκη εύρεσης μια αναπαράστασης για την αντιπροσώπευση των στοιχείων-όρων των κειμένων, έτσι ώστε να είναι δυνατή η μετέπειτα επεξεργασία τους.

Όταν έχουμε μια συλλογή από αρχεία κειμένου, μπορούμε να θεωρήσουμε καθένα από αυτά ως ένα bag-of-words, μια «σακούλα» η οποία περιλαμβάνει όλες τις λέξεις που βρίσκονται στο κείμενο.

Ο πιο συνήθης τρόπος αναπαράστασης ενός κειμένου είναι η αναπαράσταση διανύσματος (vector representation), η οποία προέρχεται από τα συστήματα ανάκτησης πληροφορίας (information retrieval). Έτσι, κάθε text document από το σύνολο κειμένων που έχουμε είναι και ένα διάνυσμα όρων (term vector) στο οποίο κάθε όρος αποτελεί ένα μοναδικό ανεξάρτητο χαρακτηριστικό (feature). Κάθε στοιχείο σε αυτό το διάνυσμα έχει και μια τιμή η οποία αντιστοιχεί στην εμφάνιση του όρου μέσα στο κείμενο.

Με βάση αυτό μπορούμε να διακρίνουμε διάφορα μοντέλα διανυσματικής αναπαράστασης των κειμένων:

Στο λογικό μοντέλο (Boolean model), κάθε έγγραφο αναπαρίσταται από ένα σύνολο λογικών τιμών κάθε μία από τις οποίες δηλώνει εάν ένας συγκεκριμένος όρος

εμφανίζεται στο έγγραφο: συνήθως η τιμή 1 σημαίνει ότι εμφανίζεται και η τιμή 0 σημαίνει απουσία του συγκεκριμένου όρου από το κείμενο. Τα πλεονεκτήματα του λογικού μοντέλου είναι η ευκολία και η ταχύτητα λειτουργιών ερώτησης, αναζήτησης, κα, εφόσον χρησιμοποιούνται λογικές πράξης AND, OR, NOT κλπ, και η δυνατότητα χρησιμοποίησης της Boolean άλγεβρας στο Boolean model. Ωστόσο, το λογικό μοντέλο συνεπάγεται ότι η απάντηση στο κατά πόσον είναι σχετικό ένα κείμενο με ένα συγκεκριμένο όρο (και κατ' επέκταση θέμα) είναι μια δυαδική (binary) απόφαση, ενώ επιπλέον μία λογική τιμή για κάθε χαρακτηριστικό δεν μπορεί να αποδώσει κατά πόσο σημαντική είναι η παρουσία μίας λέξης σε ένα κείμενο, γεγονός το οποίο συχνά μπορεί να οδηγήσει σε λάθος συμπεράσματα.

Στο μοντέλο διανυσματικού χώρου (vector space model – VSM) τα αρχεία αναπαρίστανται ως διανύσματα σε ένα πολυδιάστατο Ευκλείδειο χώρο. Κάθε άξονας στο χώρο αντιστοιχεί σε ένα χαρακτηριστικό (attribute), δηλαδή σε έναν όρο/λέξη, με αποτέλεσμα η συντεταγμένη κάθε διανύσματος ως προς έναν άξονα να χαρακτηρίζει την εμφάνιση του όρου (στον οποίο αντιστοιχεί ο άξονας) στο συγκεκριμένο διάνυσμα-αρχείο κειμένου, και μάλιστα να αποτελεί ένα «βάρος» του όρου (term weight) ως προς το συγκεκριμένο κείμενο (πόσο σημαντικός θεωρείται δηλαδή ο όρος για το κείμενο). Τα βάρη που χρησιμοποιούνται για κάθε attribute είναι πραγματικές τιμές και μπορεί να είναι είτε απλά η συχνότητα εμφάνισης της λέξης (word frequency), είτε άλλες τιμές που θα μελετήσουμε ακολούθως. Τελικά, μια συλλογή εγγράφων αναπαρίσταται από ολόκληρο το διανυσματικό χώρο.

Ας δούμε τώρα εκτενέστερα τα βάρη (weights) που χρησιμοποιούνται για τις τιμές των συντεταγμένων (που αντιστοιχούν σε όρους) στο Vector Space Model. Θα θεωρήσουμε ότι έχουμε τη συντεταγμένη του αρχείου d που αντιστοιχεί στον άξονα του όρου t.

Καταρχάς, ορίζουμε τις ακόλουθες τιμές για τους όρους (terms) και τα αρχεία (documents):

- Term Frequency - $TF(d,t)$: Είναι η συχνότητα του όρου, πόσες φορές ($n(d,t)$) δηλαδή ο όρος t εμφανίζεται στο αρχείο d.
- Document Frequency - $DF(t)$: Εκφράζει πόσα κείμενα από τη συλλογή που έχουμε περιέχουν τον όρο t.

- D : είναι ο αριθμός των αρχείων που συγκροτούν τη συλλογή κειμένων που έχουμε (άρα και ο αριθμός των διανυσμάτων)
- Inverse Document Frequency - $IDF\ (t)$: εκφράζει την «σπανιότητα» (scarcity) του όρου μέσα στη συλλογή κειμένων. Έχει διάφορους τρόπους υπολογισμού,

από τους οποίους δύο είναι οι πιο συνήθεις $IDF(t) = \log\left(\frac{D}{DF(t)}\right)$ καθώς και $IDF(t) = \log\left(\frac{1+D}{DF(t)}\right)$
ή $IDF(t) = \log\left(\frac{D-DF(t)}{DF(t)}\right)$

Μπορούμε λοιπόν τώρα να διακρίνουμε τους διάφορους τρόπους απόδοσης βάρους w σε κάθε όρο (term weighting), και άρα υπολογισμού της τιμής της συντεταγμένης.

Ένας πρώτος τρόπος είναι η θεώρηση $w(d,t) = TF(d,t)$, έτσι ώστε κάθε διάνυσμα να είναι της μορφής $dtf = (tf1, tf2, tf3, \ldots, tfn)$. Η πιο απλή μορφή για το term frequency είναι τα

term counts, ο αριθμός δηλαδή εμφάνισης μιας λέξης σε κάθε κείμενο ($TF(d,t) = n(d,t)$).

Ωστόσο συνήθως υπόκειται σε κάποια κανονικοποίηση (length normalization) έτσι ώστε να μειώνεται ο θόρυβος που προκαλείται από το μέγεθος κειμένων τα οποία εκ των πραγμάτων θα εμφανίζουν περισσότερους όρους με μεγαλύτερη συχνότητα. Έτσι, υπάρχουν διάφοροι τρόποι υπολογισμού του term frequency όπως :

$$TF(d,t) = \frac{n(d,t)}{maxt\, n(d,t)} \text{ ή } TF(d,t) = 1 + logn(d,t)$$

Παρόλα' αυτά, δεν είναι όλοι οι όροι εξίσου σημαντικοί μέσα σε μια συλλογή κειμένων. Για παράδειγμα λέξεις που εμφανίζονται συνέχεια όπως άρθρα, αντωνυμίες κ.λ.π. θα έχουν πολύ μεγάλη συχνότητα και θα αποτελούν θόρυβο για την εξακρίβωση των σημαντικών όρων που καθορίζουν το περιεχόμενο ενός κειμένου. Για το λόγο αυτό, θεωρούμε ότι η σπανιότητα (scarcity) ενός όρου μέσα στη συλλογή κειμένων αποτελεί ένα μέτρο για τη σημαντικότητα του όρου. Θεωρούμε λοιπόν ότι η σημαντικότητα είναι αντιστρόφως ανάλογη της εμφάνισης του όρου, και εισάγουμε τον όρο του inverse document frequency στον υπολογισμό του βάρους: $w(d,t) = TF(d,t) * IDF(t)$.

Συνεπώς, όροι οι οποίοι εμφανίζονται σε πάρα πολλά αρχεία κειμένου λαμβάνουν μικρό βάρος, ενώ πιο σπάνιοι όροι οι οποίοι εμφανίζονται σε λίγα αρχεία λαμβάνουν μεγάλο βάρος, και θα μπορούσαμε να πούμε λοιπόν ότι περισσότερο ενδιαφέρον παρουσιάζουν όροι οι οποίοι ούτε είναι υπερβολικά συχνοί ούτε υπερβολικά σπάνιοι.

Ο υπολογισμός του term weighting με την προσέγγιση $TF - IDF$ είναι από τους πιο συνήθεις στον τομέα της εξόρυξης δεδομένων, και υπολογίζεται με διάφορους τρόπους,

σε σχέση με τους εκάστοτε τύπους που χρησιμοποιούνται για τα μέτρα TF και IDF όπως είδαμε και προηγουμένως.

Υπενθυμίζεται ωστόσο και πάλι η ανάγκη για κανονικοποίηση της τιμής του βάρους, καθώς τα κείμενα μεγάλου μήκους τείνουν να έχουν μεγαλύτερες συχνότητες λέξεων καθώς και περισσότερους όρους. Υπάρχουν διάφοροι τρόποι κανονικοποίησης ως προς το μήκος των αρχείων (document length normalization), όπως για παράδειγμα με πολλαπλασιασμό του term frequency με κάποιο άλλο όρο, ή κανονικοποίηση της απόστασης μεταξύ των διανυσμάτων, την οποία και θα μελετήσουμε στην επόμενη ενότητα.

Συνοπτικά, μπορούμε να πούμε ότι για τον υπολογισμό του vector space model στο οποίο αντιστοιχεί μια συλλογή αρχείων, θα πρέπει αρχικά να γίνει μια προ-επεξεργασία των κειμένων: να αναγνωρισθούν δηλαδή οι λέξεις από τις οποίες αποτελείται κάθε κείμενο (bag- of-words) και μάλιστα, για να βελτιστοποιηθεί η διαδικασία εύρεσης των σημαντικών όρων κάθε κειμένου, οι λέξεις θα πρέπει να υπόκεινται σε κάποια επεξεργασία όπως αφαίρεση πολύ κοινών λέξεων οι οποίες δεν έχουν νοηματική αξία (άρθρα αντωνυμίες, κ.λ.π.), η εύρεση λέξεων αντιστοιχούν στο ίδιο θέμα αλλά έχουν διαφορετική μορφή (π.χ. παράγωγα της ίδιας λέξης, και η εύρεση τελικά όρων που είναι οι πιο αντιπροσωπευτικοί για κάθε κείμενο ξεχωριστά. Τη διαδικασία αυτή της προ-επεξεργασίας θα την εξετάσουμε αναλυτικά σε επόμενη ενότητα. Μετά από αυτό το βήμα (document indexing) προχωράμε στο βήμα της ανάθεσης βάρους σε κάθε όρο για κάθε κείμενο (term weighting) σε όλη τη συλλογή που έχουμε, ώστε κάθε βάρος να υποδηλώνει πόσο σημαντικός θεωρείται ο εκάστοτε όρος για το αντίστοιχο κείμενο.

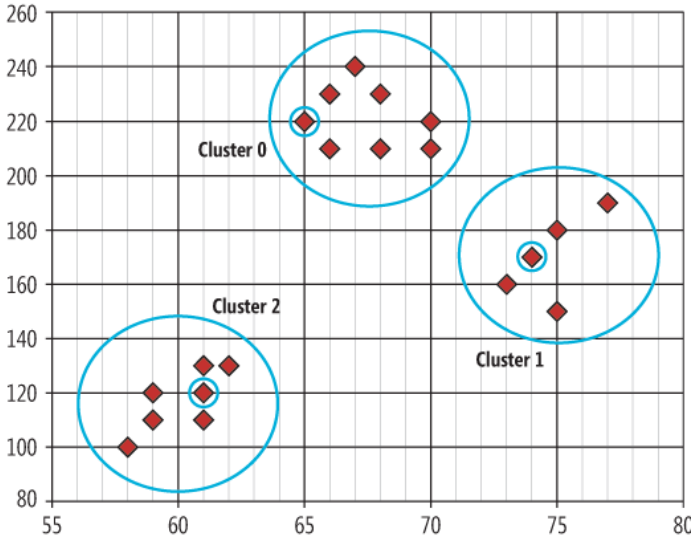
Τέλος, αναφέρονται ως μειονεκτήματα της μεθόδου του Vector Space Model το γεγονός ότι είναι αρκετά αργό ως προς το χρόνο επεξεργασίας του λόγω της πληθώρας υπολογισμών που απαιτούνται, δεν εξυπηρετεί ιδιαίτερα την ενημέρωση αλλαγών στα κείμενα εφόσον για κάθε όρο προστίθεται ένας επιπλέον άξονας και πρέπει να γίνουν υπολογισμοί της συντεταγμένης για όλα τα διανύσματα στο χώρο, ενώ τέλος η πολυδιάστατη μορφή του απαιτεί κόστος μνήμης και χαμηλή ταχύτητα σε υπολογισμούς. Ως πιθανές λύσεις προτείνονται η χρήση συνόλων λέξεων-κλειδιά (keyword-sets) για την αναπαράσταση ενός αρχείου, η οποία θα μειώνει το πλήθος των διαστάσεων, καθώς και η χρήση n-άδων λέξεων (τα λεγόμενα n- grams) δηλαδή ακολουθιών από n λέξεις (πχ το World Wide Web είναι ένα 3-gram) οι οποίες θα μπορούσαν να παρέχουν περισσότερη νοηματική πληροφορία για τα κείμενα από όσο μπορούν οι λέξεις μόνες τους.

3.2 Ομαδοποίηση Κειμένων

Θα παρουσιάσουμε τώρα μία από τις τεχνικές του text mining, η οποία αποτέλεσε και αντικείμενο της παρούσας εργασίας, την τεχνική της ομαδοποίησης (clustering).

3.2.1 Η έννοια της ομαδοποίησης

Η κατάταξη των αντικειμένων σε κλάσεις με βάση ομοιότητες που παρουσιάζουν αποτελεί μια πολύ συχνή πρακτική σε πολλούς τομείς, γι' αυτό και αποτελεί ένα αρκετά μεγάλο επιστημονικό πεδίο.
Ομαδοποίηση (clustering) ή Συσταδοποίηση ή Ανάλυση Ομαδοποίησης (Cluster Analysis) όπως αλλιώς ονομάζεται η εύρεση ομάδων αντικειμένων έτσι ώστε τα αντικείμενα σε κάθε ομάδα να είναι όμοια μεταξύ τους (ή να σχετίζονται) και διαφορετικά (ή μη σχετιζόμενα) από τα αντικείμενα των άλλων ομάδων.
Παραδείγματα στα οποία απαντάται η διαδικασία της ομαδοποίησης υπάρχουν σε κάθε τομέα: εύρεση ομάδων πελατών μιας εταιρείας που εμφανίζουν παρόμοια συμπεριφορά, ομαδοποίηση γονιδίων που έχουν την ίδια λειτουργία στον τομέα της γενετικής, ομαδοποίηση μετοχών που παρουσιάζουν παρόμοια διακύμανση τιμών, ομαδοποίηση weblog για εύρεση παρόμοιων προτύπων προσπέλασης, ομαδοποίηση σχετιζόμενων αρχείων για browsing, ομαδοποίηση κειμένων, ομαδοποίηση ασθενειών με βάση τα χαρακτηριστικά τους, κα.
Εάν θεωρήσουμε ότι τα αντικείμενα αναπαρίστανται από στοιχεία στο χώρο, μπορούμε να θεωρήσουμε ως πυκνότητα (density) μιας ομάδας αντικειμένων εκείνη την ιδιότητα που καθορίζει την ομάδα ως ένα σχετικά παχύ σμήνος από σημεία σε ένα χώρο, συγκρινόμενη με άλλες περιοχές του χώρου που μπορεί να έχουν λιγότερα αν όχι και καθόλου σημεία (Σχήμα 3.2).
Έτσι, μπορούμε να δώσουμε τον ορισμό που έδωσε Everrit [ref] για την ομάδα: Ομάδα (cluster) είναι μια συνεχής περιοχή του χώρου που περιέχει μια σχετικά υψηλή πυκνότητα από σημεία και χωρίζεται από άλλες περιοχές με σχετικά μεγάλη πυκνότητα σημείων, με περιοχές που έχουν χαμηλή πυκνότητα σημείων.



Σχήμα 3.2 Ομαδοποίηση των δεδομένων (πηγή: Microsoft.com)

3.2.2 Η διαδικασία του clustering

Μπορούμε λοιπόν να ορίσουμε την ομαδοποίηση (clustering) ως την οργάνωση μιας συλλογής από δείγματα-στοιχεία (patterns) σε ομάδες (clusters) με βάση κάποιο μέτρο ομοιότητας. Τα στοιχεία συνήθως τα περιγράφουμε με τη χρήση διανυσμάτων τιμών ή κάποιων μέτρων, ενώ επίσης μπορούμε και να τα αναπαραστήσουμε ως σημεία σε έναν πολυδιάστατο χώρο.

Στοιχεία τα οποία ανήκουν στην ίδια ομάδα παρουσιάζουν μεγαλύτερη ομοιότητα από ότι στοιχεία που ανήκουν σε διαφορετικές ομάδες. Καθώς η ομαδοποίηση σήμερα αποτελεί σημαντικό ερευνητικό πεδίο, έχει αναπτυχθεί μια μεγάλη γκάμα από τεχνικές για την αναπαράσταση των δεδομένων, έκφρασης της ομοιότητας μεταξύ στοιχείων και ομαδοποίησης των δεδομένων, με αποτέλεσμα να υπάρχει πληθώρα μεθόδων ομαδοποίησης.

Σημειώνουμε ότι τα κριτήρια ομοιότητας (similarity) που μπορούν να χρησιμοποιηθούν για να εξακριβωθεί κατά πόσον κάποια αντικείμενα έχουν αρκετά κοινά γνωρίσματα ώστε να θεωρούνται μέλη της ίδιας ομάδας, θα διαφέρουν ανάλογα με τα είδη των γνωρισμάτων των αντικειμένων, που αναφέρθηκαν στην προηγούμενη ενότητα.

Τα βήματα που ακολουθούνται συνήθως στη διαδικασία του clustering είναι τα ακόλουθα:

1) Pattern representation: αναπαράσταση των στοιχείων, που μπορεί να συνδυάζεται με την επιλογή μέρους των χαρακτηριστικών των στοιχείων ή και την παραγωγή νέων χαρακτηριστικών.

2) Similarity measure definition: ορισμός του μέτρου ομοιότητας μεταξύ των στοιχείων.

3) Clustering: η καθεαυτή διαδικασία της ομαδοποίησης, με εφαρμογή κάποιου αλγορίθμου ομαδοποίησης.

4) Data abstraction: αφαίρεση δεδομένων όταν χρειάζεται.

5) Assessment of output: προσδιορισμός και εκτίμηση του αποτελέσματος.

3.2.2.1 Ορισμοί και Συμβολισμοί

Στοιχείο (pattern) ή διάνυσμα χαρακτηριστικών x είναι ένα απλό δεδομένο το οποίο υπόκειται σε επεξεργασία από τον αλγόριθμο ομαδοποίησης. Αποτελείται από έναν αριθμό d χαρακτηριστικών και συμβολίζεται με $x = (x1, x2, \ldots, xd)$.

Χαρακτηριστικό ή γνώρισμα (feature, attribute) καλείται κάθε μέρος xi του στοιχείου x.

Η διάσταση (dimensionality) του κάθε στοιχείου καθώς και του χώρου των δεδομένων είναι ο αριθμός d των χαρακτηριστικών.

Ένα σύνολο από στοιχεία (pattern set) ορίζεται ως $X = \{ x1, x2, \ldots, xd \}$.

Η κλάση (class) αποτελεί μια ομάδα στοιχείων με κοινά ή όμοια χαρακτηριστικά. Οι αλγόριθμοι clustering προσπαθούν να δημιουργήσουν σύνολα στοιχείων τα οποία λογικά αναπαριστούν κλάσεις.

Το μέτρο της απόστασης (distance measure είναι ορισμένο στο χώρο των χαρακτηριστικών στοιχείων και φανερώνει το πόσο όμοια ή διαφορετικά είναι δύο στοιχεία μεταξύ τους.

3.2.2.2 Αναπαράσταση των στοιχείων, εισαγωγή και εξαγωγή χαρακτηριστικών

Η αναπαράσταση των στοιχείων αφορά στον αριθμό των κλάσεων, τον αριθμό των διαθέσιμων στοιχείων, στον αριθμό και τύπο των χαρακτηριστικών τα οποία ενδιαφέρουν τον clustering αλγόριθμο. Σε αυτή τη φάση επιλέγονται τα χαρακτηριστικά των στοιχείων που θεωρούνται ως καταλληλότερα για να χρησιμοποιηθούν στη διαδικασία της ομαδοποίησης, ενώ επίσης δημιουργούνται άλλα που μπορεί να κρίνεται ότι είναι πιο ενδιαφέροντα.

Τα γνωρίσματα μπορούν να διαχωριστούν στις ακόλουθες κατηγορίες:

Nominal: Οι τιμές είναι απλώς διαφορετικά ονόματα (αναγνωριστικά) με αρκετή πληροφορία ώστε να γίνει διάκριση ανάμεσά τους (=, ≠) (συμπεριλαμβάνονται και οι δυαδικές μεταβλητές 0-1). Παράδειγμα: ταχυδρομικός κώδικας, χρώμα ματιών, φύλο.

Διάταξης-Cardinal: Οι τιμές περιέχουν πληροφορία διάταξης (<,>). Παράδειγμα: Ποιότητα υλικού (καλή, πιο καλή, άριστη), αριθμοί στις διευθύνσεις.

Διαστήματος-Interval: Έχει σημασία η διαφορά μεταξύ δύο τιμών, υπάρχει μονάδα μέτρησης (+,-). Παράδειγμα: Θερμοκρασία σε Celsius ή Fahrenheit.

Ratio: Έχει σημασία και ο λόγος μεταξύ δύο τιμών (*, /). Παράδειγμα: Νομισματικές ποσότητες, ηλικία, θερμοκρασία σε Kelvin, ηλικία, μήκος.

Αφού εξαχθούν, παραχθούν και επιλεγούν τα καταλληλότερα χαρακτηριστικά, πραγματοποιείται η βέλτιστη αναπαράσταση για τα στοιχεία που θα επεξεργαστεί ο αλγόριθμος της ομαδοποίησης.

3.2.2.3 Ορισμός μέτρου ομοιότητας

Όπως έχει προαναφερθεί, το μέτρο ομοιότητας μεταξύ των στοιχείων καθορίζεται από μια συνάρτηση απόστασης όπως είναι για παράδειγμα η ευκλείδεια απόσταση.

3.2.2.4 Ομαδοποίηση

Η διαδικασία του clustering μπορεί να πραγματοποιηθεί με διάφορους τρόπους, και να έχει ένα απόλυτα καθορισμένο αποτέλεσμα (ξένες μεταξύ τους κλάσεις) είτε ασαφή (fuzzy), στο οποίο κάποια στοιχεία μπορεί να ανήκουν σε περισσότερες από μία κλάσεις. Υπάρχουν διάφορες τεχνικές clustering, τις οποίες θα μελετήσουμε στην επόμενη ενότητα.

3.2.2.5 Αφαίρεση δεδομένων

Κατά την αφαίρεση δεδομένων, το σύνολο των δεδομένων αποκτά μια απλή αναπαράσταση, τέτοια ώστε οι κλάσεις να είναι καθορισμένες με τρόπο σαφή και κατανοητό για την επεξεργασία των αποτελεσμάτων και την εξαγωγή συμπερασμάτων από τους χρήστες, όσο και για να είναι δυνατή η μετέπειτα αυτοματοποιημένη επεξεργασία των δεδομένων. Συνήθως στην αφαίρεση δεδομένων στο clustering κάθε κλάση αναπαρίσταται συνοπτικά με τη βοήθεια του κεντρικού στοιχείου (centroid) το οποίο χρησιμοποιείται ως αντιπρόσωπο στοιχείο της κλάσης.

3.2.2.6 Προσδιορισμός και εκτίμηση του αποτελέσματος

Στο τέλος του clustering γίνεται αξιολόγηση της διαδικασίας που ακολουθήθηκε και εκτίμηση του αποτελέσματος, ώστε να διευκρινιστεί κατά πόσον οι κλάσεις που δημιουργήθηκαν έχουν νόημα ή η ομαδοποίηση έγινε με τυχαίο τρόπο.

3.2.3 Αλγόριθμοι ομαδοποίησης

Οι τεχνικές ομαδοποίησης μπορούν να χωριστούν σε δύο κύριες κατηγορίες:

• στη Διαχωριστική Ομαδοποίηση (Partitional Clustering), στην οποία πραγματοποιείται ένας διαμερισμός των αντικειμένων σε μη επικαλυπτόμενα (non- overlapping) υποσύνολα (clusters) τέτοιος ώστε κάθε αντικείμενο να ανήκει σε ένα ακριβώς υποσύνολο, και

• στην Ιεραρχική Ομαδοποίηση (Hierarchical Clustering) στην οποία δημιουργούμε ένα σύνολο από εμφωλευμένα (nested) clusters, επιτρέποντας έτσι μια ομάδα να έχει υπο-ομάδες οργανωμένες σε ένα ιεραρχικό δέντρο.

Κάθε ιεραρχικός αλγόριθμος δημιουργεί μια ακολουθία από διαμερίσεις τμημάτων με μία μοναδική ομάδα στην κορυφή της δενδρικής ακολουθίας. Κάθε επίπεδο δημιουργείται από τη συγχώνευση δύο ομάδων του κατώτερου επιπέδου (από κάτω προς τα πάνω) ή την διαίρεση μιας μεγαλύτερης ομάδας σε μικρότερες (από πάνω προς τα κάτω). Η ομαδοποίηση των αντικειμένων πραγματοποιείται χρησιμοποιώντας ήδη υπάρχουσες ομάδες, σε πολυπλοκότητα τετραγωνικού χρόνου. Οι ιεραρχικοί αλγόριθμοι μπορούν να εφαρμόζονται χωρίς περιορισμό σε οποιοδήποτε είδος δεδομένων και είναι κατάλληλοι για μεγάλο όγκο δεδομένων. Παράγουν ομάδες με υψηλή ποιότητα και υπάρχει μεγάλη ανομοιογένεια μεταξύ των παραγόμενων ομάδων, ενώ ο χρήστης έχει τη δυνατότητα να αποφασίσει σε ποιο σημείο θα κόψει την

παραγωγή του δένδρου. Οι ιεραρχικοί αλγόριθμοι δεν μπορούν να χειριστούν δεδομένα με πολύ θόρυβο επειδή οι αποφάσεις για τη συγχώνευση δύο ομάδων είναι τελικές (δεν υπάρχει επιστροφή σε προηγούμενη κατάσταση). Οι ιεραρχικοί αλγόριθμοι χωρίζονται στους ιεραρχικά συσσωρευτικούς (hierarchical agglomerative) και στους ιεραρχικά διαιρετικούς (hierarchical divisive) αλγορίθμους.

Μερικοί γνωστοί ιεραρχικοί αλγόριθμοι:

- Κοντινότερος γείτονας (nearest neighbor)
- Ο απώτατος γείτονας (farthest neighbor)
- Το ελάχιστο συνδετικό δέντρο (minimum spanning tree)

Οι διαχωριστικοί αλγόριθμοι, σε αντίθεση με τους ιεραρχικούς αλγορίθμους, διαμερίζουν τα δεδομένα μόνο σε ένα σημείο. Έτσι εάν πρέπει να δημιουργηθούν Κ ομάδες με αντικείμενα ο αλγόριθμος κατάτμησης παράγει αυτά τα αντικείμενα αμέσως. Τα αντικείμενα αποδίδονται αυτόματα σε ομάδες με πολυπλοκότητα γραμμικού χρόνου. Οι διαχωριστικοί αλγόριθμοι εφαρμόζονται κυρίως σε δεδομένα που έχουν την έννοια της διαχώρισης, και λόγω της επαναληπτικής τους εκτέλεσης πολλές αποφάσεις που παίρνονται για τα δεδομένα μπορούν να ανακληθούν εάν κριθεί απαραίτητο. Η ποιότητα των παραγόμενων ομάδων εξαρτάται από το αρχικό σύνολο των αντικειμένων και συνήθως οι ομάδες βρίσκονται κοντά ως προς την ομοιότητά τους. Ο αριθμός των ομάδων είναι προκαθορισμένος από την αρχή, ενώ τέλος αντιμετωπίζουν το πρόβλημα του τοπικού ελαχίστου.

Μερικοί διαχωριστικοί αλγόριθμοι:

- Διανυσματικοί μηχανισμοί υποστήριξης (Support Vector Machines).
- Νευρωνικό Δίκτυο (Neural Network).
- K-means αλγόριθμος
- Κατηγοριοποίηση με τη μέθοδο Naïve Bayes (Naïve Bayes classifier).

3.2.4 Εξόρυξη κειμένου (Text mining) και ομαδοποίηση

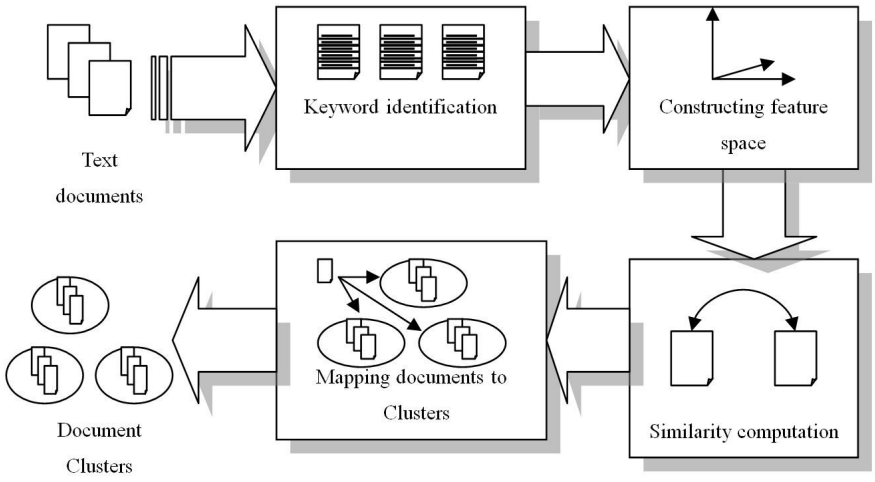
Επιστρέφοντας στην εξόρυξη κειμένου, υπενθυμίζουμε ότι η ομαδοποίηση κειμένων (clustering) αποτελεί μία από τις τεχνικές text mining, με την οποία δημιουργούμε ομάδες εγγράφων βάσει κάποιου κριτηρίου ομοιότητας, αυτόματα χωρίς να έχουμε προσδιορίσει από πριν τις κατηγορίες.

Χρησιμοποιώντας το vector space model, κάθε αρχείο αντιστοιχεί σε ένα διάνυσμα με συντεταγμένες τα βάρη που αντιστοιχούν σε κάθε όρο (άξονα στο χώρο) που εμφανίζεται στη συλλογή αρχείων που έχουμε. Αντιλαμβανόμαστε λοιπόν ότι η διαδικασία ομαδοποίησης των αρχείων κειμένου αντιστοιχίζεται απόλυτα στις μεθόδους ομαδοποίησης που αναφέρθηκαν στην προηγούμενη ενότητα.

Συγκεκριμένα, διακρίνουμε δύο προσεγγίσεις ομαδοποίησης αρχείων κειμένου: την ομαδοποίηση που βασίζεται σε λέξεις κλειδιά (keyword-based clustering) και την ομαδοποίηση που βασίζεται σε αρχεία (document based clustering). Οι δύο προσεγγίσεις διαφέρουν ως προς τα χαρακτηριστικά με βάση τα οποία ομαδοποιούνται τα αρχεία.

Οι αλγόριθμοι document-based clustering εφαρμόζονται κυρίως στο document vector space model στο οποίο κάθε στοιχείο παρουσιάζει το βάρος του όρου (term weighting) στο αντίστοιχο αρχείο. Έτσι, ένα αρχείο τοποθετείται σε ένα σημείο δεδομένων (data point) σε έναν ιδιαίτερα πολύ-διάστατο χώρο στον οποίο κάθε όρος είναι και ένας άξονας. Σε αυτό το χώρο η απόσταση μεταξύ των σημείων μπορεί να υπολογιστεί και να συγκριθεί. Σημεία δεδομένων τα οποία βρίσκονται κοντά μεταξύ τους μπορούν να συγχωνευθούν και να ομαδοποιηθούν στην ίδια ομάδα, ενώ στοιχεία σε μεγάλη απόσταση μεταξύ τους απομονώνονται σε διαφορετικές ομάδες. Συνεπώς τα αντίστοιχα αρχεία ομαδοποιούνται και χωρίζονται. Καθώς το document-based clustering βασίζεται στην «απόσταση μεταξύ των αρχείων» (document distance), είναι ιδιαίτερα σημαντικό να τοποθετούνται τα αρχεία στον κατάλληλο χώρο και να εφαρμόζονται σε αυτά οι κατάλληλες μέθοδοι υπολογισμού απόστασης.

Οι αλγόριθμοι keyword-based clustering επιλέγουν μόνο συγκεκριμένα γνωρίσματα (features) και βασισμένοι σε αυτό το σχετικά περιορισμένο πλήθος γνωρισμάτων δημιουργούν τα clusters. Αυτά τα συγκεκριμένα γνωρίσματα επιλέγονται επειδή θεωρούνται ως τα ουσιώδη γνωρίσματα μεταξύ των αρχείων, τα οποία απαντώνται σε παρόμοια αρχεία και είναι σπάνια σε ανόμοια αρχεία. Συνεπώς, για έναν keyword-based clustering αλγόριθμο είναι ιδιαίτερα σημαντικό το βήμα της επιλογής των πιο σημαντικών γνωρισμάτων.



Σχήμα 3.3 Τα βήματα της διαδικασίας ομαδοποίησης αρχείων (πηγή: www.intechopen.com)

Ας δούμε λοιπόν τα βήματα τα οποία περιλαμβάνει η διαδικασία της ομαδοποίησης αρχείων κειμένου:

Εξαγωγή Γνωρισμάτων: Το πρώτο βήμα είναι η εξαγωγή των γνωρισμάτων (feature extraction). Παίρνεται ως είσοδος το αρχικό σύνολο με τα κείμενα, και υποβάλλονται όλα αυτά τα ακατέργαστα κείμενα σε επεξεργασία (γι' αυτό και τη διαδικασία αυτή την ονομάζουμε και προ-επεξεργασία των αρχείων – preprocessing) προκειμένου να αναλυθούν και να επιλεγούν τα σχετικά χαρακτηριστικά που μπορεί να περιγράφουν αυτά τα αρχεία. Η έξοδος της διαδικασίας εξαγωγής γνωρισμάτων είναι συνήθως ένας πίνακας όρων-εγγράφων, στον οποίο κάθε στήλη αντιστοιχεί σε ένα έγγραφο και κάθε γραμμή δηλώνει ένα χαρακτηριστικό [12].

Η ποιότητα της εξαγωγής γνωρισμάτων έχει μεγάλη επίπτωση στην αποτελεσματικότητα των μετέπειτα clustering αλγορίθμων. Τα παρόμοια έγγραφα θα πρέπει να βρίσκονται κοντά στον χώρο γνωρισμάτων και τα ανόμοια έγγραφα θα πρέπει να βρίσκονται μακριά μεταξύ τους. Εάν έχουν απορριφθεί χρήσιμα και σημαντικά γνωρίσματα και έχουν συμπεριληφθεί άσχετα γνωρίσματα, υπάρχει αταξία στην απόσταση μεταξύ αρχείων, και όσο καλοί κι αν είναι οι αλγόριθμοι ομαδοποίησης, δε μπορούν να ομαδοποιήσουν τα αρχεία με λανθασμένη απόσταση. Τα βασικά στοιχεία που συμπεριλαμβάνονται στην εξαγωγή γνωρισμάτων είναι η αφαίρεση των stop-words, το stemming, η απόδοση βάρους στους όρους για κάθε αρχείο (term weighting), η εξαγωγή των σημαντικών γνωρισμάτων (key feature extraction) και η δημιουργία του πίνακα. Θα δούμε σε επόμενη ενότητα αναλυτικά τις διαδικασίες αυτές

που λαμβάνουν μέρος στην προ-επεξεργασία των αρχείων κειμένου πριν την ομαδοποίησή τους.

Όμαδοποίηση αρχείων: Το δεύτερο βήμα είναι η ομαδοποίηση των αρχείων κειμένου, στην οποία εφαρμόζονται αλγόριθμοι ομαδοποίησης προκειμένου να πάρουμε μια απεικόνιση των ομάδων (ένα clusters map), μια αναπαράσταση δηλαδή των ομάδων που έχουν δημιουργηθεί. Το clusters map όχι μόνο μπορεί να δείξει σε ποια ομάδα ανήκει ένα αρχείο, αλλά μπορεί επίσης να περιγράφει τη σχέση μεταξύ των clusters. Η είσοδος σε αυτό το βήμα είναι ο πίνακας όρων-εγγραφών. Με αυτό τον πίνακα, ένα αρχείο αναπαρίσταται ως ένα data point στον πολυδιάστατο χώρο.

Συνήθως το βήμα της ομαδοποίησης βασίζεται σε στατιστικούς ή μαθηματικούς υπολογισμούς χωρίς επαγγελματική γνώση. Κάθε όρος (γνώρισμα) χάνει το νόημά του και αντιπροσωπεύει απλά μια διάσταση στο χώρο.
Post Processing: Για διαφορετικές εφαρμογές υπάρχουν διαφορετικοί τρόποι να γίνει η μετέπειτα επεξεργασία. Μια κοινή μέθοδος περιλαμβάνει την επιλογή ενός κατάλληλου ορίου (threshold) για την παραγωγή του τελικού αποτελέσματος ομαδοποίησης. Μετά την ομαδοποίηση αρχείων παίρνουμε ένα βασικό cluster map στο οποίο οι ομάδες είναι οργανωμένες είτε σαν ένα δέντρο είτε με επίπεδο τρόπο. Έτσι, οι αλγόριθμοι της μετέπειτα επεξεργασίας (post processing algorithms) εφαρμόζονται προκειμένου να βρεθεί η σωστή σχέση μεταξύ των clusters.

ΚΕΦΑΛΑΙΟ 4: ΤΟ ΠΡΟΒΛΗΜΑ

Στα προηγούμενα κεφάλαια αναλύθηκε λεπτομερώς οι έννοιες και οι κλάδοι στην Επιστήμη των Υπολογιστών που αφορούν αυτή την εργασία. Τα άρθρα που αφορούν τα νέα για τις αγορές έχει αποδειχθεί ότι επηρεάζουν το χρηματιστήριο και τις αγορές [13] [14]. Η ανάλυση των νέων σύμφωνα με τις προηγούμενες μεθόδους μπορεί να αποκαλύψει ένα τρόπο πρόβλεψης της κίνησης της αγοράς κάτι το οποίο είναι δύσκολο πολύ αλλά πιο εύκολα μπορεί να δείξει την κίνηση μιας μετοχής μια εταιρείας ή ενός δείκτη που αφορά το συγκεκριμένο άρθρο που αναλύεται. Η εργασία αυτή αφορά την κατασκευή μιας αυτοματοποιημένης διαδικασίας, η οποία αντλεί επιτυχώς την ογκώδη πληροφορία από το διαδίκτυο και προσπαθεί να κατασκευάσει ένα σύνολο θετικών και αρνητικών λέξεων που σαν στόχο έχουν να αναλύουν έπειτα τα άρθρα που γράφονται.
Το κύριο πρόβλημα που προσπαθούμε να επιλύσουμε είναι αν τελικά μπορεί να δημιουργηθεί μια ομάδα από λέξεις η οποία μπορεί να χρησιμοποιηθεί στο κομμάτι της εξόρυξης κειμένου που αφορούν οικονομικά νέα.

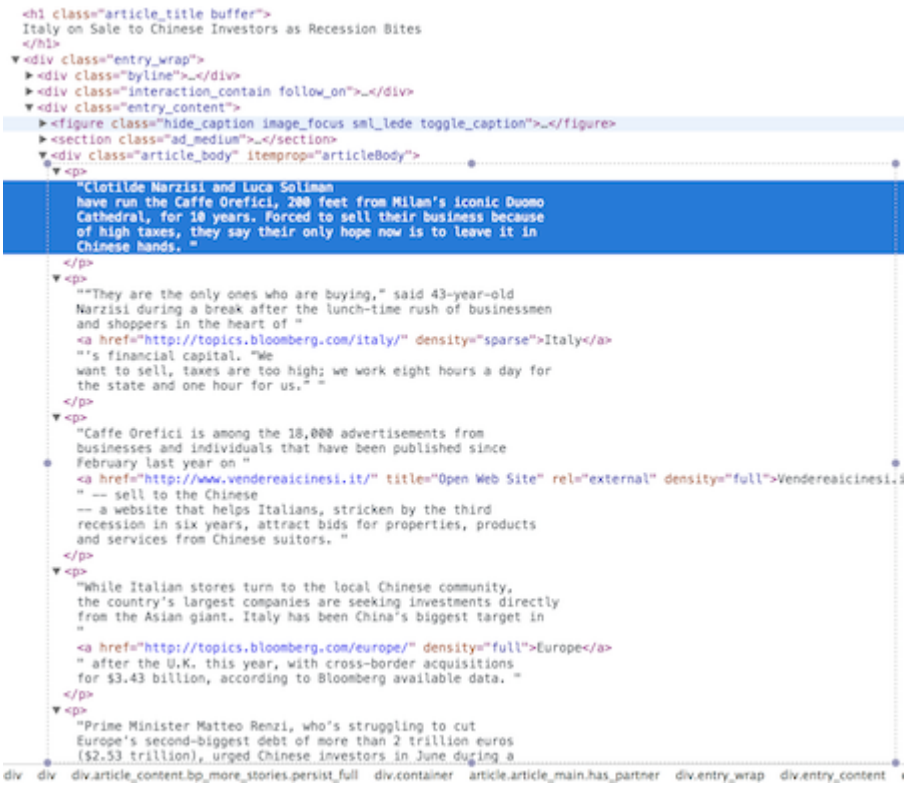
4.1 Εξόρυξη της πληροφορίας

Για την ανάλυση των νέων που γράφονται καθημερινώς χρειάστηκε να κατασκευαστεί ένα πρόγραμμα που είχε ως στόχο να αντλεί την πληροφορία από το διαδίκτυο. Επειδή στοχεύσαμε στα οικονομικά νέα, το πρόγραμμα αυτό είχε ως κύριο σημείο εξόρυξης της πληροφορίας τις ιστοσελίδες που έχουν το μεγαλύτερο αριθμό επισκεπτών και άρα το μεγαλύτερολογικά αντίκτυπο στην αγορά. Ορισμένες απο τις ιστοσελίδες που επικεντρωθήκαμε είναι οι ακόλουθες:

- Google finance (https://www.google.com/finance?hl=en)
- Yahoo finance (http://finance.yahoo.com/)
- Bloomberg (http://www.bloomberg.com/news/markets/)
- Cnbc (http://www.cnbc.com/)

Πολλές από τις οικονομικές σελίδες προσφέρουν ενημέρωση μέσω RSS. Ένα έγγραφο RSS περιλαμβάνει πλήρες κείμενο ή συνοπτικό κείμενο και μεταδεδομένα (metadata), όπως η ημερομηνία έκδοσης και το όνομα του συγγραφέα. Αυτό ο τρόπος ενημέρωσης είναι και ο ευκολότερος για να κατασκευαστεί ένα πρόγραμμα που μπορεί να προσπελάσει το κείμενο και να το μετατρέψει σε τοπικό αρχείο.
Ο επόμενος τρόπος εξόρυξης πληροφορίας μέσω ιστοσελίδων που είναι το ίδιο εύκολος στη κατασκευή είναι η κατανάλωση (consume) μιας διεπαφής προγραμματισμού εφαρμογών. Η Διεπαφή Προγραμματισμού Εφαρμογών (API) είναι μια διεπαφή των προγραμματιστικών διαδικασιών που ένα λειτουργικό σύστημα παρέχει προκειμένου να επιτρέπει να γίνονται προς αυτό αιτήσεις από άλλα

προγράμματα και ανταλλαγή δεδομένων [15]. Οι περισσότερες ιστοσελίδες σήμερα υποστηρίζουν την ανταλλαγή πληροφοριών μέσω ενός API.
Ο τελευταίος είναι ο τρόπος που ακολουθήσαμε για να εξορύξουμε την πληροφορία από τα κείμενα. Αυτή η διαδικασία μοιάζει με αυτό που κάνουν οι μηχανές αναζήτησης δηλαδή προσπελάσαμε/κατεβάσαμε αυτόματα όλοκληρες τις σελίδες (τον HTML κώδικα) που μας ενδιέφεραν καθώς και τις σελίδες που άνηκαν στην αρχική και ήταν προσβάσιμοι μέσω των συνδέσμων που περιείχε (web structure mining). Και μέσω ενός προκαθορισμένου τρόπου εξορύξαμε το κείμενο από τα περιττά στοιχεία του HTML κώδικα.



Σχήμα 4.1 Κείμενο στον HTML κώδικα από bloomberg.com.

4.2 Σχεδιασμός προγράμματος διαχείρησης πληροφορίας

Το πρόγραμμα κατασκευάστηκε στη JAVA και λαμβάνει την πληροφορία από το διαδίκτυο με τους παραπάνω τρόπους. Έχοντας στη μνήμη του υπολογιστή τα κείμενα μπορεί έπειτα να γίνει η επεξεργασία τους και να μετρηθούν αρχικά δίαφορα χαρακτηριστικά, όπως η συχνότητα των λέξεων, η συχνότητα που εμφανίζονται ορισμένες λέξεις στη σειρά κ.α. Φυσικά για το τελευταίο περιοριστήκαμε σε μικρό

αριθμό λέξεων που μπορούν να διατυπωθούν στη σειρά γιατί αλλιώς θα αντιμετωπίζαμε πρόβλημα με τη μνήμη αφού ο συνδυασμός των λέξεων μπορεί να είναι πολύπλοκος.
Τα κείμενα έπειτα τα αποθηκεύσαμε σε ιδιωτικές βάσεις δεδομένων και στην πραγματικότητα τα κρατήσαμε πάνω στο διαδίκτυο για πιθανή μελλοντική μελέτη και αυτό επιλέχθηκε γιατί ήταν δυνατό να δημιουργήσουμε συγκεκριμένο χώρο που θα αποτελούταν μόνο απο txt αρχεία.
Το πρόγραμμα το οποίο κατασκευάστηκε έχει την δυνατότητα να τραβάει την πληροφορία παράλληλα χρησιμοποιώντας τις κλάσεις Thread και ThreadPool αντίστοιχα [Appendix].

```
public class Multiple {
        public static void main(String[] args) {
        String n[] = {"googlef","yahoof","bloomberg","cnbc"};
        ExecutorService executor = Executors.newCachedThreadPool();
                for (int i = 0; i < n.length; i++) {
                        Source dt = new Source(n[i]);
                        executor.execute(new DataMineThread("Thead "+i, dt));
                }
        }
}
```

Παραπάνω είναι ένα κομμάτι κώδικα που ανοίγει επιτυχημένα ένα ThreadPool και έντασει μέσα σε αυτό αντικείμενα τύπου DataMineThread οποία επεκτείνουν την κλάση Thread και έχουν ως σκοπό να εξορύξουν την πληροφορία για τις 4 πηγές που του έχουμε ορίσει με ένα από τους τρείς τρόπους που αναφέρθηκαν πιο πριν.
Να σημειωθεί ότι οι αλφαριθμητικές μεταβλητές googlef yahoof Bloomberg cnbc αφορούν τις πηγές εξόρυξης πληροφορίας.

4.3 Επεξεργασία της πληροφορίας

Έχοντας λάβει την πληροφορία σε μορφή κειμένου προχωράμε στην επεξεργασία του κειμένου με στόχο να καταλήξουμε σε λέξεις που είναι σημαντικές ως προς το κείμενο που έχει γραφθεί.
Το πρώτο και βασικό βήμα είναι να επεξεργαστούμε γλωσσικά τα κείμενα που δεν έχουν λάβει μέχρι τώρα καμία επεξεργασία. Η διαδικασία μπορεί να περιγραφεί με τα παρακάτω βήματα:

1. Δημιουργήθηκε ο κώδικας που απομακρύνει τα μη λεκτικά σύμβολα όπως τα σημεία στήξης.

2. Επεξεργάστηκαν οι λέξεις που δεν προσδίδουν κανένα νόημα από μόνες τους στο κείμενο και παράλληλα εμφανίζονται πολύ συχνά γνωστές και ως stop-words [16].
3. Δεδομένου ότι πολλές λέξεις έχουν όμοια ρίζα αλλά το μήκος τους είναι διαφορετικό, δηλαδή έχουν διαφορετική μορφή αλλά παρουσιάζουν πολύ μεγάλη ομοιότητα αντιστοιχίσαμε τις λέξεις αυτές στη ρίζα τους (stemming) [17].
4. Πολλές από τις λέξεις έχουν αντίστοιχες συνώνυμες λέξεις. Έτσι με τη χρήση ενός λεξικού αντιστοιχίσαμε όλες τις σημασιολογικά ίδιες λέξεις με μια.

Για το πρώτο βήμα, την αφαίρεση των σημείων στήξης από το κείμενο, έγινε πολύ απλά με τον παρακάτω κώδικα:

```
    public static String characterCheck(String str){
      if (str.substring(str.length() - 1).equals(".") || str.substring(str.length() - 1).equals(";") ||
str.substring(str.length() - 1).equals(",")){
         return str.substring(0, str.length()-1);
      }else if(str.substring(str.length() - 1).equals("\"")){
         if(str.substring(str.length() - 2).equals(".") || str.substring(str.length() - 2).equals(";") ||
str.substring(str.length() - 2).equals(",")){
            return str.substring(0, str.length()-2);
         }else{
            return str.substring(0, str.length()-1);
         }
      }else{
         return str;
      }
   }
```

Για το δεύτερο βήμα αρχικά εντοπίστηκαν οι προθέσεις, τα άρθρα, οι αντωνυμίες κ.α. οι οποίες βρέθηκαν εύκολα στις πιο υψηλές θέσεις στο πίνακα συχνοτήτων των λέξεων και αφαιρέθηκαν από τον μελλοντικό έλεγχο με τον εξής τρόπο.
Το σύνολο το λέξεων αποτελείται από λέξεις όπως the, their, theirs, them, themselves, then, thence, there, there's, thereafter, thereby, therefore, therein, theres, thereupon, these, they, they'd, they'll, they're, they've, think, third κ.α.[Appendix]

Για το τρίτο βήμα χρησιμοποιήθηκε ο αλγόριθμος που αφαιρεί το πρόθεμα (prefix) και το απόθεμα (suffix) της κάθε λέξης ώστε να απομείνει μόνο η ρίζα. Ο αλγόριθμος που χρησιμοποιήθηκε είναι αυτός του Porter [18] ο οποίος υποστηρίζει επιτυχώς ότι οι καταλήξεις στην αγγλική γλώσσα δημιουργούνται απο συνδιασμούς μικρότερων και απλούστερων καταλήξεων.

Για το τέταρτο και τελευταίο βήμα χρησιμοποιήθηκε το ηλεκτρονικό λεξικό WordNet του Princeton [citation] το οποίο βασίζεται σε μια μεγάλη βάση δεδομένων από λέξεις και σημασιολογικές σχέσεις των λέξεων αυτών [19].
Για να καταναλώσουμε (consume) τον API του, WordNet Searching (JAWS) χρειάζεται να κάνουμε import το πακέτο του WordNet με την ακόλουθη εντολή:

```
import edu.smu.tspell.wordnet.*;
```

Και με τον ακόλουθο κώδικα μπορούμε να λάβουμε τα συνώνυμα (synsets) των λέξεων.

```
public static void main(String[] args){
if (args.length > 0){
//  Concatenate the command-line arguments
StringBuffer buffer = new StringBuffer();
      for (int i = 0; i < args.length; i++){
               buffer.append((i > 0 ? " " : "") + args[i]);
      }
      String wordForm = buffer.toString();
      //  Get the synsets containing the wrod form
WordNetDatabase database = WordNetDatabase.getFileInstance();
Synset[] synsets = database.getSynsets(wordForm);
//  Display the word forms and definitions for synsets retrieved
        if (synsets.length > 0){
               System.out.println("The following synsets contain '" +
wordForm + "' or a possible base form " + "of that text:");
        for (int i = 0; i < synsets.length; i++){
               System.out.println("");
               String[] wordForms = synsets[i].getWordForms();
               for (int j = 0; j < wordForms.length; j++){
                       System.out.print((j > 0 ? ", " : "") + wordForms[j]);
               }
               System.out.println(": " + synsets[i].getDefinition());
        }
        }else{
               System.err.println("No synsets exist that contain " + "the
word form '" + wordForm + "'");
        }
}else{
System.err.println("You must specify " + "a word form for which to retrieve
synsets.");
}
}
```

Τα παραπάνω χρησιμοποιήθηκαν για να μας δώσουν την εικόνα του κάθε κειμένου από το οποίο προσπαθούμε να εξορύξουμε την πληροφορία. Αυτή η εικόνα αφορά μόνο τον αριθμό των συχνοτήτων που εμφανίστηκαν οι λέξεις οι οποίες πλέον είναι ομαδοποιήμενες με βάση τη ρίζα τους (στελέχωση κειμένου) και αφήνοντας πίσω τις λέξεις που δεν έχουν κανένα νόημα καθώς και τα σημεία στήξης. . Παράλληλα, μπορούμε να εντοπίσουμε αποτελεσματικότερα τους όρους που θεωρούνται πιο

σημαντικοί με την απόδοση βάρους στο κάθε όρο. Το βάρος μπορεί να αποδοθεί με κάποιο αλγόριθμο που αναφέρθηκε στις προηγούμενες ενότητες ή με μια απλή μέθοδο βαρύτητας με βάση τη συχνότητα εμφάνισης της κάθε λέξης. Τέλος, αφού εκτελέστηκαν τα παραπάνω για ένα πρώτο μεγάλο δείγμα από λέξεις που ανήκουν σε ένα σύνολο κειμένων είχαμε τη δυνατότητα να μην λάβουμε υπόψην μας αυτές τις λέξεις που η συχνότητα εμφάνισης τους είναι πολύ μικρή (prune). Με αυτό τον τρόπο καταλήξαμε στο παρακάτω λεξιλόγιο (σύνολο λέξεων) (Σχήμα 4.2).

Good		Bad		
optimism	substantial	fell	devastate	bungle
rally	reassure	pessimistic	late	worst
profit	darling	rogue	missed	flood
kind	capable	violated	abrupt	invasion
surpass	win	betrayed	credibility	fails
securities	approved	resignation	inaccurate	corruption
unprecedented	monumental	misleading	suspicion	murder
recover	significantly	accuse	manipulation	discredit
better	interested	abusing	unclear	demise
benefit	major	deadly	sued	illegal
recources	best	outbreak	sank	marred
supporting	advanced	difficult	suffering	indictment
wealthy	reopen	annihilated	damaged	decimated
respect	hoping	kill	rejected	tainted
benign	favors	infectious	expire	dictatorship
quickly	correction	pandemics	disrupt	guilty
useful	poised	outbreak	critics	crime
wonderful	jumped	vulnerable	dispute	indicted
safe	sympathizes	weaken	portends	colluding
famed	progress	broken	bickering	murdered
success	renewed	rout	impedes	warning
fortunate	managing	loss	deficit	ridiculously
significant	updates	denouncing	struggling	fallen
nimble	reasonable	revolt	opposition	trouble
fantastic	aided	protests	bearish	suing
easily	agree	depressed	debasement	wondering
constructive	savers	collapse	skepticism	infractions
correction	convenient	panic	slump	fraud
good	great	undermine	crude	refugees
professional	improve	volatile	complaint	complaints
opportunities	clarity	fruitless	resigned	delinquency

preferred	precious	threats	obscured	blistering
valuable	outperform	aggressive	unemployment	imbalances
recommended	milestones	crisis	steal	roughly
merit	outperforming	damping	disgruntled	disastrous
usefulness	wisdom	unreliable	overpriced	attack
boon	looming	failed	diluted	worthless
attractive	achieve	abandon	bottlenecks	guarantees
major	steadily	worsen	unreasonable	difficult
bargain	plenty	contaminates	distressed	disaster
willing	relieve	pain	dethrone	destructive
savings	advantage	robbing	liability	suspects
favored	outstanding	disaster	retreat	wondering
strongest	intriguing	problem	assault	troubled
unrivaled	powerful	waning	mired	pollutes
popular	extraordinary	unfavorable	retaliate	disrupts
expansion	aid	rival	attack	degrades
kinds	laugh	delayed	militants	controversy
revived	surged	dumb	resisted	earthquake
strenght	inspired	tenable	revolt	unfortunate
cheer	triumphs	unskilled	jobless	mistakes
gained	soared	hurting	suspected	emergency
burgeoning	innovation	shocking	lethal	worry
efficiency	favoring	danger	lousy	shocked
surged	rescues	downgraded	conflict	refutes
		recession	clashed	bad
		nervous	enemy	risk

Σχήμα 4.2 Το λεξιλόγιο θετικών και αρνητικών λέξεων που εξήχθη χρησιμοποιώντας τον προτεινόμενο αλγόριθμο

Το λεξιλόγιο αυτό αποτελείται από 281 λέξεις. Μαζί με αυτές μπορέσαμε να δημιουργήσουμε και ένα σύνολο από ομάδες λέξεων που κάθε μια είχε από δύο εως τρείς λέξεις, οι οποίες είχαν την μεγαλύτερη συχνότητα στα κείμενα που αναλύθηκαν.
Γι'αυτές τις ομάδες λέξεων δεν λήφθηκε υπόψην το δεύτερο κριτήριο, αυτό των stop-words και σε πολλές από αυτές περιέχεται μέσα μια λέξη των stop-words. Ο τρόπος με τον οποίο καταλήξαμε σε αυτές προέκυψε αφού μελετήσαμε τα δεδομένα για μονές λέξεις. Παρατηρήσαμε πως οι μονές λέξεις που εμφανίζονται σε κάθε κείμενο έχουν ένα κοινό χαρακτηριστικό με τις ομάδες των λέξεων, το πόσο θετικές ή αρνητικές σημασιολογικά ήταν.

Good		Bad	
first company	be successful	probably fell	financial crisis
chief investment	net income	year-low	threatened retaliaton
solar panel	additional cash	repeatedly denied	atomic bombs
same time	mobile business	rogue actions	dangerous conflict
same day	great thing	at risk	overnight loans
financial stocks	fuel efficient cars	striking workers	record low
read growth	wholly owned	disruptions threatened	so different
enormous innovation	new plant	financial loss	deflationary pressure
big story	new attraction	record low	speculative positions
long-term investement	new attractions	be paid	terrible loss
high return	economic expansion	red light	negative growth
solid management	bullish long-term	political pressure	enviromental issues
net sales	person familiar	free cash	enormous pressure
shares soared	much cash	high price	
big day	very happy	total dept	
on track	much cash flow	was lower	
senior direct	annual performance	never get	
so inexpensive	something profound	massive cuts	
fastest growing	all-time high	tough competition	
open minded	also increased	thin air	
third round	significant increase	way too early	
foreign investment	good thing	persistent deflation	
profitable strategy	opportunity big time	federal dept	
all-time highs	significant difference	large speculators	
great deal	be worth	in dept	
successful calls	best-selling	allegedly sent	
global investment	significant achievement	ill will	
strong demand	worldwide sales leadership	net loss	
best shot	global best selling	civil war	
national expansion	global reach	reprisal threats	
most popular	burgeoning wealth	real concers	
tremendous amount	top-selling	low interest	

Σχήμα 4.3 Λεξιλόγιο διπλών και τριπλών λέξεων.

Το πρόγραμμα ανέλυσε σύμφωνα με το παραπάνω λεξιλόγιο διάφορα κείμενα και διαμόρφωσε για κάθε ένα από αυτά μια εικόνα η οποία είναι εξαρτημένη από την συχνότητα των εμφανίσεων των λέξεων του λεξιλογίου.

Κείμενο 4.1: Pandemic of pension woes plagues nation (πηγή: yahoo.finance.com και cnbc.com)

Detroit, you're not alone. Across the nation, cities and states are watching Detroit's largest-ever municipal bankruptcy filing with great trepidation. Years of underfunded retirement promises to public sector workers, which helped lay Detroit low, could plunge them into a similar and terrifying financial hole. A CNBC.com analysis of more than 120 of the nation's largest state and local pension plans finds they face a wide range of burdens as their aging workforces near retirement. Thanks to a patchwork of accounting practices and rosy investment assumptions, it's not even clear just how big a financial hole many states and cities have dug for themselves. That may soon change, thanks to a new set of government accounting standards that could serve as a nasty wake-up call to states and cities relying on rosy scenarios and head-in-the-sand accounting. Even less clear is who will pay to clean up the messes. Will it be the millions of retirees owed trillions of dollars in benefits, the bondholders who lent states and cities trillions more, or local taxpayers who may have to pay more to cover the shortfalls or see deeper cuts in public services? Regardless, the painful process will likely play out for years. "Moving pension plans is like steering a blimp: You turn the wheel and you go 6 miles before it starts to turn," said John Tuohy, Arlington County, Va., deputy treasurer, who chairs the pension committee of the Government Finance Officers Association. "In the political process, that kind of patience is very difficult." Many state and local governments have set aside enough money to comfortably make good on promised retirement benefits. Seventeen states have funded more than 80 percent of their projected pension liability, a level that's generally seen as financially sound. Most of the rest have been scrambling to make up investment losses inflicted by the 2008 market collapse and the shortfalls in sales, property and incomes taxes produced by the Great Recession. But even as the economy and housing markets have recovered, most states are still falling behind in closing their pension funding gaps. In the last year, 34 states have seen their pension funds stretched further as they've failed to make the full contributions needed to meet the projected cost of retirement promises. Much like a family that fails to save regularly to build a retirement nest egg, shortchanging those contributions increases the risk that the fund eventually will go broke. (Read more: If Detroit cuts pensions, will your city be next?) Nine states-Hawaii, Alaska, Kansas, Rhode Island, New Hampshire, Louisiana, Connecticut, Kentucky and Illinois-have now set aside less than 60 percent of what they

need. Illinois has saved just 43 cents to cover every dollar of what it needs to pay 350,000 retirees and 500,000 current plan participants who are counting on a pension check. In Detroit, city officials argue that pension payments to retirees simply have to be cut because the money just isn't there to pay them. But union officials there and in other cash-strapped cities say that's the city's problem. "Our members were promised certain things," said Tom Ryan, president of the firefighters' union in Chicago, where years of underfunding have prompted proposals to cut workers' retirement benefits. "They enter dangerous situations every day, and the only thing they want to look forward to when they can no longer perform their duties is to be able to retire with some sort of security. People expect us to be there, and we are always there. We expect that the city holds up their end of the bargain when we signed on to be firefighters and paramedics for the city of Chicago." Without a pension check, public sector workers face a bleak retirement. Many are ineligible for Social Security. "If we were talking about doing this to people with Social Security there would be rioting in the street," said Ryan. "But because it's public servants on pensions it seems to be OK to do this." Most cities and states with funding gaps still have time to fix the problem. Of the roughly 20,000 municipalities in the country, only a handful have completely run out of cash and been forced to seek shelter in bankruptcy court. (Read more: Detroit joins list of failed US towns) What's less clear is whether those states and cities have the political will to make the necessary, unpopular decisions, according to Jean-Pierre Aubry, assistant director of state and local research at the Center for Retirement Research at Boston College. "Even the ones that are really up against it have somewhere around 10 to 15 years of a window to turn things around," he said. "That's not a whole lot of time. And it's not clear if these governments that couldn't make contributions when times were good and the economy was better will be able to make them now or going forward." That political challenge is playing out in Illinois, where a $95 billion pension fund gap has deadlocked the state legislature over reform proposals for two years. In March, the nation's fifth most populous state settled with the Securities and Exchange Commission after the agency charged Illinois with fraud for misleading bond investors about its pension funding problems between 2005 and 2009. (Read more: Detroit bankruptcy case could bring unwanted change for munimarket) After lawmakers adjourned in May without fixing the problem, the state's credit rating was slashed, raising future borrowing costs. Earlier this summer, Gov. Pat Quinn used a budget line-item veto to freeze lawmakers' salaries effective August. Last week, the Illinois House speaker and Senate president hauled Quinn into court to get their pay restored. Such gridlock only increases the cost of fixing the problem. As pension payments consume a greater share of tax dollar-for example, Illinois' badly underfunded pensions now consume a fifth of the state's revenues-other services are starved of funds. "This is happening in too many cities and towns across America, where social services, because they can be cut, are cut, financial analyst Meredith Whitney, CEO and founder of Meredith Whitney Advisory

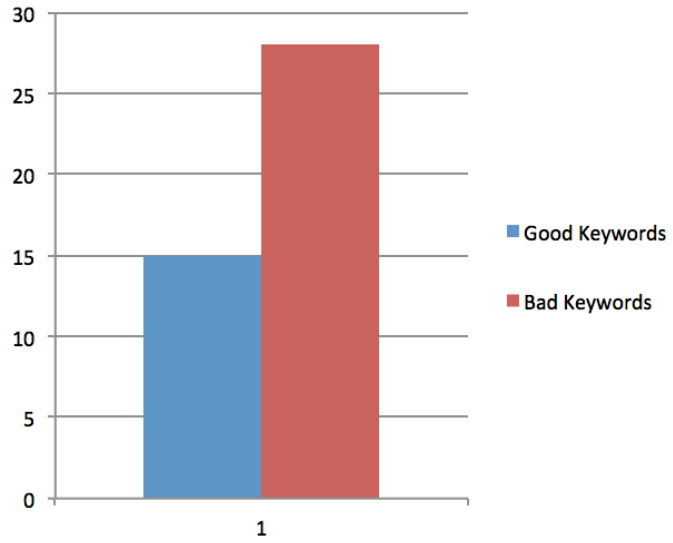
Group, told CNBC: "(But) because pensions and bonds constitutionally cannot be cut, they're the protected class. I think you're going to see a real issue of neighbor against neighbor on these very issues." But as Detroit has demonstrated, budget-balancing service cuts only lower the quality of life in a community, chasing residents away and further eroding the tax base. Raising taxes, on the other hand discourages new business expansion, further reducing revenues. For cities and states that have failed to fund their pension promises, it's a vicious cycle. To be sure, many jurisdictions have avoided falling into that the trap by keeping up with their pension promises. Even after sustaining heavy losses in the 2008 stock market crash, seven states-Wisconsin, South Dakota, North Carolina, Washington, New York, Tennessee and Delaware-have set aside more than 90 percent of their estimated future pension payments. "There are great stories about the cities and towns that are doing well that are investing in key things and being mindful of their fiscal discipline," said Whitney Assessing the financial health of a jurisdiction is compounded by the fuzzy math used to calculate just how much a state or city needs to set aside to meet its pension promises. Even in the best of times, pension accounting is fertile ground for voodoo economics and political expediency because those projections are based on a series of all but unknowable assumptions. It's hard to predict with precision, for example, just how many years of service a current employee will accumulate or how many checks they'll collect in their lifetime. Future pension cost estimates have been made more problematic by a common practice known as "spiking"-in which retirement-ready workers rack up hours of overtime and apply unused vacation to swell their last paychecks to lock in a higher monthly pension payment. (Read more: Detroit bottom could spell boom for Motor City entrepreneurs.) Managers of many underfunded plans have come up with a neat trick to make the problem seem to disappear. By simply assuming a higher investment return in the future, they can lowball the reported amount needed to meet future payments.

Even as low interest rates have badly depressed investment performance, many fund managers continue to project returns of 8 percent or higher. In 2011, the latest data available, the 100 largest public pension funds projected an average return of 7.84 percent. But their actual return over the prior 10 years was just 5.6 percent a year, according to a survey by Pensions and Investments, a trade publication. Eventually, those assumptions come home to roost, according to Rhode Island state Treasurer Gina Raimondo, who helped shepherd an overhaul of the state's ailing pension system in 2011. "Real people get hurt when politicians aren't honest and realistic about the magnitude of these issues," she said. "If you use a set of assumptions that makes the problem look smaller on paper today, that's irrelevant 10 years from now when the cash runs out and someone needs a pension check." (Read more: 'It's degrading': Bankrupt New England mill town offers Detroit a bleak preview) Many fund managers also use a technique called "smoothing"-which allows them to book investment gains and losses slowly for as long as five years. In good times, the scheme lets state and city officials

ride a bull market years after it's over, underfunding pensions to pay other government expenses, cut taxes or increase pension benefits without paying for the added longer-term cost. "The boom of the '90s was probably the worst thing that happened because (states and cities) ended up with a number of overfunded plans," said Arlington County's Tuohy. "And promises were made based on those overfunded plans." But smoothing also prolongs the pain of a severe market downturn, including heavy losses like those sustained in the 2008 market collapse. Thanks to smoothing, the burden of those investment losses continues to weigh on pension funds even as the stock market has recovered in the past year. That's one reason the Government Accounting Standards Board, which sets the bookkeeping rules for pension plan managers, has banned smoothing and requires underfunded plans to put away their rose-colored glasses when estimating future investment returns. When implemented next year, the new rules will paint a bleaker funding picture, cutting the average funding ratio of assets to liability, which stood at 75 percent in 2011, to 57 percent, according to a study by the Center for Retirement Research. Bond rating agency Moody's recently issued its own state-by-state reality check, based on its estimates of "adjusted" pension fund shortfalls that better reflect the new accounting realities. Moody's figures that 15 states are in better shape than the current reporting rules would indicate. The rest have bigger liabilities (some much bigger) than found in their current financial statements. Based on the adjusted funding gaps, nine states-Massachusetts, Pennsylvania, Colorado, Louisiana, Hawaii, New Jersey, Kentucky and Connecticut-would see their funding liabilities exceed an entire year's worth of state revenues. In Illinois, the adjusted funding gap amounts to 241 percent of state revenues. Since the Great Recession officially lifted in 2009, all 50 states have undertaken ever-more intense reforms as the pension funding problem deepened. This year alone, more than 1,200 bills have been introduced covering a range of fixes. The list includes suspending cost of living increases for retirees and shifting some of the investment risk on future retirees with the addition of a defined contribution plan similar to a 401(k). Some states have gone further by raising employee contributions or shifting the entire burden onto new workers with defined contribution plans. "That deals with the next generation," said Verne Sedlacek, president of Commonfund. "But we've got this whole group of people who worked for city and state governments for decades who had an expectation of payment. And they can't be paid." The most drastic reforms also leave jurisdictions at a marked disadvantage when they go to recruit the next generation of police, firefighters and teachers. Because cuts to public pension plans haven't been offset with wage increases, they leave public workers making about 20 percent less than their counterparts in the private sector, according to a Center for Retirement Research study. "So you now have two people with the same human capital-and the one in the private sector makes 20 percent more than the one in the public sector," said Boston College's

Aubry. "The question is how are you going to attract and retain quality public sector employees with that disparity?"

Σ'αυτό το κείμενο βλέπουμε τις λέξεις οι οποίες έχουν αρνητική σημασία να είναι με κόκκινο χρώμα ενώ οι λέξεις που έχουν θετική σημασία να είναι με πράσινο χρώμα. Από το κείμενο αυτό προκύπτει το παρακάτω διάγραμμα συχνοτήτων (Διάγραμμα 4.1):

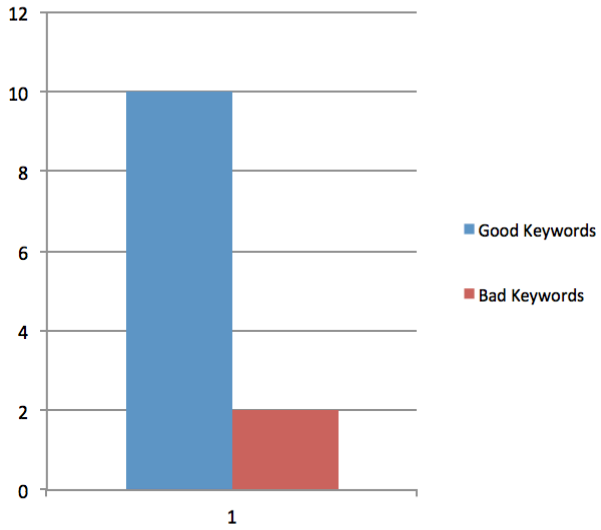


Διάγραμμα 4.1 Αριθμός εμφανίσεων των λέξεων στο κείμενο Pandemic of pension woes plagues nation (πηγή: yahoo.finance.com και cnbc.com)

Όπως φαίνεται στο Ιστόγραμμα μπορούμε εύκολα να καταλάβουμε αν για το θέμα του κείμενου γράφεται κάτι θετικό ή αρνητικό. Η συχνότητα εμφανίσεων των αρνητικών λέξεων είναι πολύ μεγαλύτερη από τις θετικές και αυτό επιβεβαιώνεται από τη σημασιολογία του κειμένου. Αν διαβάσουμε το κείμενο μπορούμε να δούμε πως όντως περιγράφεται το πόσο καταστροφικό στη προκειμένη περίπτωση αποτελεί το συνταξιοδοτικό πρόγραμμα για το Αμερικάνικο έθνος τη δεδομένη στιγμή που γράφθηκε αυτό το κείμενο.

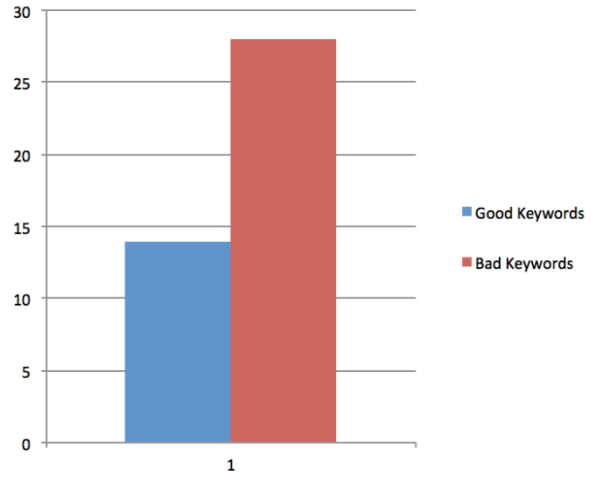
Αντίστοιχα παρουσιάζονται και άλλα κείμενα μόνο με τα διαγράμματα συχνοτήτων των λέξεων που στόχο έχουν να δείξουν ότι μόνο με το μέγεθος των συχνοτήτων μπορούμε να έχουμε μια εκτίμηση αρκετά καλή για το αν το κείμενο έχει γραφεί έτσι ώστε να περιγράψει κάτι θετικά ή αρνητικά.

Κείμενο 4.2: Goldman to Axa Ride London's Crossrail Office Boom (πηγή: Bloomberg.com).



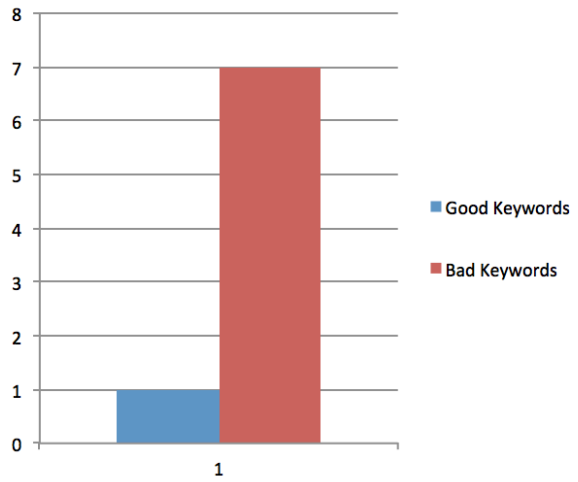
Διάγραμμα 4.2 Συχνότητες των λέξεων στο κείμενο Goldman to Axa Ride London's Crossrail Office Boom

Κείμενο 4.3: U.S. Stocks Fall Amid Growing Speculation on Fed Cuts (πηγή: Bloomberg.com).



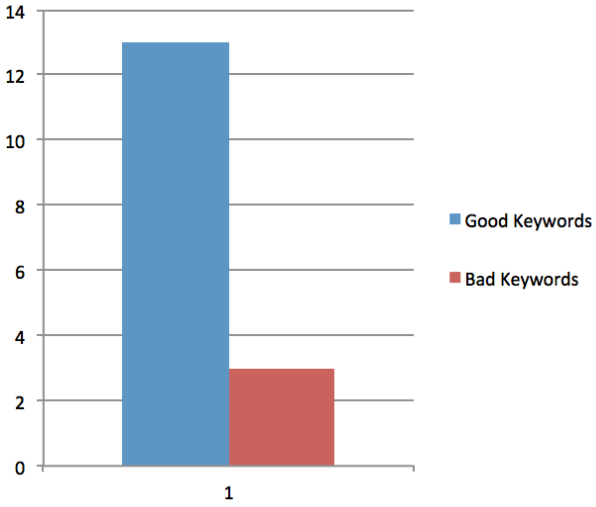
Διάγραμμα 4.3 Συχνότητες των λέξεων στο κείμενο U.S. Stocks Fall Amid Growing Speculation on Fed Cuts

Κείμενο 4.4: US natgas boom drags on nuclear power (πηγή: yahoo.finance.com).



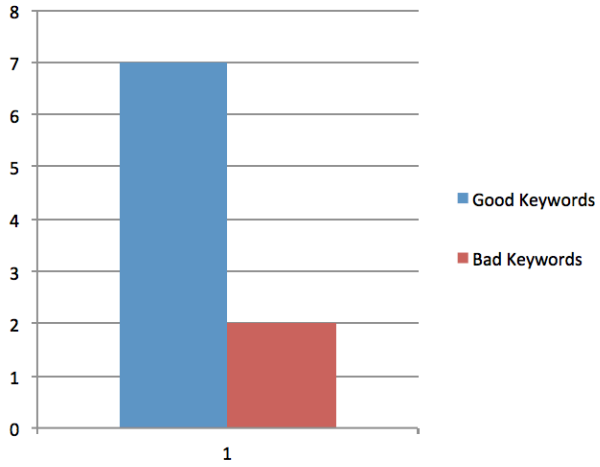
Διάγραμμα 4.4 Συχνότητες των λέξεων στο κείμενο US natgas boom drags on nuclear power

Κείμενο 4.5: You really like me? What now for Facebook's stock (πηγή: yahoo.finance.com).



Διάγραμμα 4.5 Συχνότητες των λέξεων στο κείμενο You really like me? What now for Facebook's stock

Κείμενο 4.6: The future of banking: Putting human tellers in ATMs (πηγή: cnbc.com)



Διάγραμμα 4.6 Συχνότητες των λέξεων στο κείμενο The future of banking: Putting human tellers in ATMs

Ο όρος της βαρύτητας μιας λέξης έχει να κάνει με το πόσο σημαντική είναι μια λέξη ως προς τη συχνότητα της σε σχέση με όλο το κείμενο. Όπως είδαμε στα κείμενα παρατηρούνται ορισμένες συνχότητες να είναι σχεδόν τριπλάσιες ή τετραπλάσιες σε σχέση με υπόλοιπες (4:1). Για να υπολογίσουμε ένα δείκτη βάρους για κάθε λέξη χρησιμοποιήσαμε τη διασπορά συχνοτήτων. Η διασπορά των λέξεων που κατείχαν υψηλότερη βαρύτητα στην ανάλυση μου είναι είχαν και την μεγαλύτερη διασπορά συχνοτήτων. Με τη βοήθεα της βιβλιοθήκης Apache Common Math και συγκεκριμένα των κλάσεων DescriptiveStatistics και SummaryStatistics υπολογίστηκαν τα βάρη των λέξεων.

Good			
optimism	0.02	substantial	0.02
rally	0.04	reassure	0.03
profit	0.05	darling	0.01
kind	0.04	capable	0.02
surpass	0.01	win	0.06
securities	0.02	approved	0.05
unprecedented	0.01	monumental	0.01
recover	0.06	significantly	0.05
better	0.13	interested	0.04

Σχήμα 4.4 Μέρος από τον πίνακα βάρους των θετικών λέξεων

Όπως βλέπουμε (Σχήμα 4.4) ισχύει η αρχική μας υπόθεση για τα βάρη των λέξεων που έχουν μεγαλύτερη συχνότητα εμφάνισης συνήθως τετραπλάσια ή πενταπλάσια σε σχέση με τις λέξεις που η συχνότητα του σε ένα κείμενο είναι πολλές φορές ίση με ένα ή μηδέν.
Έτσι λοιπόν για μια λέξη όπως better, best το βάρος της θα είναι ίσο με 13% σε σχέση με τη λέξη όπως monumental.

ΚΕΦΑΛΑΙΟ 5 : Συμπέρασμα

Με την εργασία αυτή καταφέραμε επιτυχώς να αποδείξουμε ότι μελετώντας ένα μεγάλο όγκο δεδομένων μπορούμε με τη βοήθεια διαφόρων επιστημών στο πεδίο της τεχνητής νοημοσύνης να καταλήξουμε σε σωστά συμπεράσματα σχετικά με το μήνυμα τωνκείμενων που γράφονται καθημερινά. Εφαρμόζοντας απλές τεχνικές μπορούμε να εξορύξουμε τη σημαντική πληροφορία από ένα κείμενο. Όσον αφορά το πως χαρακτηρίστηκαν ορισμένες λέξεις ως θετικές ή αρνητικές θα μπορούσαμε να είχαμε αποτυπώσει αυτόματα αυτό το χαρακτηριστικό αν μελετούσαμε ταυτόχρονα την χρονοσειρά μιας μετοχής. Δηλαδή είδαμε πως αν πρόκειται να γραφεί κάτι θετικό συνήθως οι λέξεις κλειδιά που βρήκαμε απαρτίζουν σχεδόν όλο το κειμενο. Επομένως, αν η μετοχή έδειχνε μια θετική κίνηση τότε και οι λέξεις θα μπορούσαν να χαρακτηριστούν ως θετικές και να συνεχίζει η «εκπαίδευση» του αλγορίθμου γιαυτές τις λέξει όσος μεγαλώνει η χρονοσειρά. Όπως μεγαλώνει η χρονοσειρά έτσι στο μέλλον μπορούμε να έχουμε και ένα μεγαλύτερο λεξιλόγιο. Σε αυτή την εργασία καταφέραμε καταναλώνοντας ένα συστήμα διεπαφής προγραμματισμού εφαρμογών και εξορύσσοντας την πληροφορία από τον HTML κώδικα διαφόρων ιστοσελίδων με οικονομικά νέα, να έχουμε πρόσβαση σε διάφορα κείμενα. Από αυτά τα κείμενα με τεχνικές εξόρυξης κειμένου καταφέραμε και καταλήξαμε σε ένα στοιχειώδη λεξιλόγιο το οποίο το δοκιμάσαμε σε καινούργια άρθρα που αφορούσαν είτε εταιρείες είτε δείκτες του χρηματιστηρίου. Η ανάλυση των κειμένων με βάση το πρόγραμμα μας, μας έδειξε πως σχεδόν πάντα μπορούμε να κατηγοριοποιήσουμε το άρθρο σε θετικό ή αρνητικό με τη βοήθεια απλών δεικτών βάρους που δημιουργήσαμε. Μια κατηγοριοποίηση είναι θετική όταν στο κείμενο παρουσιάζεται κάτι θετικό για το θέμα που αφορά και αρνητική αντίστοιχα.

Appendix 1: Τμήμα από κώδικα σε Java

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package dissertation;

import java.io.BufferedReader;
import java.io.DataInputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileWriter;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.util.ArrayList;

/**
 *
 * @author Dimitris
 */
public class Dissertation {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here /reading all the text
        ArrayList<Keywords> keySet = new ArrayList<Keywords>();
        File folder = new File("/Users/Dimitris/Documents/dissertation/articles0609");
        File[] listOfFiles = folder.listFiles();
        for (int i = 0; i < listOfFiles.length; i++) {
            File file = listOfFiles[i];
            if (file.isFile() && file.getName().endsWith(".txt")) {
                keySet.add(parseFile(file.toString()));
            }
        }
        //keySet.add(parseFile("/Users/Dimitris/Documents/dissertation/article3.txt"));
        generateTxtFile(keySet, "output0609.txt");
```

```
    }

    public static Keywords parseFile(String fileName){
        Keywords key = new Keywords(fileName);
        try{
            FileInputStream fstream = new FileInputStream(key.getFileName());
            DataInputStream in = new DataInputStream(fstream);
            BufferedReader br = new BufferedReader(new InputStreamReader(in));
            String strLine;
            boolean flag;
            int j;
            while ((strLine = br.readLine()) != null)   {
                String[] tokens = strLine.split(" ");
                //going through the words of the line
                for(int i=0; i<tokens.length; i++){
                    flag=true;
                    j=0;

                    //comparing the word with each element of the keywords array
                    while(flag && j<key.getKeyBad().length){
                        if(j<key.getKeyGood().length){//avoid the index out of bounds error
bad.length > good.length
                            if(characterCheck(tokens[i]).equals(key.getKeyGood()[j])){
                                key.getCounterGood()[j]++;
                                flag=false;
                                //System.out.println("G "+key.getKeyGood()[j]);
                            }
                        }
                        if(j<key.getKeyBad().length){
                            if(characterCheck(tokens[i]).equals(key.getKeyBad()[j])){
                                key.getCounterBad()[j]++;
                                flag=false;
                                //System.out.println("b "+key.getKeyBad()[j]);
                            }
                        }
                        j=j+1;
                    }
                }
            }
            in.close();
```

```
            key.sysout("Good");
        }catch (Exception e){
            System.err.println("Error: " + e.getMessage());
        }
        return key;
    }

    public static String characterCheck(String str){
        if (str.substring(str.length() - 1).equals(".")||str.substring(str.length() -
1).equals(";")||str.substring(str.length() - 1).equals(",")){
            return str.substring(0, str.length()-1);
        }else if(str.substring(str.length() - 1).equals("\"")){
            if(str.substring(str.length() - 2).equals(".")||str.substring(str.length() -
2).equals(";")||str.substring(str.length() - 2).equals(",")){
                return str.substring(0, str.length()-2);
            }else{
                return str.substring(0, str.length()-1);
            }
        }else{
            return str;
        }
    }

    private static void generateTxtFile(ArrayList<Keywords> set, String fileName){
        try{
            FileWriter fw = new FileWriter(fileName);
            PrintWriter pw = new PrintWriter(fw);
            for(int i=0; i<set.size(); i++){
                //Write to file for the first row
                pw.println(set.get(i).getFileName());
                pw.println("Good Keywords");
                for(int j=0; j<set.get(i).getKeyGood().length; j++){
                    if(set.get(i).getCounterGood()[j]!=0){
                        pw.print(set.get(i).getKeyGood()[j]);
                        pw.print(",");
                        pw.print(set.get(i).getCounterGood()[j]);
                        pw.println();
                    }
                }
                pw.println("Bad Keywords");
```

```
            for(int j=0; j<set.get(i).getKeyBad().length; j++){
                if(set.get(i).getCounterBad()[j]!=0){
                    pw.print(set.get(i).getKeyBad()[j]);
                    pw.print(",");
                    pw.print(set.get(i).getCounterBad()[j]);
                    pw.println();
                }
            }
            pw.println();
            //Flush the output to the file
            pw.flush();
        }
        //Close the Print Writer
        pw.close();
        //Close the File Writer
        fw.close();
    }
    catch(IOException e){
        e.printStackTrace();
    }
  }
}


/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package dissertation;

/**
 * @author Dimitris
 */
public class Keywords {
   private String keyGood[];
   private String keyBad[];
   private int counterGood[];
   private int counterBad[];
   private String fileName;
```

```
    public Keywords(String fileName){
        this.fileName = fileName;
        keyGood = new String[]{"optimism","rallied","rally","profit","kind","surpass","securities","unprecedented",
         000
"better","benefit","recources","supporting","wealthy","respect","benign","quickly","useful",
"wonderful",

"successful","safe","famed","success","fortunate","significant","nimble","fantastic","valua
ble","easily",

"constructive","correction","good","professional","opportunities","profitable","best","reco
mmended","merit",

"usefulness","boon","attractive","major","bargain","willing","savings","favored","strongest
","unrivaled",

"popular","expansion","kinds","succeeding","revived","strenght","profits","cheer","gained"
,"burgeoning",

"favoring","rescues","recover","reasonable","surged","achieved","strengthen","substantia
l","outstanding",

"agreement","darling","reassure","benefits","capable","win","approved","monumental","in
triguing","benefiting",

"interested","advanced","reopen","hoping","approval","profited","favors","poised","jumpe
d","efficiency",

"sympathizes","progress","renewed","managing","updates","improving","aided","agree","
savers","convenient",

"great","preferred","improve","clarity","precious","winning","outperform","milestones","sig
nificantly",

"reasonably","outperforming","wisdom","looming","achieve","steadily","plenty","relieve","
advantage",

"powerful","extraordinary","aid","laugh","opportunity","inspired","triumphs","soared","inno
vation",
```

```
    /*new*/          "stability","praise","heal","fabulous"};

        keyBad                                          =                                                  new
String[]{"fell","pessimistic","rogue","violated","betrayed","resignation","accusations","misl
eading","accused","abusing","deadly",

"outbreak","difficult","devastation","kill","killed","infectious","pandemics","vulnerable","we
akness","broken","threatens",

"loss","denouncing","revolt","protests","losing","depressed","collapse","panicked","under
mine","volatility","volatile","fruitless",

"threats","threaten","aggressive","crisis","weakening","unreliable","failed","abandon","wo
rsen","pain","robbing","disaster","problem",

"waning","unfavorable","rival","delayed","dumb","tenable","unskilled","hurting","painful","
danger","downgraded","recession","nervous",

"wondering","annihilated","bad","risk","lousy","troubled","struggling","unfortunate","mista
kes","emergency","devastate","late","missed",

"abrupt","abruptly","credibility","inaccurate","suspicion","manipulation","unclear","sued","
sank","suffering","damaged","rejected","expire",

"disrupt","critics","dispute","portends","bickering","impedes","deficit","weakened","opposi
tion","bearish","debasement","skepticism","slump",

"crude","complaint","resigned","obscured","stealing","steal","disgruntled","overpriced","di
luted","bottlenecks","unreasonable","distressed",

"dethrone","liability","retreat","attacked","assault","mired","retaliate","attack","attacks","mi
litants","resisted","accuses",

"suspected","lethal","abandoned","conflict","clashed","enemy","refugees","devasting","in
vasion","damping","lost","jobless","unemployment",

"rout","worry","shocked","losses","bungle","worst","loss","threat","worried","flood","weak
en","fails","corruption","murder","discredit",
```

```
"demise","illegal","marred","indictment","decimated","tainted","dictatorship","guilty","crim
e","indicted","colluding","murdered","warning",

"ridiculously","fallen","trouble","suing","violations","infractions","fraud","complaints","prob
lems","delinquencies","delinquency",

"blistering","imbalances","roughly","disastrous","disrupted","worthless","guarantees","de
structive","panic","suspects",

"criticism","pollutes","disrupts","degrades","contaminates","shocking","refutes","controve
rsy","earthquakes","earthquake",
     /*new*/          "vitriol","skeptics","fallout"};

        counterGood = new int[keyGood.length];
        counterBad = new int[keyBad.length];
    }
    public String getFileName(){
        return fileName;
    }
    public void setFileName(String fileName){
        this.fileName = fileName;
    }
    public String[] getKeyGood(){
        return keyGood;
    }
    public String[] getKeyBad(){
        return keyBad;
    }
    public int[] getCounterGood(){
        return counterGood;
    }
    public int[] getCounterBad(){
        return counterBad;
    }
    public void sysout(String type){
        if(type.equals("Good")){
            for(int i=0; i<getKeyGood().length; i++){
                if(getCounterGood()[i]!=0)
                    System.out.println(getKeyGood()[i]+" "+getCounterGood()[i]);
```

```
            }
        }
        if(type.equals("Bad")){
            for(int i=0; i<getKeyBad().length; i++){
                if(getCounterBad()[i]!=0)
                    System.out.println(getKeyBad()[i]+" "+getCounterBad()[i]);
            }
        }
    }
}

*/

import java.io.*;

class Stemmer
{  private char[] b;
   private int i,     /* offset into b */
               i_end, /* offset to end of stemmed word */
               j, k;
   private static final int INC = 50;
                    /* unit of size whereby b is increased */
   public Stemmer()
   {  b = new char[INC];
      i = 0;
      i_end = 0;
   }

   /**
    * Add a character to the word being stemmed.  When you are finished
    * adding characters, you can call stem(void) to stem the word.
    */

   public void add(char ch)
   {  if (i == b.length)
      {  char[] new_b = new char[i+INC];
         for (int c = 0; c < i; c++) new_b[c] = b[c];
         b = new_b;
      }
      b[i++] = ch;
   }


   /** Adds wLen characters to the word being stemmed contained in a portion
    * of a char[] array. This is like repeated calls of add(char ch), but
```

```
 * faster.
 */

public void add(char[] w, int wLen)
{  if (i+wLen >= b.length)
   {  char[] new_b = new char[i+wLen+INC];
      for (int c = 0; c < i; c++) new_b[c] = b[c];
      b = new_b;
   }
   for (int c = 0; c < wLen; c++) b[i++] = w[c];
}

/**
 * After a word has been stemmed, it can be retrieved by toString(),
 * or a reference to the internal buffer can be retrieved by getResultBuffer
 * and getResultLength (which is generally more efficient.)
 */
public String toString() { return new String(b,0,i_end); }

/**
 * Returns the length of the word resulting from the stemming process.
 */
public int getResultLength() { return i_end; }

/**
 * Returns a reference to a character buffer containing the results of
 * the stemming process.  You also need to consult getResultLength()
 * to determine the length of the result.
 */
public char[] getResultBuffer() { return b; }

/* cons(i) is true <=> b[i] is a consonant. */

private final boolean cons(int i)
{  switch (b[i])
   {  case 'a': case 'e': case 'i': case 'o': case 'u': return false;
      case 'y': return (i==0) ? true : !cons(i-1);
      default: return true;
   }
}

/* m() measures the number of consonant sequences between 0 and j. if c is
   a consonant sequence and v a vowel sequence, and <..> indicates arbitrary
   presence,

      <c><v>       gives 0
```

```
     <c>vc<v>     gives 1
     <c>vcvc<v>   gives 2
     <c>vcvcvc<v> gives 3
     ....
*/

private final int m()
{  int n = 0;
   int i = 0;
   while(true)
   {  if (i > j) return n;
      if (! cons(i)) break; i++;
   }
   i++;
   while(true)
   {  while(true)
      {  if (i > j) return n;
            if (cons(i)) break;
            i++;
      }
      i++;
      n++;
      while(true)
      {  if (i > j) return n;
         if (! cons(i)) break;
         i++;
      }
      i++;
    }
}

/* vowelinstem() is true <=> 0,...j contains a vowel */

private final boolean vowelinstem()
{  int i; for (i = 0; i <= j; i++) if (! cons(i)) return true;
   return false;
}

/* doublec(j) is true <=> j,(j-1) contain a double consonant. */

private final boolean doublec(int j)
{  if (j < 1) return false;
   if (b[j] != b[j-1]) return false;
   return cons(j);
}
```

```
/* cvc(i) is true <=> i-2,i-1,i has the form consonant - vowel - consonant
   and also if the second c is not w,x or y. this is used when trying to
   restore an e at the end of a short word. e.g.

      cav(e), lov(e), hop(e), crim(e), but
      snow, box, tray.

*/

private final boolean cvc(int i)
{  if (i < 2 || !cons(i) || cons(i-1) || !cons(i-2)) return false;
   {  int ch = b[i];
      if (ch == 'w' || ch == 'x' || ch == 'y') return false;
   }
   return true;
}

private final boolean ends(String s)
{  int l = s.length();
   int o = k-l+1;
   if (o < 0) return false;
   for (int i = 0; i < l; i++) if (b[o+i] != s.charAt(i)) return false;
   j = k-l;
   return true;
}

/* setto(s) sets (j+1),...k to the characters in the string s, readjusting
   k. */

private final void setto(String s)
{  int l = s.length();
   int o = j+1;
   for (int i = 0; i < l; i++) b[o+i] = s.charAt(i);
   k = j+l;
}

/* r(s) is used further down. */

private final void r(String s) { if (m() > 0) setto(s); }

/* step1() gets rid of plurals and -ed or -ing. e.g.

       caresses  ->  caress
       ponies    ->  poni
       ties      ->  ti
       caress    ->  caress
```

```
      cats      ->  cat

      feed      ->  feed
      agreed    ->  agree
      disabled  ->  disable

      matting   ->  mat
      mating    ->  mate
      meeting   ->  meet
      milling   ->  mill
      messing   ->  mess

      meetings  ->  meet

*/

private final void step1()
{  if (b[k] == 's')
   {  if (ends("sses")) k -= 2; else
      if (ends("ies")) setto("i"); else
      if (b[k-1] != 's') k--;
   }
   if (ends("eed")) { if (m() > 0) k--; } else
   if ((ends("ed") || ends("ing")) && vowelinstem())
   {  k = j;
      if (ends("at")) setto("ate"); else
      if (ends("bl")) setto("ble"); else
      if (ends("iz")) setto("ize"); else
      if (doublec(k))
      {  k--;
         {  int ch = b[k];
            if (ch == 'l' || ch == 's' || ch == 'z') k++;
         }
      }
      else if (m() == 1 && cvc(k)) setto("e");
   }
}

/* step2() turns terminal y to i when there is another vowel in the stem. */

private final void step2() { if (ends("y") && vowelinstem()) b[k] = 'i'; }

/* step3() maps double suffices to single ones. so -ization ( = -ize plus
   -ation) maps to -ize etc. note that the string before the suffix must give
   m() > 0. */
```

```
private final void step3() { if (k == 0) return; /* For Bug 1 */ switch (b[k-1])
{
   case 'a': if (ends("ational")) { r("ate"); break; }
             if (ends("tional")) { r("tion"); break; }
             break;
   case 'c': if (ends("enci")) { r("ence"); break; }
             if (ends("anci")) { r("ance"); break; }
             break;
   case 'e': if (ends("izer")) { r("ize"); break; }
             break;
   case 'l': if (ends("bli")) { r("ble"); break; }
             if (ends("alli")) { r("al"); break; }
             if (ends("entli")) { r("ent"); break; }
             if (ends("eli")) { r("e"); break; }
             if (ends("ousli")) { r("ous"); break; }
             break;
   case 'o': if (ends("ization")) { r("ize"); break; }
             if (ends("ation")) { r("ate"); break; }
             if (ends("ator")) { r("ate"); break; }
             break;
   case 's': if (ends("alism")) { r("al"); break; }
             if (ends("iveness")) { r("ive"); break; }
             if (ends("fulness")) { r("ful"); break; }
             if (ends("ousness")) { r("ous"); break; }
             break;
   case 't': if (ends("aliti")) { r("al"); break; }
             if (ends("iviti")) { r("ive"); break; }
             if (ends("biliti")) { r("ble"); break; }
             break;
   case 'g': if (ends("logi")) { r("log"); break; }
} }

/* step4() deals with -ic-, -full, -ness etc. similar strategy to step3. */

private final void step4() { switch (b[k])
{
   case 'e': if (ends("icate")) { r("ic"); break; }
             if (ends("ative")) { r(""); break; }
             if (ends("alize")) { r("al"); break; }
             break;
   case 'i': if (ends("iciti")) { r("ic"); break; }
             break;
   case 'l': if (ends("ical")) { r("ic"); break; }
             if (ends("ful")) { r(""); break; }
             break;
   case 's': if (ends("ness")) { r(""); break; }
```

```
            break;
} }

/* step5() takes off -ant, -ence etc., in context <c>vcvc<v>. */

private final void step5()
{   if (k == 0) return; /* for Bug 1 */ switch (b[k-1])
    {  case 'a': if (ends("al")) break; return;
       case 'c': if (ends("ance")) break;
                 if (ends("ence")) break; return;
       case 'e': if (ends("er")) break; return;
       case 'i': if (ends("ic")) break; return;
       case 'l': if (ends("able")) break;
                 if (ends("ible")) break; return;
       case 'n': if (ends("ant")) break;
                 if (ends("ement")) break;
                 if (ends("ment")) break;
                 /* element etc. not stripped before the m */
                 if (ends("ent")) break; return;
       case 'o': if (ends("ion") && j >= 0 && (b[j] == 's' || b[j] == 't')) break;
                                 /* j >= 0 fixes Bug 2 */
                 if (ends("ou")) break; return;
                 /* takes care of -ous */
       case 's': if (ends("ism")) break; return;
       case 't': if (ends("ate")) break;
                 if (ends("iti")) break; return;
       case 'u': if (ends("ous")) break; return;
       case 'v': if (ends("ive")) break; return;
       case 'z': if (ends("ize")) break; return;
       default: return;
    }
    if (m() > 1) k = j;
}

/* step6() removes a final -e if m() > 1. */

private final void step6()
{  j = k;
   if (b[k] == 'e')
   {  int a = m();
      if (a > 1 || a == 1 && !cvc(k-1)) k--;
   }
   if (b[k] == 'l' && doublec(k) && m() > 1) k--;
}

/** Stem the word placed into the Stemmer buffer through calls to add().
```

```
 * Returns true if the stemming process resulted in a word different
 * from the input.  You can retrieve the result with
 * getResultLength()/getResultBuffer() or toString().
 */
public void stem()
{  k = i - 1;
   if (k > 1) { step1(); step2(); step3(); step4(); step5(); step6(); }
   i_end = k+1; i = 0;
}

/** Test program for demonstrating the Stemmer.  It reads text from a
 * a list of files, stems each word, and writes the result to standard
 * output. Note that the word stemmed is expected to be in lower case:
 * forcing lower case must be done outside the Stemmer class.
 * Usage: Stemmer file-name file-name ...
 */
public static void main(String[] args)
{
   char[] w = new char[501];
   Stemmer s = new Stemmer();
   for (int i = 0; i < args.length; i++)
   try
   {
      FileInputStream in = new FileInputStream(args[i]);

      try
      { while(true)

        {  int ch = in.read();
           if (Character.isLetter((char) ch))
           {
              int j = 0;
              while(true)
              {  ch = Character.toLowerCase((char) ch);
                 w[j] = (char) ch;
                 if (j < 500) j++;
                 ch = in.read();
                 if (!Character.isLetter((char) ch))
                 {
                    /* to test add(char ch) */
                    for (int c = 0; c < j; c++) s.add(w[c]);

                    /* or, to test add(char[] w, int j) */
                    /* s.add(w, j); */

                    s.stem();
```

```
                {  String u;

                   /* and now, to test toString() : */
                   u = s.toString();

                   /* to test getResultBuffer(), getResultLength() : */
                   /* u = new String(s.getResultBuffer(), 0, s.getResultLength()); */

                   System.out.print(u);
                }
                break;
              }
            }
          }
          if (ch < 0) break;
          System.out.print((char)ch);
        }
      }
      catch (IOException e)
      {  System.out.println("error reading " + args[i]);
         break;
      }
    }
    catch (FileNotFoundException e)
    {  System.out.println("file " + args[i] + " not found");
       break;
    }
  }
}
```

Appendix 2: Δείγμα συχνοτήτων μετά από τον έλεγχο λεξιλογίου

/articles0608/Batista Said to Grant Lenders Collateral Swap- Corporate Brazil.txt
Good Keywords
securities,7
better,2
outstanding,1
agreement,1
Bad Keywords
risk,1
missed,1
slump,1

liability,1
losses,1
worst,1
fallen,2
guarantees,1

/articles0608/Family Offices Chasing Wealthy’s $46 Trillion in Assets.txt
Good Keywords
profit,1
wealthy,6
managing,1
inspired,1
Bad Keywords
crisis,2
problem,1
bad,1
lost,1
trouble,1

/articles0608/Goldman to Axa Ride London’s Crossrail Office Boom.txt
Good Keywords
profit,1
better,1
benefit,1
best,1
willing,1
benefits,1
benefiting,1
approval,2
managing,1
Bad Keywords
weakness,1
risk,1

/articles0608/McDonald’s Franchisees Go Rogue With Meetings.txt
Good Keywords
profit,1
better,2
quickly,1
good,2
profitable,1
expansion,1
renewed,2
great,1
improve,1

Bad Keywords
fell,3
weakness,1
revolt,1
struggling,1
unemployment,1

/articles0608/Public Pensions Up 12% Get Most in 2 Years as Stocks Soar.txt
Good Keywords
better,1
benefit,2
significant,1
good,1
gained,1
benefits,2
managing,1
improving,1
Bad Keywords
fell,1
loss,1
volatile,1
crisis,2
recession,1
emergency,1
deficit,2
lost,2
losses,3
weaken,1

/articles0608/Standard Chartered First-Half Profit Rises 4%.txt
Good Keywords
profit,7
good,1
expansion,2
Bad Keywords
fell,3
difficult,1
loss,1
bad,1
missed,1
losses,1

/articles0608/U.S. Stocks Decline Before Speech From Fed’s Evans.txt
Good Keywords
rallied,2
profit,5

correction,1
gained,1
advanced,2
Bad Keywords
fell,5
volatility,1
downgraded,1
critics,1
deficit,1
crude,2
lost,1
fallen,1

Βάρη θετικών λέξεων:

Good			
optimism	0.02	substantial	0.02
rally	0.04	reassure	0.03
profit	0.05	darling	0.01
kind	0.04	capable	0.02
surpass	0.01	win	0.06
securities	0.02	approved	0.05
unprecedented	0.01	monumental	0.01
recover	0.06	significantly	0.05
better	0.13	interested	0.04
benefit	0.08	major	0.03
recources	0.02	best	0.11
supporting	0.02	advanced	0.02
wealthy	0.07	reopen	0.03
respect	0.03	hoping	0.04
benign	0.02	favors	0.03
quickly	0.07	correction	0.02
useful	0.08	poised	0.03
wonderful	0.03	jumped	0.07
safe	0.09	sympathizes	0.02
famed	0.06	progress	0.04
success	0.10	renewed	0.03
fortunate	0.03	managing	0.03
significant	0.04	updates	0.06
nimble	0.02	reasonable	0.03
fantastic	0.06	aided	0.02
easily	0.03	agree	0.04
constructive	0.01	savers	0.02

correction	0.03	convenient	0.03
good	0.10	great	0.09
professional	0.02	improve	0.06
opportunities	0.07	clarity	0.02
preferred	0.03	precious	0.01
valuable	0.03	outperform	0.02
recommended	0.02	milestones	0.02
merit	0.01	outperforming	0.02
usefulness	0.04	wisdom	0.01
boon	0.01	looming	0.01
attractive	0.04	achieve	0.04
major	0.05	steadily	0.06
bargain	0.03	plenty	0.06
willing	0.04	relieve	0.05
savings	0.03	advantage	0.07
favored	0.04	outstanding	0.01
strongest	0.05	intriguing	0.02
unrivaled	0.01	powerful	0.07
popular	0.05	extraordinary	0.05
expansion	0.04	aid	0.04
kinds	0.03	laugh	0.02
revived	0.04	surged	0.05
strenght	0.04	inspired	0.03
cheer	0.06	triumphs	0.04
gained	0.02	soared	0.02
burgeoning	0.02	innovation	0.03
efficiency	0.03	favoring	0.04
surged	0.05	rescues	0.05

Appendix 3: Λίστα stop-words

a, a’s, able, about, above, according, accordingly, across, actually, after, afterwards, again, against, ain’t, all, allow, allows, almost, alone, along, already, also, although, always, am, among, amongst, an, and, another, any, anybody, anyhow, anyone, anything, anyway, anyways, anywhere, apart, appear, appreciate, appropriate, are, aren’t, around, as, aside, ask, asking, associated, at, available, away, awfully, be, became, because, become, becomes, becoming, been, before, beforehand, behind, being, believe, below, beside, besides, best,

better, between, beyond, both, brief, but, by, c'mon, c's, came, can, can't, cannot, cant, cause, causes, certain, certainly, changes, clearly, co, com, come, comes, concerning, consequently, consider, considering, contain, containing, contains, corresponding, could, couldn't, course, currently, definitely, described, despite, did, didn't, different, do, does, doesn't, doing, don't, done, down, downwards, during, each, edu, eg, eight, either, else, elsewhere, enough, entirely, especially, et, etc, even, ever, every, everybody, everyone, everything, everywhere, ex, exactly, example, except, far, few, fifth, first, five, followed, following, follows, for, former, formerly, forth, four, from, further, furthermore, get, gets, getting, given, gives, go, goes, going, gone, got, gotten, greetings, had, hadn't, happens, hardly, has, hasn't, have, haven't, having, he, he's, hello, help, hence, her, here, here's, hereafter, hereby, herein, hereupon, hers, herself, hi, him, himself, his, hither, hopefully, how, howbeit, however, i'd, i'll, i'm, i've, ie, if, ignored, immediate, in, inasmuch, inc, indeed, indicate, indicated, indicates, inner, insofar, instead, into, inward, is, isn't, it, it'd, it'll, it's, its, itself, just, keep, keeps, kept, know, knows, known, last, lately, later, latter, latterly, least, less, lest, let, let's, like, liked, likely, little, look, looking, looks, ltd, mainly, many, may, maybe, me, mean, meanwhile, merely, might, more, moreover, most, mostly, much, must, my, myself, name, namely, nd, near, nearly, necessary, need, needs, neither, never, nevertheless, new, next, nine, no, nobody, non, none, noone, nor, normally, not, nothing, novel, now, nowhere, obviously, of, off, often, oh, ok, okay, old, on, once, one, ones, only, onto, or, other, others, otherwise, ought, our, ours, ourselves, out, outside, over, overall, own, particular, particularly, per, perhaps, placed, please, plus, possible, presumably, probably, provides, que, quite, qv, rather, rd, re, really, reasonably, regarding, regardless, regards, relatively, respectively, right, said, same, saw, say, saying, says, second, secondly, see, seeing, seem, seemed, seeming, seems, seen, self, selves, sensible, sent, serious, seriously, seven, several, shall, she, should, shouldn't, since, six, so, some, somebody, somehow, someone, something, sometime, sometimes, somewhat, somewhere, soon, sorry, specified, specify, specifying, still, sub, such, sup, sure, t's, take, taken, tell, tends, th, than, thank, thanks, thanx, that, that's, thats, the, their, theirs, them, themselves, then, thence, there, there's, thereafter, thereby, therefore, therein, theres, thereupon, these, they, they'd, they'll, they're, they've, think, third, this, thorough, thoroughly, those, though, three, through, throughout, thru, thus, to, together, too, took, toward, towards, tried, tries, truly, try, trying, twice, two, un, under, unfortunately, unless, unlikely, until, unto, up, upon, us, use, used, useful, uses, using, usually, value, various, very, via, viz, vs, want, wants, was, wasn't, way, we, we'd, we'll, we're, we've, welcome, well, went, were, weren't, what, what's, whatever, when, whence, whenever, where, where's, whereafter, whereas, whereby, wherein, whereupon, wherever, whether, which, while, whither, who, who's, whoever, whole, whom, whose, why, will, willing, wish, with, within, without, won't, wonder, would, would, wouldn't, yes, yet, you, you'd, you'll, you're, you've, your, yours, yourself, yourselves, zero

ΒΙΒΛΙΟΓΡΑΦΙΑ

1. https://www.data.gov/open-gov/
2. Whitby, B., Reflections on Artificial Intelligence: The Legal, Moral and Ethical Dimensions. Intellect, Oxford, p. 20 (1996)
3. Barr, A. and Feigenbaum, E., Handbook of Artificial Intelligence 1, Pitman, p. 3 (1981)
4. Rich, E., Artificial Intelligence, McGraw-Hill, p. 1 (1983)
5. Russell, S.J. and Norvig, P., Artificial Intelligence: A Modern Approach, 2nd ed., Upper Saddle River, New Jersey: Prentice Hall, (2003)
6. Luger, G. and Stubblefield, W., Artificial Intelligence: Structures and Strategies for Complex Problem Solving, 5th ed, The Benjamin/Cummings Publishing Company, Inc, p. 623–630 (2004).
7. Stumme, G., Hotho, A. and Berendt, B., Web Semantics: Science, Services and Agents on the World Wide Web, Semantic Grid - The Convergence of Technologies 4, Issue 2, p. 124-143 (2006)
8. Wu, X., Kumar, V., Quinlan, J.R., Ghosh, J., Yang, Q., Motoda, H., McLachlan, J.G., Ng, A., Liu, B., Yu, P.S., Zhou, Z., Steinbach, M., Hand, D.J. and Steinberg, D., Top 10 algorithms in data mining, Knowledge and information systems 14, Issue 1, p. 1-37 (2007)
9. Salzberg, S.L., C4.5: Programs for Machine Learning 16, Issue 3, J. Ross Quinlan. Morgan Kaufmann Publishers, Inc., p 235-240 (1994)

10. Xu, G., Zhang, Y. and Li, L., Web Mining and Social Networking Techniques and Applications, Springer , London p. 5-11 (2011)
11. Umbrich, J., Karnstedt, M. and Land, S., Towards Understanding the Changing Web: Mining the Dynamics of Linked-Data Sources and Entities (in progress)
12. Singh, P.K. and Husain, M.S., Methodological Study of Opinion Mining and Sentiment Analysis Techniques, International Journal on Soft Computing (IJSC) 5, (2014)
13. Fung, G.P.C., Yu, J.X. and Lam, W., Stock prediction: Integrating text mining approach using real-time news, Computational Intelligence for Financial Engineering, IEEE, p. 395 - 402 (2003)
14. Nikfarjam, A., Emadzadeh, E. and Muthaiyah, S., Text mining approaches for stock market prediction, Computer and Automation Engineering, IEEE, p. 256 - 260 (2010)
15. Customer Information Manager (CIM) SOAP API Documentation (2013)
16. Silva, C. and Ribeiro, B., The importance of stop word removal on recall values in text categorization, IEEE International Joint Conference on Neural Networks, IJCNN, p. 1661-1666 (2003)
17. Rao, Y., Lei, J., Wenyin, L., Li, Q. and Chen, M., Building emotional dictionary for sentiment analysis of online news, World Wide Web, 17(4), p. 723-742 (2014)
18. Porter, M.F., Program 14, (3), p 130-137 (1980)
19. Miller, G.A., Beckwith, R., Fellbaum, C., Gross, D. and Miller, K.J., Introduction to WordNet: An On-line Lexical Database, Int J Lexicography 3, (4) p. 235-244 (1990)