ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Εξόρυξη Δεδομένων από το Twitter και Εφαρμογή Αλγορίθμων Μη-Επιβλεπόμενης Μηχανικής Μάθησης για Συσταδοποίηση Κειμένων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

Ομήρου Πανταζή

Επιβλέπων : Ιωάννης Βασιλείου
Καθηγητής Ε.Μ.Π.

Αθήνα, Απρίλιος 2016

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Εξόρυξη Δεδομένων από το Twitter και Εφαρμογή Αλγορίθμων Μη-Επιβλεπόμενης Μηχανικής Μάθησης για Συσταδοποίηση Κειμένων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

Ομήρου Πανταζή

Επιβλέπων : Ιωάννης Βασιλείου
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 1η Απριλίου 2016.

(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*
.....	.....	.....
Ιωάννης Βασιλείου	Στέφανος Κόλλιας	Ανδρέας-Γεώργιος Σταφυλοπάτης
Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.

Αθήνα, Απρίλιος 2016

(Υπογραφή)

.....

ΟΜΗΡΟΣ ΠΑΝΤΑΖΗΣ

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

Περίληψη

Στη σημερινή εποχή, το ποσοστό των Χρηστών κοινωνικών δικτύων που εκμεταλλεύονται την ευκαιρία που τους δίνεται από αυτά να εκφράσουν την άποψη τους πάνω σε ένα συγκεκριμένο Θέμα αυξάνεται καθημερινά. Αντικείμενο μελέτης αυτής της διπλωματικής εργασίας είναι η συσχέτιση απόψεων διαφόρων Χρηστών πάνω σε Θέματα της επικαιρότητας. Τα Θέματα αυτά μπορεί να αναφέρονται σε πολιτική, οικονομικά, αθλητισμό, στα μέσα μαζικής ενημέρωσης κλπ. Το κοινωνικό δίκτυο ενδιαφέροντος για αυτή την έρευνα είναι το Twitter. Για την συλλογή των δεδομένων έγινε χρήση των δυνατοτήτων του προγραμματιστικού περιβάλλοντος Twitter API και για την αποθήκευση τους η Μη-Σχεσιακή βάση δεδομένων τύπου γράφου, Neo4j. Ακολούθως πετύχαμε αυτόματη Μοντελοποίηση των δεδομένων σε Θέματα με χρήση των αλγορίθμων μη-επιβλεπόμενης μηχανικής μάθησης, Latent Dirichlet Allocation (LDA) και K-Means. Για την επίτευξη του παραπάνω χρησιμοποιήσαμε Απλό Κείμενο, Επισημασμένα Ονόματα Χρηστών του Twitter και Hashtags. Τα αποτελέσματα της μελέτης μπορούν να ερμηνευτούν εύκολα μέσω της οπτικοποίησης τους σε διάγραμμα διασποράς. Το σύστημα έχει αναπτυχθεί κατά μεγάλο βαθμό με τη γλώσσα προγραμματισμού Python και τις ποικίλες βιβλιοθήκες που αυτή προσφέρει. Τέλος, η οπτικοποίηση των αποτελεσμάτων των παραπάνω αλγορίθμων καθώς και η δυνατότητα εφαρμογής τεχνικών ανάλυσης πάνω στα δεδομένα μας, προσφέρονται στο χρήστη μέσω Web Εφαρμογής που δημιουργήθηκε με το πλαίσιο Flask.

Λέξεις κλειδιά: κοινωνικά δίκτυα, Twitter, μηχανική μάθηση, συσταδοποίηση, εξόρυξη δεδομένων, ανάλυση δεδομένων, συσταδοποίηση εγγράφων, βάσεις δεδομένων γράφου, Neo4j, Cypher, LDA, K-Means, Python

Abstract

Nowadays, an increasing percentage of social network Users take advantage of the opportunity they have been given to express their opinion on a specific Topic. Subject of this thesis is the opinion correlation from a variety of social network Users based on text data we have collected and are focused on conversations around specific trending Topics. These Topics can be referring to politics, economics, sports, media etc. The social network of interest of this study is Twitter. For the purpose of Data Mining and Data Storage we exploited the capabilities of Twitter API and Neo4j Non-Relational graph database respectively. Subsequently, we achieved automated Topic Modeling using Unsupervised Machine Learning algorithms Latent Dirichlet Allocation (LDA) and K-Means. To achieve the above mentioned goal we used pure Tweet text, Mentioned Twitter Usernames and Hashtags. The results of the research can be easily interpreted through their visualization in a scatter diagram. The visualization of the above mentioned results along with the capability to perform various techniques of data analysis are available to the User through a Web Application built on top of the Flask web framework.

Key words: social networks, Twitter, machine learning, clustering, data mining, data analysis, document clustering, graph databases, Neo4j, Cypher, LDA, K-Means, Python

Ευχαριστίες

Αρχικά, θα ήθελα να ευχαριστήσω θερμά τον Λέκτορα Χριστόδουλο Ευσταθιάδη και τον μεταδιδακτορικό ερευνητή Άρη Μπελεσιώτη τόσο για την ανεκτίμητη και έγκαιρη καθοδήγηση τους μέσα από τις συζητήσεις μας όσο και για τον χρόνο που μου αφιέρωσαν γενικότερα.

Ευχαριστώ ιδιαιτέρως τον επιβλέποντα Καθηγητή κ. Ιωάννη Βασιλείου, που μου επέτρεψε να εργαστώ πάνω στο θέμα της διπλωματικής εργασίας που επέλεξα.

Ευχαριστώ επίσης τον Επίκουρο Καθηγητή Φώτιο Πετρόπουλο για την βοήθεια και την καθοδήγηση του από τα σχολικά μέχρι και τα φοιτητικά μου χρόνια.

Θέλω επίσης να ευχαριστήσω τους γονείς μου Ανδρέα και Δέσποινα, τον αδερφό μου Ιάσονα αλλά και όλη την οικογένεια μου για τη στήριξή τους καθ'όλη τη διάρκεια των σπουδών μου.

Ακόμα θα ήθελα να ευχαριστήσω τους φίλους και συναδέλφους Χατζ και Μήτρο για τις ατελείωτες ώρες που περάσαμε μαζί ευχάριστα, διαβάζοντας παρέα καθ'όλη τη διάρκεια της φοίτησης μας στην σχολή των ΗΜΜΥ, αλλά και όλους τους υπόλοιπους φίλους μου για την στήριξη τους και την πολύτιμη φιλία τους η οποία διαρκεί ήδη πολλά περισσότερα χρόνια από τη διάρκεια των σπουδών μου. Βασίλη, Μάξιμε, Γιάννη, Αντώνη, Πέτρο, Τζαννέ, Θανάση, Βαγγέλη, Ποσειδώνα, Ντίνο σας ευχαριστώ.

Τέλος, ευχαριστώ την Έλενα για τη στήριξη και την υπομονή της προς το πρόσωπο μου.

Πίνακας Περιεχομένων

Κατάλογος Σχημάτων

1

Εισαγωγή

1.1 Κοινωνικά Δίκτυα και Εξόρυξη Δεδομένων

Τα τελευταία χρόνια όλο και περισσότεροι χρήστες του διαδικτύου κάνουν χρήση των μέσων κοινωνικής δικτύωσης. Ενδεικτικά αναφέρουμε ότι το 2016 ο αριθμός των χρηστών κοινωνικών δικτύων υπολογίζεται να φτάσει τα 2.13 δισεκατομμύρια παγκοσμίως από 1.4 δισεκατομμύρια που ήταν το 2012 **[ST].** Εύκολα συμπεραίνει κανείς ότι αυτή η τάση είναι ικανή να δημιουργήσει τεράστιο όγκο πληροφορίας γύρω από αυτά. Μερικοί από τους λόγους που ωθούν τους ανθρώπους στην χρήση των μέσων κοινωνικής δικτύωσης είναι η δυνατότητα έκφρασης προσωπικής άποψης,η επικοινωνία με άλλους χρήστες, η ενημέρωση για θέματα που τους αφορούν και η διαφήμιση.

Το επικρατέστερο κοινωνικό δίκτυο στις μέρες μας είναι το Facebook το οποίο αριθμεί 1.55 δισεκατομμύρια ενεργούς χρήστες **[ST]** και παίζει σημαντικό ρόλο στην πορεία των κοινωνικών δικτύων από τη στιγμή έναρξης της λειτουργίας του μέχρι και σήμερα. Ένα εξίσου σημαντικό κοινωνικό δίκτυο με βασικό γνώρισμα την έκφραση απόψεων είναι το Twitter το οποίο αριθμεί 316 εκατομμύρια ενεργούς χρήστες **[ST]** και έχει μορφή μικροιστολογίου. Για να πάρουμε μια ιδέα σχετικά με τον όγκο της πληροφορίας που παράγεται καθημερινά από το Twitter αναφέρουμε οτι ο μέσος αριθμός των ενεργών χρηστών για μια ημέρα εκτιμάται στα 100 εκατομμύρια και τα τιτιβίσματα αυτών 58 εκατομμύρια **[SB].**

Μπορεί να γίνει εύκολα αντιληπτό οτι αυτός ο κατακλυσμός των κοινωνικών δικτύων από δεδομένα διαφόρων μορφών δημιουργεί την ανάγκη *εξόρυξης, επεξεργασίας* και *ανάλυσης* αυτών έτσι ώστε να μετατραπούν σε πληροφορίες που θα μας επιτρέψουν να εξάγουμε χρήσιμα συμπεράσματα ανάλογα με το εκάστοτε θέμα ενδιαφέροντος. Το μεγαλύτερο κομμάτι της

πληροφορίας που διαχέεται στο διαδίκτυο έχει τη μορφή κειμένου. Για αυτό το λόγο η *εξόρυξη κειμένου* είναι πολύ σημαντική στις μέρες μας. Μια συγκεκριμένη περιοχή έρευνας γύρω από την *εξόρυξη κειμένου (Text Mining)*, καταλαμβάνει η *συσταδοποίηση εγγράφων (Document Clustering)*. Η *συσταδοποίηση εγγράφων* βρίσκει αρκετές εφαρμογές στον πραγματικό κόσμο όπως συμμετοχή στο φιλτράρισμα της πληροφορίας που προσφέρουν ορισμένες μηχανές αναζήτησης. Προκειμένου να γίνει ευκολότερη γιά κάποιον χρήστη η πρόσβαση στην πληροφορία που αυτός επιθυμεί να δεί, μπορούν να χρησιμοποιηθούν τέτοιες μέθοδοι ομαδοποίησης των πιθανών εγγράφων που τον ενδιαφέρουν σε μια λίστα σημαντικών κατηγοριών. Ένα παράδειγμα χρήσης μεθόδων συσταδοποίησης εγγράφων προσφέρει η *Google* για να επιστρέφει στα εκάστοτε ερωτήματα χρηστών συγκεκριμένες ιστοσελίδες, αντιμετωπίζοντας τις ως *Συλλογές Θεμάτων* (*Topics*).

1.2 Αντικείμενο της Διπλωματικής

Σκοπός της παρούσας διπλωματικής είναι η συλλογή δεδομένων από τα κοινωνικά δίκτυα και η εφαρμογή αλγορίθμων Μη-Επιβλεπόμενης Μηχανικής Μάθησης για τη Συσταδοποίηση των Κειμένων που προέρχονται από αυτά. Για το λόγο αυτό αναπτύχθηκε σύστημα που συλλέγει, αναλύει και ομαδοποιεί τα δεδομένα με τέτοιο τρόπο ώστε να εξαχθούν χρήσιμες πληροφορίες προς τον τελικό χρήστη. Σε αυτό το πλαίσιο επιλέξαμε ως κοινωνικό δίκτυο ενδιαφέροντος της μελέτης το Twitter.

Το Twitter είναι μέσο κοινωνικής δικτύωσης που έχει τη μορφή μικροιστολογίου. Τα μικροϊστολόγια επιτρέπουν στους χρήστες να ανταλλάζουν σύντομες προτάσεις κειμένου, συνδέσμους εικόνων αλλά και βίντεο.

Η μελέτη των χαρακτηριστικών του περιεχομένου αυτών των δεδομένων και κυρίως κειμένων μπορεί να χαρακτηριστεί ως σημαντική για διάφορους λόγους, όπως είναι: η ζωντανή παρακολούθηση των συμβάντων, η πραγματοποίηση προτάσεων προς τον χρήστη ως προς την περιήγηση ή της δημιουργίας φιλίας με άλλους χρήστες, η ανάλυση των συναισθημάτων κλπ.

Για τη επίτευξη του στόχου μας έγινε χρήση αλγορίθμων *μηχανικής μάθησης* που ομαδοποιούν τα δεδομένα σε κατάλληλες συστάδες.

1.3 Οργάνωση Κειμένου

Η εργασία αυτή είναι οργανωμένη στα εξής κεφάλαια:

Στο κεφάλαιο 2 δίνεται το θεωρητικό υπόβαθρο των βασικών μεθόδων και τεχνολογιών που σχετίζονται με τη διπλωματική αυτή. Πιο συγκεκριμένα ορίζονται οι βασικές θεωρητικές έννοιες και περιγράφονται οι αλγόριθμοι που χρησιμοποιήθηκαν.

Στο κεφάλαιο 3 περιγράφονται οι σχετικές με το θέμα εργασίες.

Στο κεφάλαιο 4 παρουσιάζεται η ανάλυση και η σχεδίαση του συστήματος, δηλαδή η περιγραφή των υποσυστημάτων και των εφαρμογών του.

Στο κεφάλαιο 5 φαίνεται η περιγραφή της υλοποίησης του συστήματος, με ανάλυση των βασικών αλγορίθμων καθώς και λεπτομέρειες σχετικά με τις πλατφόρμες και τα προγραμματιστικά εργαλεία που χρησιμοποιήθηκαν.

Στο κεφάλαιο 6 παρουσιάζουμε την εργασία μας σε web εφαρμογή.

Τέλος, στο κεφάλαιο 7 δίνεται η συνεισφορά αυτής της διπλωματικής εργασίας καθώς και μελλοντικές επεκτάσεις.

2

Θεωρητικό Υπόβαθρο

2.1 Twitter

2.1.1 Περιγραφή

Το κοινωνικό δίκτυο Twitter ιδρύθηκε το 2006 στο Σαν Φρανσίσκο της Καλιφόρνια από τους Jack Dorsey, Evan Williams, Biz Stone. Αυτή τη στιγμή αριθμεί περισσότερους από 300 εκατομμύρια ενεργούς χρήστες (2015). Το Twitter είναι μια υπηρεσία κοινωνικής δικτύωσης και δημιουργίας μικρό-ιστολογίων (micro-blogging) που επιτρέπει στους χρήστες του να στέλνουν και να διαβάζουν μηνύματα γνωστά και ως tweets. Τα Tweets είναι δημοσιεύσεις με περιορισμένους χαρακτήρες (μέχρι 140) που εμφανίζονται στη σελίδα του προφίλ του χρήστη και μεταδίδονται στους ακόλουθους (Followers) αυτού. Αυτός ο περιορισμός είναι που ξεχωρίζει τα μικρό-ιστολόγια από τα τυπικά ιστολόγια (blogs) και οδηγεί τους χρήστες να περάσουν το μήνυμα τους σε όσο το δυνατόν πιο συμπυκνωμένη μορφή. Σε αυτά τα κείμενα του Twitter συνηθίζεται να αναφέρεται κάποιο νέο, μια είδηση,κάτι που συνέβη στο Χρηστη ή κάτι που αυτός σκέφτηκε, γενικώς μια πληροφορία. Πέρα των χαρακτήρων το tweet μπορεί να περιλαμβάνει φωτογραφίες, βίντεο, συνδέσμους και hashtags τα οποία έχουν ως σκοπό την κατηγοριοποίηση συζητήσεων και δημοσιεύσεων. Ένα hashtag, δηλαδή, μπορεί να προσδώσει νόημα σε μία δημοσίευση η οποία μπορεί υπό άλλες συνθήκες να μην είχε νόημα, διότι δείχνει σε τι αναφέρεται. Η βασική ιδέα του Twitter ως Κοινωνικό δίκτυο είναι ότι ένας Χρήστης(User) ακολουθεί κάποιους άλλους χρήστες για τους οποίους ενδιαφέρεται να δεί τι θα πουν και τι θα κάνουν. Αντίστοιχα οι άλλοι χρήστες πού ενδιαφέρονται για αυτόν μπορούν

με τη σειρά τους να τον ακολουθήσουν (Follow). Όσοι χρήστες έχουν επιλέξει να ακολουθήσουν κάποιον χρήστη θα ενημερωθούν για το εκάστοτε κείμενο που αυτός θα δημοσιεύσει και θα μπορούν να το διαβάσουν, να απαντήσουν (Reply) σε αυτό και να το αναδημοσιεύσουν (Retweet).Υπολογίζεται πως κάθε μέρα δημοσιεύονται περισσότερα από 58 εκατομμύρια tweets.

2.1.2 Γιατί Twitter;

Το περιορισμένο μέγεθος των Tweets περιορίζει το φάσμα χρήσης τους για επιστημονικούς σκοπούς. Παρόλους τους περιορισμούς, πολλοί ερευνητές κάνουν χρήση εργαλείων εξόρυξης κειμένου για ανάλυση των δημοσιεύσεων που γίνονται στο Twitter. Αυτό συμβαίνει για πολλούς λόγους, με κάποιους εκ των οποίων να φαίνονται παρακάτω:

1. Το Twitter είναι εξαιρετικά δημοφιλής πλατφόρμα για τα μέσα ενημέρωσης γι'αυτό και παρέχει περισσότερο χώρο για έρευνα.
2. Με το Twitter είναι εύκολο να ακολουθήσεις τη ροή μιας συζήτησης.
3. Το Twitter κάνει χρήση *Hashtags* που κάνουν πιο εύκολη τη συλλογή, ταξινόμηση, και την επέκταση των αναζητήσεων κατά τη συλλογή των δεδομένων.
4. Τα επιθυμητά δεδομένα μπορούν εύκολα να ανακτηθούν αφου σημαντικά συμβάντα, ειδήσεις και εκδηλώσεις στο Twitter τείνουν να επικεντρώνονται γύρω από κάποιο *Hashtag*.
5. Τα APIs του Twitter είναι πιο ανοιχτά και προσβάσιμα σε σύγκριση με τα αντίστοιχα που παρέχουν άλλες πλατφόρμες κοινωνικών μέσων μαζικής ενημέρωσης, γεγονός που καθιστά το Twitter ευνοϊκότερη επιλογή για τους προγραμματιστές που επιζητούν πρόσβαση σε δεδομένα. Αυτό αυξάνει, κατά συνέπεια, και τη διαθεσιμότητα των εργαλείων για τους ερευνητές και δημιουργεί μια κοινότητα επικοινωνίας που διευκολύνει κατά πολύ το έργο τους.
6. Πολλοί ερευνητές τυχαίνει να κάνουν και προσωπική χρήση του Twitter, νιώθοντας έτσι πιο άνετα διεξάγωντας έρευνα πάνω σε μια γνώριμη πλατφόρμα. **[TR]**
7. Στο Twitter πολλές φορές συγκεντρώνονται ποικίλες απόψεις πάνω σε κάποιο γεγονός της επικαιρότητας, γεγονός που κάνει την ομαδοποίηση τους σε Θέματα Συζήτησης ιδανική ώστε ο τελικός χρήστης να δεί το περιεχόμενο που συμπίπτει περισσότερο με τα δικά του ενδιαφέροντα ή προτιμήσεις.

2.1.3 Twitter APIs

REST APIs: Το *REST(Representational State Transfer)* αποτελεί την βασική αρχιτεκτονική αρχή του διαδικτύου. Το εντυπωσιακό είναι ότι οι *εξυπηρετούμενοι(clients)* και οι *εξυπηρετητές(servers)* μπορούν να αλληλεπιδρούν μεταξύ τους με πολύπλοκους τρόπους χωρίς ο εξυπηρετούμενος να γνωρίζει εκ των προτέρων για τον εξυπηρετητή και τους πόρους που αυτός διαθέτει. Ο βασικός περιορισμός είναι οτι εξυπηρετητής και εξυπηρετούμενος πρέπει να συμφωνούν στο μέσο που θα χρησιμοποιηθεί, το οποίο στην περίπτωση του διαδικτύου είναι η γλώσσα *HTML(HyperText Markup Language).*

Ένα API που υπακούει στις αρχές του *REST* δεν απαιτεί από τον εξυπηρετούμενο να γνωρίζει κάτι σχετικά με την δομή του API. Αντί αυτού, ο εξυπηρετητης πρέπει να παρέχει οποιαδήποτε πληροφορία χρειάζεται ο εξυπηρετούμενος για να αλληλεπιδράσει με την υπηρεσία.

Τα ***REST APIs*** **[TD]** του *Twitter* παρέχουν προγραμματιστική πρόσβαση για ανάγνωση και εγγραφή δεδομένων, από και προς το *Twitter*. Ο προγραμματιστής μπορεί να δημοσιεύσει ένα καινούργιο Tweet, να διαβάσει το προφίλ ενός χρήστη, δεδομένα σχετικά με τους ακόλουθους αυτου κλπ. Το *REST API* κάνει ταυτοποίηση των εφαρμογών και των χρηστών του *Twitter* μέσω του *OAuth,* το οποίο επιτρέπει ασφαλή αποστολή εγκεκριμένων αιτημάτων πρόσβασης στο API με τη χρήση διαπιστευτηρίων μοναδικών για κάθε χρήστη. Εφόσον τηρούνται οι προϋποθέσεις το API παρέχει απαντήσεις σε μορφή *JSON* προς το περιβάλλον που τις ζητά.

Για την διαξαγωγή ειδικών αναζητήσεων καθώς και για τη ανάγνωση πληροφοριών από το προφίλ του κάθε χρήστη απαιτείται και γίνεται χρήση του *REST API.*

Streaming API: Τα Streaming APIs **[TD]** παρέχουν στους προγραμματιστές απευθείας πρόσβαση με χαμηλή καθυστέρηση(latency) στην παγκόσμια ροή δεδομένων που προέρχονται από Tweets, εντός του Twitter. Μιά ορθή υλοποίηση ενός streaming εξυπηρετούμενου είναι η εμφάνιση μηνυμάτων που δείχνουν Tweets και διάφορα άλλα γεγονότα που συμβαίνουν, χωρίς την ύπαρξη του κόστους που σχετίζεται με την καταγραφή ενός τερματικού REST.

Τα κύρια "αντικείμενα" που λαμβάνουμε από τα APIs είναι τα Tweets, οι χρήστες(Users) και οι Οντότητες(Entities) όπως καθώς και τα επιμέρους χαρακτηριστικά αυτών.

Το Twitter παρόλο που δεν επιβάλλει κάποιο όριο ως προς το πλήθος των δεδομένων που παρέχει στους προγραμματιστές μέσω των APIs του, θέτει περιορισμούς σχετικά με τη λήψη των δεδομένων αυτών εντός σύντομων χρονικών διαστημάτων. Οι περιορισμοί αυτοί εφαρμόζονται ανα λογαριασμό χρήστη. Για παράδειγμα τα ερωτήματα αναζήτησης (Search) δεν μπορούν να ξεπερνούν τα 180 σε χρονικό παράθυρο 15 λεπτών.

2.2 Εξόρυξη Δεδομένων

2.2.1 Περιγραφή

Η *Εξόρυξη Δεδομένων(Data Mining)* είναι μία υπολογιστική διαδικασία που ανακαλύπτει μοτίβα σε μεγάλα σύνολα δεδομένων διασταυρώνοντας μεθόδους της τεχνητής νοημοσύνης, της μηχανικής μάθησης κλπ, καθώς και στατιστικά στοιχεία, με συστήματα βάσεων δεδομένων **[DMC]**. Ο γενικός στόχος της διαδικασίας εξόρυξης δεδομένων είναι η εξαγωγή πληροφοριών από ένα σύνολο δεδομένων και ταυτόχρονα η μετατροπή σε μια κατανοητή δομή για περαιτέρω χρήση **[DMC].**

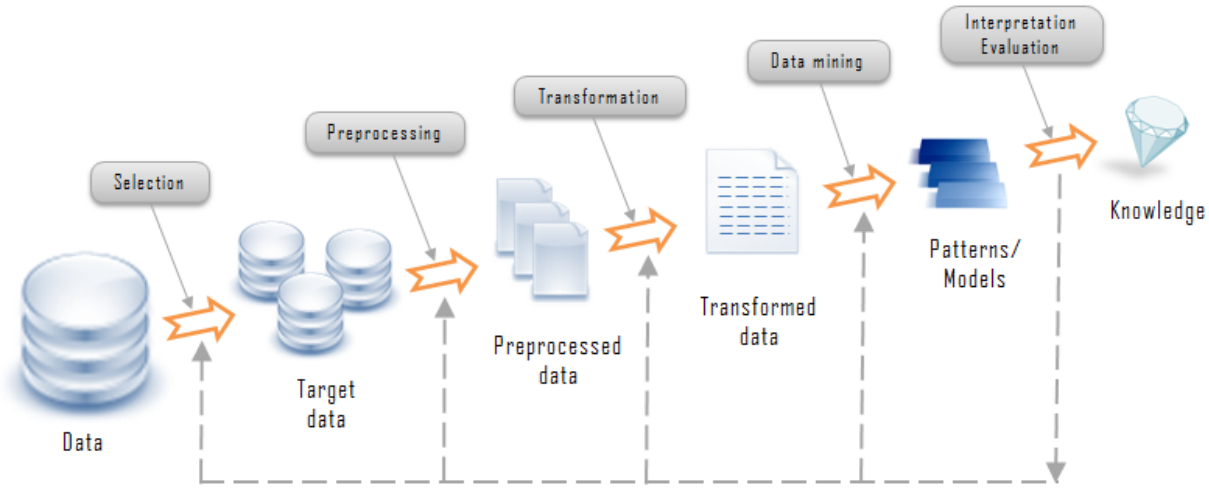
Η όλη διαδικασία της εξόρυξης δεδομένων και εξαγωγής πληροφοριών ωστόσο περιλαμβάνει αρκετά στάδια από τα οποία πρέπει να περάσει κανείς.

Αρχικά πρέπει να έχουμε ***πρόσβαση στη βάση*** που περιέχει τα δεδομένα του ενδιαφέροντος μας και μέσω των διαθέσιμων προγραμματιστικών εργαλείων να εφαρμόσουμε τα κατάλληλα "ερωτήματα" στη βάση, οι απαντήσεις των οποίων θα μας δώσουν το ***σύνολο δεδομένων*** όπου επιθυμούμε να εργαστούμε. Στην παρούσα εργασία δεχόμαστε δεδομένα από τη βάση του μέσου κοινωνικής δικτύωσης Twitter τα οποία επιλέγουμε ανάλογα ώστε να πληρούν τα κριτήρια μας μέσω των προγραμματιστικών εργαλείων που μας παρέχονται και θα δούμε στη συνέχεια.

Στη συνέχεια τα επιθυμητά δεδομένα αποθηκεύονται στη βάση με την οποία θα εργαστούμε. Τα δεδομένα αυτά παρόλο που είναι κατάλληλα μπορεί να μην είναι στην κατάλληλη μορφή για επεξεργασία. Ανάλογα την περίπτωση χρήσης λοιπόν καλούμαστε να κάνουμε κάποια ***προεπεξεργασία (preprocessing)*** για να τα φέρουμε στα μέτρα τόσο του προβλήματος όσο και των αλγορίθμων μηχανικής μάθησης που θα χρησιμοποιηθούν στη συνέχεια. Δεδομένου οτί οι οντότητες που θα παράγουν την πληροφορία στην περίπτωση του Twitter είναι τα Tweets, οι τεχνικές προεπεξεργασίας θα είναι τεχνικές που εφαρμόζονται κατά κόρον πάνω σε κείμενα όπως αφαίρεση συγκεκριμένων λέξεων, αφαίρεση τονισμού στις λέξεις, αλλά και άλλες τεχνικές σχετικές με την συχνότητα εμφάνισης των λέξεων-φράσεων.

Έπειτα, και δεδομένου ότι υπάρχουν τα κατάλληλα δεδομένα στην κατάλληλη μορφή από τα προηγούμενα στάδια γίνεται η **εξόρυξη** αυτών όπως αναφέρθηκε και παραπάνω, με μεθόδους συσταδοποίησης και κατάταξης που βασίζονται σε αλγόριθμους μηχανικής μάθησης.

Τέλος, άν όλα έχουν κυλήσει ομαλά, τα μοντέλα που χρησιμοποιήθηκαν έχουν εξάγει κάποια ***μοτίβα*** που αποτελούν και την πληροφορία ή οποία καλείται για ***ερμηνεία* και *αξιολόγηση*** από κάποιον αναλυτή ή πρόγραμμα ανάλυσης έτσι ώστε να πούμε ότι έχουμε ***γνώση***. Αυτή η γνώση ιδανικά θα παρουσιάζεται ***οπτικοποιημένη,*** κάνοντας έτσι πιό εύκολη την αναπαράσταση της σε άτομα που δεν γνωρίζουν σε απόλυτο βαθμό την περίπτωση. Στο παρακάτω διάγραμμα φαίνεται σε στάδια η διαδικασία εξόρυξης δεδομένων:



Εικόνα 2.1: Στάδια Εξόρυξης Δεδομένων

2.2.2 Γιατί Εξόρυξη Δεδομένων?

Τεράστιες ποσότητες πολύπλοκων δεδομένων δημιουργούνται από διάφορες πηγές οι οποίες διασυνδέονται με πολλούς τρόπους:

1. **Επιστημονικά δεδομένα** από διαφορετικά πεδία έρευνας όπως : αστρονομία, φυσική, βιολογία, οικονομία, γεωλογία, μετεωρολογία, ιατρική κλπ.
2. Τεράστιες **συλλογές κειμένων** που μπορούν να βρεθούν στο Διαδίκτυο και πιο συγκεκριμένα σε επιστημονικά άρθρα, σε δημοσιογραφικές πηγές, και σε δημοσιεύσεις στο Twitter, στο Facebook κλπ.

3. **Δεδομένα συμπεριφοράς** τα οποία λαμβάνονται κατά τη χρήση υπηρεσιών από κάποιον χρήστη με τη συγκατάθεση αυτού, είτε το γνωρίζει είτε όχι. Για παράδειγμα μπορούν να αντληθούν δεδομένα από τη χρήση των έξυπνων τηλεφώνων (smartphones), από ιστορικά αναζήτησης, από τη συμπεριφορά περιήγησης στον ιστό αλλά και από παρελθοντική ανάγνωση διαφημίσεων(ad).
4. **Δεδομένα συναλλαγών**, συνήθως ανώνυμα που μπορεί να έχουν πραγματοποιηθεί σε καταστήματα πώλησης ή και εντός πιστωτικών καρτών κλπ.

Προφανώς όλα τα παραπάνω μπορούν να συνδυαστούν. Ένα κοινωνικό δίκτυο όπως το Twitter για παράδειγμα μπορεί να παρέχει πληροφορίες από κείμενα, δεδομένα διαδικτυακής συμπεριφοράς, συναναστροφής με διαφημίσεις ακόμα και γεωχωρική πληροφορία. Αυτά τα δεδομένα καλούμαστε εμείς να αναλύσουμε και να εξάγουμε γνώση.

2.3 Βάση Δεδομένων

2.3.1 Εισαγωγή

Σε προηγούμενη ενότητα είδαμε πως μπορούμε να κάνουμε εξόρυξη δεδομένων από το Twitter, εδώ θα δούμε πως επιτυγχάνεται η αποθήκευση αυτών σε κάποια βάση *δεδομένων* με κατάλληλο τρόπο ώστε να είναι έτοιμα για επεξεργασία. Η βάση δεδομένων είναι μια συλλογή δεδομένων οργανωμένη με τέτοιο τρόπο ώστε να καθιστά εύκολη την πρόσβαση, την διαχείριση και την ενημέρωση των δεδομένων αυτών.

Οι βάσεις δεδομένων μπορεί να είναι είτε Σχεσιακές (Relational) είτε Μη-Σχεσιακές (Non-Relational). Ο πιο διαδεδομένος και ευρέως χρησιμοποιούμενος τύπος βάσεων δεδομένων απ τη δεκαετία του 80 μέχρι και σήμερα είναι οι σχεσιακές (Relational) ή αλλιώς SQL βάσεις, δηλαδή βάσεις που μπορούν να διαχειριστούν μέσω της γλώσσας ερωταπαντήσεων SQL (Structured Query Language).

Οι βάσεις που ακολουθούν το σχεσιακό μοντέλο (RDBMS) αποθηκεύουν άκρως δομημένα δεδομένα σε πίνακες με στήλες προκαθορισμένου τύπου και πολλαπλές εγγραφές που

περιλαμβάνουν δεδομένα ίδιου τύπου, εν μέρει χάρη στην ακαμψία της οργάνωσης τους, απαιτούν διατήρηση της αυστηρής δόμησης των δεδομένων κατά την ανάπτυξη εφαρμογών.

Στις σχεσιακές βάσεις δεδομένων η οργάνωση γίνεται σε μορφή πινάκων οι οποίοι έχουν για χαρακτηριστικό ένα πρωτεύων κλειδί και δευτερεύοντα κλειδιά τα οποία φανερώνουν τις σχέσεις μεταξύ αυτών.

Οι ενώσεις μεταξύ πινάκων υπολογίζονται κατά τα ερωτήματα (Queries) στη βάση όπου γίνεται το ταίριασμα μεταξύ πρωτευόντων και δευτερευόντων κλειδιών. Αυτό, δεδομένου ότι θα υπάρχει μεγάλος αριθμός εγγραφών απαιτούν ενέργειες οι οποίες θα έχουν εκθετικό κόστος τόσο σε υπολογιστική ισχύ όσο και σε μνήμη. Το κόστος για σύνδεση πινάκων αυξάνεται περαιτέρω αν έχουμε και ύπαρξη σχέσεων "πολλά-προς-πολλά", διότι πρέπει να γίνει χρήση ενδιάμεσων πινάκων που θα έχουν τα ξένα κλειδιά των προς σύνδεση πινάκων. Κατά το παρελθόν παρόλο που σε κάποιες περιπτώσεις η χρήση αυτού του αυστηρού μοντέλου δεν ήταν καθόλου ενδεδειγμένη, η έλλειψη βιώσιμων εναλλακτικών λύσεων και η εμμονή με το σχεσιακό μοντέλο, απέτρεπε εναλλακτικά μοντέλα βάσεων δεδομένων να σπάσουν την επικρατούσα τάση.

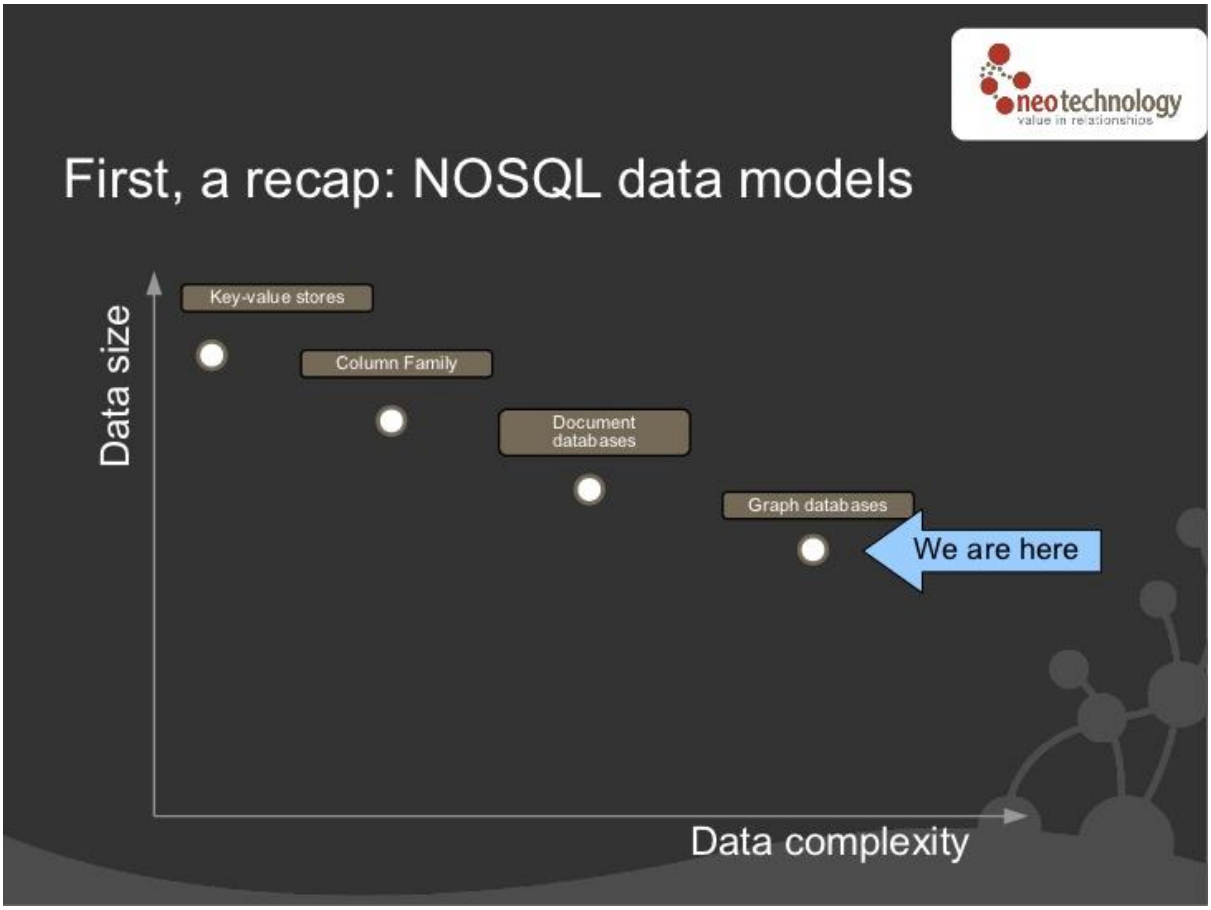
Τα τελευταία χρόνια ωστόσο έχει παρατηρηθεί μια σημαντική άνοδος στη δημοτικότητα μιας άλλης οικογένειας βάσεων δεδομένων, γνωστών και ως Μη-Σχεσιακές ή αλλιώς ***NoSQL*** (Not only SQL). Ο όρος NoSQL χαρακτηρίζει περισσότερο το ότι αυτές οι βάσεις δεν είναι SQL-κεντρικές σχεσιακές βάσεις δεδομένων, παρά τι στην ουσία είναι. Το γεγονός οτί την τελευταία δεκαετία τα δεδομένα που καλούμαστε να χειριστούμε είναι σημαντικά μεγαλύτερα σε μέγεθος, αλλάζουν γρηγορότερα και έχουν μεγάλυτερη ποικιλία στη δομή τους, καθιστά τα παραδοσιακά RDBMS συστήματα δύσκολο να ανταποκριθούν. Αυτές είναι και οι προκλήσεις που αντιμετωπίζουν οι Μη-Σχεσιακές NoSQL βάσεις δεδομένων.

Αντίθετα με τις SQL που παρουσιάζουν δυσκολία κλιμάκωσης σε πολλά δεδομένα κατά τα ερωτήματα οι NoSQL βάσεις δεδομένων υιοθετούν διαφορετικές προσεγγίσεις για να αντιμετωπίσουν τον μεγάλο όγκο δεδομένων και να αποφύγουν τις πολλές ενώσεις.

Πιο συγκεκριμένα όπως γνωρίζουμε, οι σχεσιακές βάσεις δεδομένων έχουν αυστηρά καθορισμένο σχήμα και η αλλαγή αυτού εφόσον έχουν μπεί δεδομένα στην βάση, απαιτεί μεγάλο λειτουργικό κόστος και είναι μια διαδικασία που συχνά αποφεύγεται. Αντίθετα οι NoSQL βάσεις δεδομένων έχουν δυναμικό σχήμα βάσης. Έτσι όσον αφορά τον υψηλό ρυθμό αλλαγής των δεδομένων τόσο ως προς τον αριθμό όσον και ως προς την δομή υπάρχει μεγαλύτερη ευελιξία και το κόστος μειώνεται. Τα δεδομένα μπορούν να προστίθονται με ευκολία στη πορεία ανεξαρτήτως τύπου και δεν είναι απαραίτητο κάθε γραμμή να περιέχει εγγραφή για κάθε στήλη όπως ακολουθείται στις σχεσιακές βάσεις.

Ακόμα η οριζόντια κλιμάκωση που χρησιμοποιούν οι NoSQL βάσεις δεδομένων σημαίνει μείωση του φορτίου υπολογισμών μέσω της αύξησης του αριθμού των εξυπηρετητών(Servers). Αυτό μπορεί να επιτευχθεί είτε με την προσθήκη εξυπηρετητών φθηνού υλικού είτε με τη δημιουργία στιγμιοτύπων στο νέφος (Cloud). Με αυτόν τον τρόπο επιτυγχάνεται μια κλιμάκωση πιο αποδοτική από οικονομικής άποψεως σε σχέση με τις κατακόρυφα κλιμακούμενες SQL βάσεις δεδομένων οι οποίες για να το πετύχουν καλούνται να αυξήσουν την υπολογιστική ισχύ του Hardware του εξυπηρετητή.

Ενδεικτικά αναφέρουμε μερικές επικρατούσες βάσεις δεδομένων τόσο στο σχεσιακό SQL μοντέλο: *MySQL, Oracle DB, DB2, SQLite, PostgreSQL* και *Microsoft-SQL*, όσο και στο μη-σχεσιακό NoSQL μοντέλο: *MongoDB, CouchDB, Redis, RavenDB, Cassandra, Hbase* και *Neo4j*.



Εικόνα 2.2: Σύγκριση NOSQL βάσεων δεδομένων

[**NOSQLDM**]

Σε κάθε περίπτωση πρέπει η βάση δεδομένων που θα επιλέξει κάποιος να ταιριάζει με τη φύση των δεδομένων που πρόκειται να αποθηκευτούν σε αυτή. Ένα κοινωνικό δίκτυο είναι ένα καλό

παράδειγμα ενός πυκνά συνδεδεμένου, ποικίλης δομής δικτύου. Στην παρούσα εργασία τα δεδομένα προέρχονται από το κοινωνικό δίκτυο Twitter και ο ιδανικότερος τρόπος αναπαράστασης των πυκνών μεταξύ τους σχέσεων είναι ένας ***γράφος***. Οι γράφοι είναι εξαιρετικά χρήσιμοι στην κατανόηση δεδομένων ποικίλων τύπων πάνω σε πεδία όπως οι επιστήμες, οι επιχειρήσεις και η πολιτική. Ο πραγματικός κόσμος είναι πλούσιος και αλληλένδετος. Ομοιόμορφος και περιορισμένος σε κάποια κομμάτια του, εξειδικευμένος και ακανόνιστος σε κάποια άλλα. Τα δεδομένα σε μια βάση δεδομένων τέτοιου τύπου αποθηκεύονται σε δομές γράφου με κόμβους (οντότητες), ιδιότητες (πληροφορίες σχετικά με τις οντότητες) και ακμές (συνδέσεις μεταξύ των οντοτήτων). Η ευελιξία του μοντέλου γραφήματος επιτρέπει να προστίθονται νέοι κόμβοι και νέες σχέσεις, χωρίς να διακυβεύεται το υπάρχον δίκτυο ή να απαιτείται προσαρμογή των δεδομένων. Έτσι τα αρχικά δεδομένα και ο σκοπός τους παραμένουν ανέπαφα. Ο γράφος μπορεί να παρέχει εικόνα τόσο για την απευθείας σχέση μεταξύ στοιχείων ενός κοινωνικού δικτύου ανάλογα τον τύπο της, όσο και για τα φυσικά μονοπάτια που σχηματίζονται από μια σειρά σχέσεων μέσα στο δίκτυο. Έχοντας αυτά κατά νου και λαμβάνοντας υπόψη την συνεχή ανάπτυξη των μη-σχεσιακών μοντέλων, επιλέξαμε μια βάση δεδομένων γράφου, καθώς θεωρήσαμε ότι μπορεί να μοντελοποιήσει καλύτερα τον τομέα του προβλήματος μας. Πιο συγκεκριμένα καταλήξαμε στην Neo4j την πιο διαδεδομένη βάση δεδομένων γράφου κατά τη στιγμή εκπόνησης της διπλωματικής.

2.3.2 Neo4j

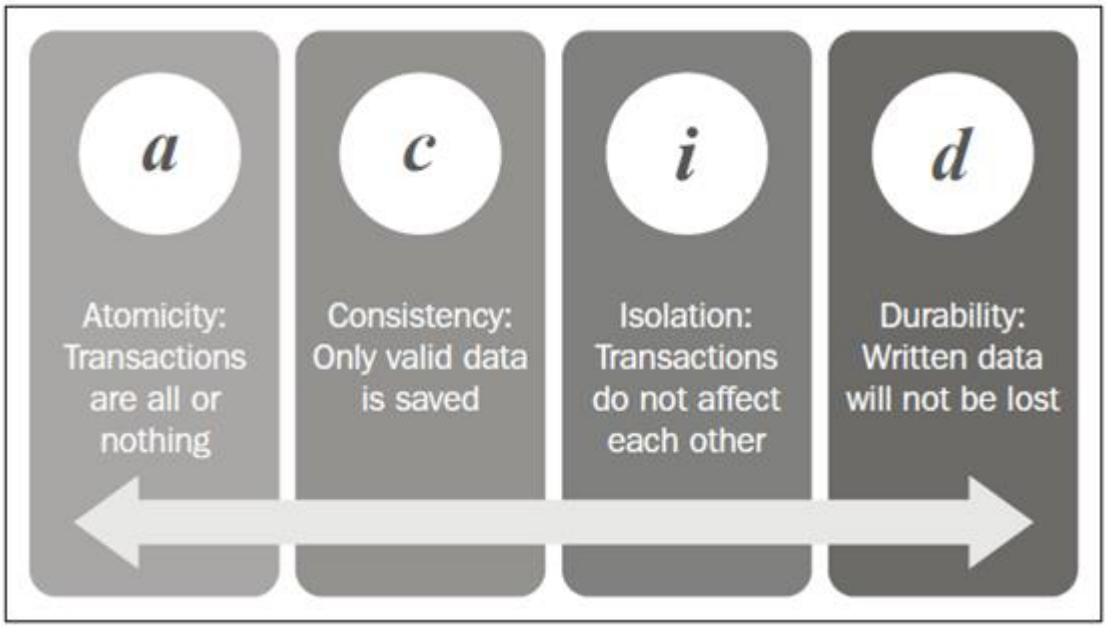
Η *Neo4j* αναπτύχθηκε από την Neo Technology, Inc., με έδρα το San Francisco Bay Area των ΗΠΑ και το Malmö της Σουηδίας. Πρόκειται για μια open-source βάση δεδομένων γράφου υλοποιημένη σε Java και προσβάσιμη από λογισμικό γραμμένο σε διάφορες γλώσσες χρησιμοποιώντας τη γλώσσα ερωτημάτων *Cypher* μέσω συναλλακτικού HTTP τερματικού.

Η Neo4j αποθηκεύει τα δεδομένα ως *Κορυφές(Vertices)* και *Ακμές(Edges)* ή αλλιώς σε ορολογία Neo4j, ως *Κόμβους(Nodes)* και *Σχέσεις(Relationships)*. Οι Χρήστες (Users) για παράδειγμα του κοινωνικού δικτύου θα αναπαριστώνται ως κόμβοι και οι συνδέσεις ακολουθίας (Follow) ως σχέσεις μεταξύ κόμβων τύπου χρηστών.

Όπως πολλά άλλα *έργα ανοικτού κώδικα (open-source)* και ειδικότερα πολλά NoSQL Συστήματα Διαχείρισης Βάσης Δεδομένων ανοικτού κώδικα, η Neo4j δημιουργήθηκε για

συγκεκριμένο σκοπό. Ο σκοπός είναι η εύρεση μιας δραστικής νέας προσέγγισης για αντιμετώπιση κάποιων προβλημάτων που η επίλυση τους με τα παραδοσιακά μέχρι τότε τεχνολογικά μέσα, παρουσίαζε δυσκολίες. Ολόκληρη η δομή της Neo4j, συμπεριλαμβανόμενων και των συστατικών χαμηλού-επιπέδου (low-level), όπως για παράδειγμα η διάταξη των δυαδικών αρχείων της, είναι φτιαγμένη για να συναναστρέφεται με δεδομένα τύπου γράφου. Αυτό είναι για πολλούς λόγους σημαντικό, αφού αποτελεί τη βάση για τις πολλές βελτιώσεις από άποψη ταχύτητας και όχι μόνο που προσφέρει η Neo4j σε σχέση με άλλα συστήματα.**[LN]**

Ένα χαρακτηριστικό της Neo4j που αποτελεί σημαντικό της προτέρημα είναι ότι είναι μια ***ACID*-συμβατή** βάση δεδομένων. ACID είναι το ακρωνύμιο που αντιπροσωπεύει τους 4 στόχους χαρακτηριστικών που πολλά Συστήματα Διαχείρισης της Βάσης Δεδομένων (DBMS) επιδιώκουν να διαθέτουν. Αυτοί οι στόχοι φαίνονται στο ακόλουθο διάγραμμα:



Εικόνα 2.3: Συστατικά ACID-συμβατών βάσεων

[**LN**]

Τα χαρακτηριστικά περιγράφονται πιο αναλυτικά παρακάτω:

Ατομικότητα (Atomicity): Αυτό σημαίνει ότι οι αλλαγές που συμβαίνουν στη βάση πρέπει να γίνονται με βάση τον κανόνα *"όλα η τίποτα"* (*all or nothing rule*). Οι συναλλαγές λέμε ότι είναι "ατομικές" όταν η αποτυχία ενός κομματιού της συναλλαγής έχει ως αποτέλεσμα την επαναφορά (rollback) όλης της συναλλαγής.

Συνέπεια(Consistency): Σημαίνει μόνο συνεπή δεδομένα επιτρέπεται να εισαχθούν στη βάση δεδομένων. Σε σχεσιακή ορολογία, αυτό θα σήμαινε ότι το σχήμα της βάσης πρέπει να εφαρμόζεται και να διατηρείται σε κάθε περίπτωση. Η κύρια απαίτηση συνέπειας για τη Neo4j είναι ότι οι σχέσεις γράφου πρέπει να έχουν ένα αρχικό και ένα τελικό κόμβο, δηλαδή να μην υπάρχει σχέση "στον αέρα". Πάραυτα, οι κανόνες συνέπειας στη Neo4j είναι πολύ πιο χαλαροί σχέση με τους αντίστοιχους σχεσιακούς, αφού αυτή εφαρμόζει το πρότυπο του "προαιρετικού" σχήματος βάσης.

Απομόνωση(Isolation): Προϋποθέτει ότι τα στιγμιότυπα πολλαπλών συναλλαγών που εκτελούνται παράλληλα στην ίδια βάση δεδομένων δεν επηρεάζονται μεταξύ τους. Κάθε συναλλαγή πρέπει να ακολουθεί την προγραμματισμένη ροή της, ανεξαρτήτως τι συμβαίνει στο σύστημα την ίδια στιγμή. Μία από τις σημαντικές εφαρμογές αυτού είναι η περίπτωση όπου μια συναλλαγή γράφει στη βάση δεδομένων και κάποια άλλη ταυτόχρονα διαβάζει από τη βάση δεδομένων. Σε μια απομονωμένη βάση, η συναλλαγή ανάγνωσης δεν μπορεί να ξέρει για τη συναλλαγή εγγραφής που λαμβάνει χώρα παράλληλα, μέχρι η συναλλαγή εγγραφής να ολοκληρώσει την εκτέλεση και να καταχωρηθεί πλήρως. Όσο η λειτουργία εγγραφής δεν έχει καταχωρηθεί, η λειτουργία ανάγνωσης επιτρέπεται να αλληλεπιδράσει μόνο με τα "παλιά» δεδομένα.

Αντοχή(Durability): Αυτό βασικά σημαίνει ότι οι καταχωρημένες συναλλαγές δεν μπορούν απλά να εξαφανιστούν και να χαθούν. Η υποχρεωτική εγγραφή των μητρώων καταχώρησης συναλλαγών - ακόμα και αν αυτές δεν έχουν ανανεωθεί ακόμα - στο δίσκο διασφαλίζει την ποιότητα στα περισσότερα συστήματα βάσεων δεδομένων όπως επίσης και στη Neo4j.

Η σύνοψη όλων αυτών είναι πιθανόν ότι η Neo4j έχει σχεδιαστεί από το μηδέν έτσι ώστε να αποτελέσει μια βάση δεδομένων *πολλαπλών χρήσεων*. Κατέχει πολλές από τις αρετές των παραδοσιακών Σχεσιακών Συστημάτων Βάσης Δεδομένων που ξέρουμε σήμερα, με τη διαφορά ότι χρησιμοποιεί ένα σημαντικά διαφορετικό μοντέλο δεδομένων το οποίο ταιριάζει κατάλληλα σε πυκνά συνδεδεμένες περιπτώσεις χρήσης.

Τα χαρακτηριστικά που αναφέραμε βοηθούν στα συστήματα όπου πραγματικά χρειάζεται η επιστροφή δεδομένων από το σύστημα σε ένα Online περιβάλλον. Αυτό σημαίνει ότι τα ερωτήματα που θέλουμε να "ρωτήσουμε" τη βάση πρέπει να μπορούν να απαντηθούν στο χρονικό διάστημα μεταξύ ενός αιτήματος στον ιστό και της απόκρισης του. Με άλλα λόγια, σε χιλιοστά του δευτερολέπτου. Για αυτό το λόγο η Neo4j είναι φτιαγμένη για Online *επεξεργασία συναλλαγών* ***(Online Transaction Processing - OLTP).***

Όπως είδαμε και προηγουμένως οι SQL-βάσεις παρουσιάζουν σημαντικά ελαττώματα όταν έχουν να κάνουν με πολύπλοκα μοντέλα δεδομένων. Οι υπολογισμοί για ερωτήματα τα οποία γίνονται πάνω σε μεγάλα σύνολα δεδομένων ή εμπλέκουν παραπάνω των 2 πινάκων παίρνουν αρκετό χρόνο. Αντίθετα στη Neo4j όπως και γενικότερα στις βάσεις δεδομένου γράφου δεν υπάρχουν αυτά τα προβλήματα, διότι χρησιμοποιούν ισχυρές μαθηματικές έννοιες από τη θεωρία γραφημάτων επιτρέποντας στις ***πράξεις ενώσεων(join)*** να είναι αποτελεσματικά προϋπολογισμένες. Ο κύριος λόγος για τις δυνατότητες πρόβλεψης της Neo4j είναι η περιορισμένη φύση της διάσχισης γράφου. Ανεξαρτήτως του αριθμού κόμβων και σχέσεων στο γράφο συνεπώς και του μεγέθους της βάσης, η διάσχιση θα περάσει μόνο από αυτούς που συνδέονται στον αρχικό κόμβο,σύμφωνα με τους κανόνες διάσχισης. Για αυτό το λόγο τα ερωτήματα ενώσεων δεδομένων στη Neo4j είναι εξαιρετικά απλά, αποτελεσματικά και γρήγορα.

Στο ακόκλουθο διάγραμμα φαίνεται η χρονική σύγκριση μεταξύ της *Neo4j* και *RDBMS* για ποικίλα βάθη διάσχισης:

Depth	RDBMS execution time(s)	Neo4j execution time(s)	Records returned
2	0.016	0.01	~2500
3	30.267	0.168	~110,000
4	1543.505	1.359	~600,000
5	Unfinished	2.132	~800,000

Εικόνα 2.4: Η εύρεση φίλων σε διάφορες αποστάσεις για μια σχεσιακή βάση δεδομένων σε σχέση με την Neo4j

[**GDB**]

Όπως φαίνεται η αύξηση των δεδομένων κατά χιλιάδες δεν επηρεάζει σημαντικά την απόδοση της Neo4j. Η διάσχιση δεν γίνεται πιο αργή όσο αυξάνεται το βάθος και αυτή η μικρή αύξηση που βλέπουμε οφείλεται στον αριθμό των αποτελεσμάτων που επιστρέφονται.

Οι γράφοι όπως ξέρουμε από τη θεωρία γραφημάτων μπορεί να είναι διαφόρων σχημάτων και μεγεθών. Η Neo4j είναι *βάση δεδομένων γράφου με ιδιότητες (property graph database)* και διαθέτει συγκεκριμένο τύπο **δομής δεδομένων** ο οποίος είναι αρκετά ευέλικτος για να υποστηρίζει την αστάθεια συνόλων δεδομένων που προέρχονται από τον πραγματικό κόσμο

Το **μοντέλο δεδομένων** γράφου μας δίνει 4 θεμελιώδεις μονάδες για να δομήσουμε και να αποθηκεύσουμε τα δεδομένα μας, τις οποίες βλέπουμε παρακάτω:

Κόμβοι(Nodes): Τυπικά χρησιμοποιούνται για την αποθήκευση των πληροφοριών μιας οντότητας. Στην περίπτωση μας που ασχολούμαστε με το δίκτυο του Twitter, αυτοί είναι οι Users (*Χρήστες),* τα *Tweets* , τα *Hashtags* και τα *Links (Σύνδεσμοι)*.

Σχέσεις(Relationships): Χρησιμοποιούνται για να συνδέουν τους κόμβους μεταξύ τους αποκλειστικά και ως εκ τούτου παρέχουν το μέσο δόμησης των οντοτήτων. Θα λέγαμε ότι οι σχέσεις είναι ισοδύναμες με ρητά αποθηκευμένες και προϋπολογσμένες λειτουργίες ένωσης (JOIN) σε κάποιο σχεσιακό σύστημα. Όπως είδαμε και πρίν τα JOINs δεν είναι πια χρονοβόρα, αλλά τόσο απλα όσο μια διάσχιση μίας σχέσης που συνδέει 2 κόμβους. Οι σχέσεις στη Neo4j πάντα έχουν, έναν αρχικό κόμβο, ένα τελικό κόμβο και μια κατεύθυνση. Μπορεί να είναι αυτοσυσχέτιστες (looping) αλλά ποτέ "αιωρούμενες", δηλαδή χωρίς αρχικό ή τελικό κόμβο.
Στο παράδειγμα μας οι σχέσεις που έχουμε είναι οι: *'Follows' (Ακολούθηση), 'Posts' (Δημοσίευση), 'Mentions' (Επισήμανση), 'Tags' (Πρόσθεση ετικέτας), 'Contains' (Συμπερίληψη) , 'Retweet of' (Αναδημοσίευση)* και *'Reply to' (Απάντηση).*

Ιδιότητες(Properties): Τόσο οι Κόμβοι όσο και οι Σχέσεις διαθέτουν ιδιότητες, οι οποίες ουσιαστικά δημιουργούν αποτελεσματικά *ζευγάρια κλειδιού/τιμής (key/value pairs)*. Τα "keys" των ιδιοτήτων είναι συμβολοσειρές (strings) και τα "values" είναι αυθαίρετοι τύποι δεδομένων. Το να διαθέτουν οι κόμβοι ιδιότητες είναι τελείως φυσιολογικό. Όπως ακριβώς μια εγγραφή σε σχεσιακή βάση μπορεί να έχει 1 ή περισσότερες ιδιότητες έτσι και ένας Κόμβος στους γράφους μπορεί να έχει 1 ή περισσότερες ιδιότητες.Αντίθετα το να διαθέτουν ιδιότητες οι Σχέσεις όπως γίνεται στη Neo4j, είναι λιγότερο συνηθισμένο. Η χρήση ιδιοτήτων στις Σχέσεις γίνεται για να τονιστεί η δύναμη ή η ποιότητα μιας σχέσης και μπορεί να χρησιμοποιηθεί σε ερωτήματα/διασχίσεις για την αποτίμηση των μοτίβων πού αναζητάμε. Αυτό προσδίδει σημασιολογία στις σχέσεις και επιτρέπει την εφαρμογή διαφόρων αλγορίθμων από τη θεωρία

γραφημάτων (πχ. Dijkstra). Ένα παράδειγμα θα μπορούσε να είναι Σχέσεις που διαθέτουν βάρη ως ιδιότητα προσδίδοντας ισχύ σε αυτές ανάλογα με την τιμή του.

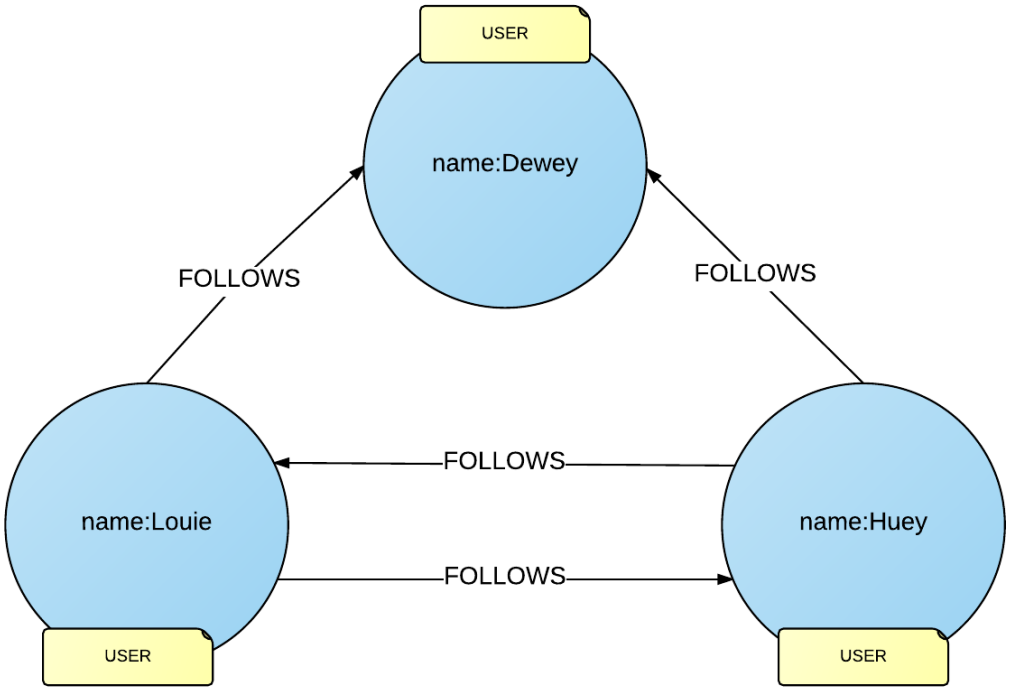
Ετικέτες(Labels): Οι ετικέτες είναι μια θεμελιώδης δομή του μοντέλου δεδομένων και προστέθηκε στη Neo4j από την έκδοση 2.0 (2013) και μετά. Αποτελούν ένο μέσο γρήγορης και αποτελεσματικής δημιουργίας υπογράφων. Η ανάθεση ετικετών στους κόμβους, κάνει το το μοντέλο δεδομένων των περισσοτέρων χρηστών αρκετά απλούστερο. Δεν χρειάζεται πλέον να γίνεται χρήση ιδιότητας που να αποθηκεύει τον *τύπο* της οντότητας στους κόμβους. Πλέον αυτό γίνεται αυτόματα από τη Neo4j, κάτι το οποίο είναι μεγάλη παρακαταθήκη, για τώρα αλλά και για το μέλλον. Οι ετικέτες χρησιμοποιούνται για τον *περιορισμό των ερωτημάτων*, για *δεικτιοδότηση* και για τον *ορισμό περιορισμών*.

Τέλος ένα από τα κύρια χαρακτηριστικά της βάσης Neo4j σήμερα, είναι η **γλώσσα ερωτημάτων** της, η **Cypher**. Η Cypher είναι μία εκφραστική γλώσσα ταιριάσματος προτύπων πού κάνει τα συστήματα διαχείρισης βάσης δεδομένων κατανοητά και εφαρμόσιμα για κάθε χρήστη της βάσης, ακόμα και για αυτούς που έχουν λιγότερες τεχνικές γνώσεις.

2.3.3 Ερωτήματα σε Cypher

Όπως αναφέραμε και πρίν η Cypher είναι μία *εκφραστική* αλλά και *συμπαγής* γλώσσα ερωτημάτων πάνω σε βάσεις δεδομένων γράφου. Συγκεκριμένα για την Neo4j, η τάση της να παρουσιάζει τα γραφήματα ως *διαγράμματα* την καθιστά ιδανική για περιγραφή γραφημάτων προγραμματιστικά. Για το λόγο αυτό, κάναμε χρήση της Cypher στην παρούσα εργασία.

Η Cypher έχει σχεδιαστεί ώστε να γίνεται εύκολα *αναγνώσιμη* και *κατανοητή από* προγραμματιστές. Η ευκολία της χρήσης της προκύπτει από το γεγονός οτι είναι σε συμφωνία με τον διαισθητικό τρόπο των ανθρώπων να περιγράφουν τους γράφους χρησιμοποιώντας διαγράμματα. Η Cypher επιτρέπει σε κάποιον χρήστη ή εφαρμογή να κάνει ερωτήματα στη βάση για να βρεί δεδομένα που ταιριάζουν με το συγκεκριμένο μοτίβο που περιγράφει.



Εικόνα 2.5: Παράδειγμα μοτίβου αλλησοσύνδεσης οντοτήτων στη βάση

Το παραπάνω μοτίβο περιγράφει τρείς χρήστες του Twitter απ τους οποίους οι 2 (Louie,Huey) αλληλο-ακολουθούνται και επίσης ακολουθούν έναν τρίτο χρήστη (Dewey) . Η αντίστοιχη αναπαράσταση του ερωτήματος σε Cypher είναι::

- (dewey)<-[:FOLLOWS]-(huey)-[:FOLLOWS]->(louie)-[:FOLLOWS]->(dewey)

Αυτό το μοτίβο περιγράφει το μονοπάτι που συνδέει ένα κόμβο που αποκαλούμε *huey* σε δύο άλλους κόμβους που αποκαλούμε *louie* και *dewey* αλλά και τους άλλους κόμβους μεταξύ τους. Τα τρία ονόματα που επιλέξαμε είναι αναγνωριστικά. Τα αναγνωριστικά μας επιτρέπουν να αναφερθούμε στον ίδιο κόμβο παραπάνω από μία φορά σε ένα μοτίβο βοηθώντας έτσι στην περιγραφή ενός γραφήματος δύο διαστάσεων μέσω μιας μονοδιάστατης γλώσσας ερωτημάτων. Τα μοτίβα σε Cypher ακολουθούν πολύ φυσικά τον τρόπο με τον οποίο θα σχεδίαζε κάποιος γράφους σε έναν πίνακα.

Όπως σε κάθε γλώσσα ερωτημάτων βάσης δεδομένων, έτσι και στην Cypher υπάρχουν μερικές λειτουργικές λέξεις που έχουν ιδιαίτερη σημασία στη σύνθεση του κάθε ερωτήματος.

Μερικές απαραίτητες **λέξεις κλειδιά που χρησιμοποιεί η Cypher** εξηγούνται παρακάτω:

START

Με το START μπορούν να οριστούν για το ερώτημα ένα ή περισσότερα σημεία αρχής (κόμβους ή ακμές) στον γράφο. Σημεία δηλαδή του γράφου από όπου θα ξεκινήσουμε για να απαντήσουμε στο ερώτημα. Πρέπει λοιπόν να μπορούμε να φτάσουμε σε αυτά τα σημεία πολύ γρήγορα. Οπότε χρησιμοποιούμε κάποιο ευρετήριο ή το id των κόμβων και των ακμών από όπου θέλουμε να ξεκινήσουμε την ερώτηση μας. Δεδομένου ότι η Neo4j χρησιμοποιεί κανονική αποθήκευση γράφου (native graph storage), συνεπώς μπορούμε σε O(1) να προσπελάσουμε τους ζητούμενους κόμβους και ακμές με τη χρήση του id τους.

Παράδειγμα: START n=node:User(Name = “Omiros”)

RETURN n

MATCH

Ο όρος MATCH αποτελεί τον πυρήνα τον περισσότερων ερωτημάτων σε Cypher. Ουσιαστικά καθορίζει το μοτίβο του γράφου, το οποίο θέλουμε να αναζητήσουμε “ζωγραφίζοντας” τα ζητούμενα. Περικλείουμε τους κόμβους μέσα σε παρενθέσεις και εκφράζουμε τις σχέσεις μέσω 2 παυλών σε συνδυασμό με το σύμβολο του μεγαλύτερου (-->) ή του μικρότερου (<--) ανάλογα την κατεύθυνση της σχέσης. Ανάμεσα στις παύλες, εντός αγκυλών και ύστερα από τον ειδικό χαρακτήρα “:” τοποθετούμε το όνομα του τύπου της σχέσης. Παρομοίως τοποθετούμε το όνομα τύπου ενός κόμβου μετα τον ειδικό χαρακτήρα “:”. Με κεφαλαία γράφουμε τις λέξεις κλειδιά, ενώ με μικρά τις μεταβλητές. Στις μεταβλητές ουσιαστικά δεσμεύουμε κόμβους ή σχέσεις του γράφου, ώστε να ικανοποιείται το μοτίβο.

Παράδειγμα: MATCH (me:User)-[:FOLLOWS]->(friend)

WHERE

Δεδομένης της ύπαρξης του MATCH, ο όρος WHERE ορίζει κάποιες συνθήκες που πρέπει να ικανοποιούνται ώστε, περιορίζοντας έτσι το τμήμα του γράφου που θα λάβουμε ως απάντηση στο αρχικό μοτίβο που σχηματίστηκε από το MATCH.

Παράδειγμα: WHERE me.Name = "Omiros" AND me.Location = “Athens”

RETURN

Ορίζει ποιοι κόμβοι, σχέσεις, χαρακτηριστικά ή και συνδυασμός στοιχείων που ικανοποιούν το μοτίβο θα επιστραφούν στο χρήστη ως απάντηση.

Παράδειγμα: RETURN me.Name, collect(friend), count(*) as Friends

CREATE

Δημιουργεί κόμβους και σχέσεις με τα χαρακτηριστικά που θα ορίσουμε εμείς.

Παράδειγμα: CREATE (p:User), (p)-[:FOLLOWS {Since: '2/2/2016'}]-> (me:User {Name:"Omiros"})

SET, REMOVE

Ενημερώνει τις τιμές των ιδιοτήτων κάποιας σχέσης ή κάποιου κόμβου.

Παράδειγμα: SET me.Age = 42

REMOVE me.Location

MERGE

Αντίστοιχα με το CREATE δημιουργεί κόμβους και σχέσεις με τη διαφορά οτι εξασφαλίζει σε περίπτωση που το συγκεκριμένο στοιχείο υπάρχει ήδη στο γράφο ότι δεν θα υπάρχουν διπλές εγγραφές. Στην περίπτωση αυτές απλά ανανεώνει όποιες ιδιότητες του πιθανόν να αλλάξαμε.

Παράδειγμα: MERGE (me:User {Name:"Omiros"})

ON MATCH me SET me.Accessed = timestamp()
ON CREATE me SET me.Age = 42

DELETE

Διαγράφει κόμβους, σχέσεις και ιδιότητες που θα επιλέξουμε.

Παράδειγμα: MATCH (me)
OPTIONAL MATCH (me)- [r]-()
DELETE me, r

ORDER BY,LIMIT

Ταξινομεί και σελιδοποιεί τα αποτελέσματα.

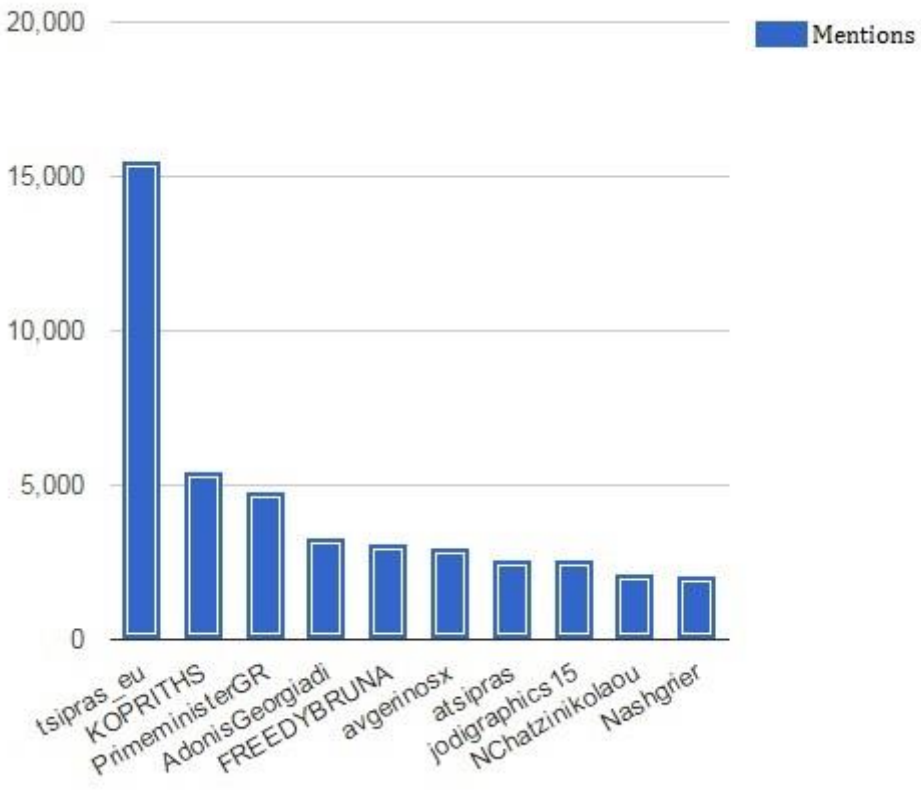
Παράδειγμα: ORDER BY Friends DESC

LIMIT 10

Εφαρμόσαμε ενδεικτικά κάποια ερωτήματα σε Cypher πάνω στα δεδομένα που συλλέξαμε:

Cypher Query για εύρεση των top 10 σε επισημάνσεις χρηστών

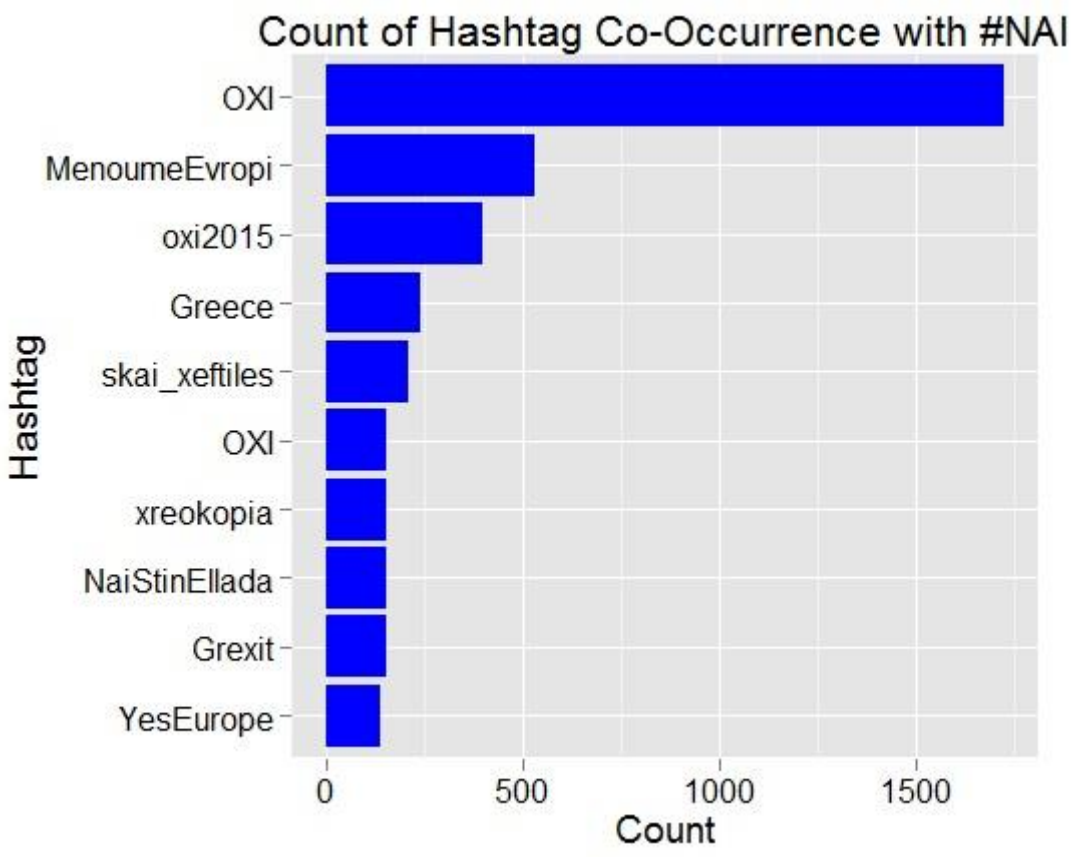
```
MATCH (t:Tweet)-[:MENTIONS]->(u:User)
RETURN u.Screen_Name AS Username, COUNT(t) AS Mentions
ORDER BY Mentions DESC
LIMIT 10
```



Εικόνα 2.6: Top 10 χρήστες σε επισημάνσεις(mentions) στο δίκτυο μας

Cypher Query για συνύπαρξη άλλων hashtag με το Hashtag #NAI

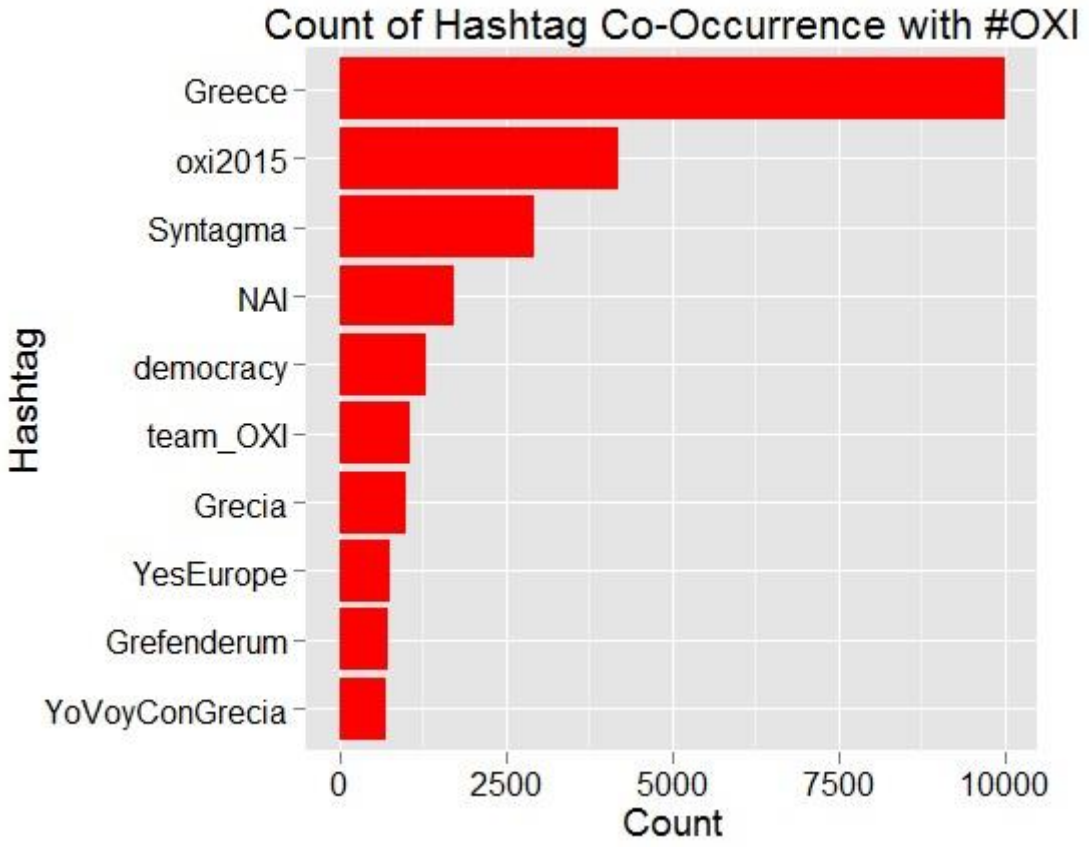
```
MATCH (:Hashtag {Word:'NAI'})<-[:TAGS]-(:Tweet)-[:TAGS]->(h:Hashtag)
WHERE h.Word <>'dimopsifisma' AND h.Word<>'Greferendum' AND h.Word<>'referendum' and h.Word<>'greferendum'
RETURN h.Word AS Hashtag, COUNT(*) AS Count
ORDER BY Count DESC
LIMIT 10
```



Εικόνα 2.7: Συχνότερα εμφανιζόμενα Hashtags σε κοινό tweet με το hashtag #NAI

Cypher Query για συνύπαρξη άλλων hashtag με το Hashtag #OXI

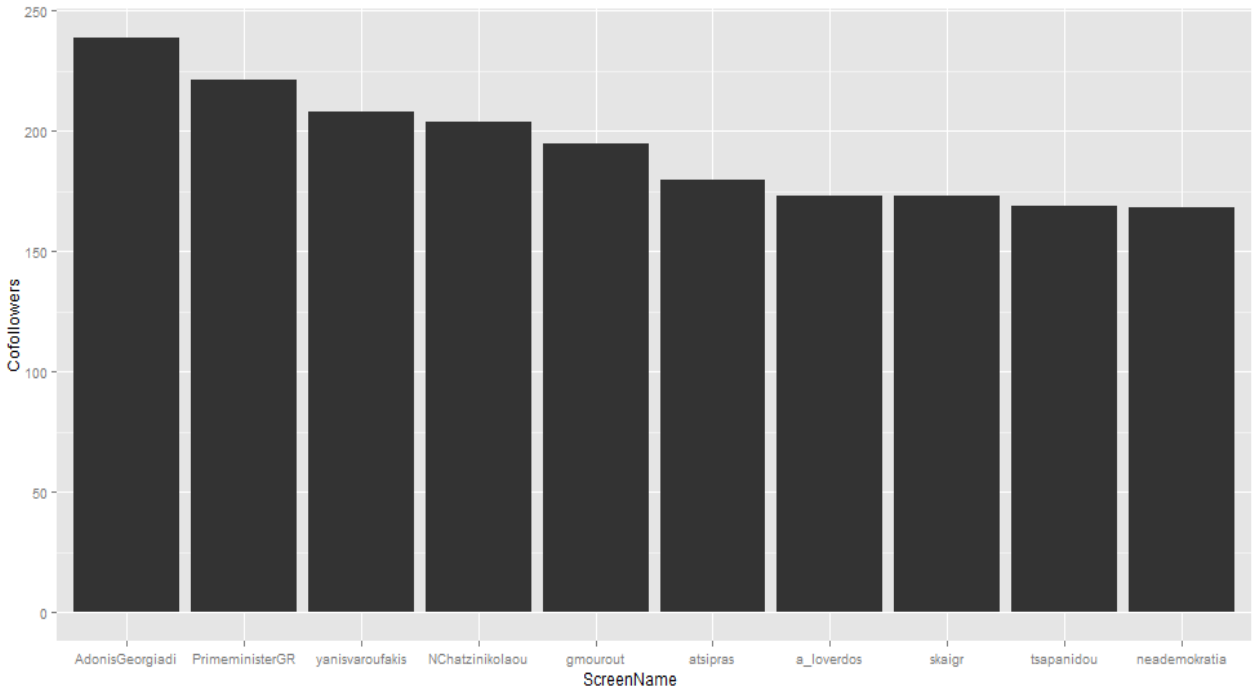
```
MATCH (:Hashtag {Word:'OXI'})<-[:TAGS]-(:Tweet)-[:TAGS]->(h:Hashtag)
WHERE h.Word <>'dimopsifisma' AND h.Word<>'Greferendum' AND h.Word<>'referendum' and h.Word<>'greferendum'
RETURN h.Word AS Hashtag, COUNT(*) AS Count
ORDER BY Count DESC
LIMIT 10
```



Εικόνα 2.8: Συχνότερα εμφανιζόμενα Hashtags σε κοινό tweet με το hashtag #OXI

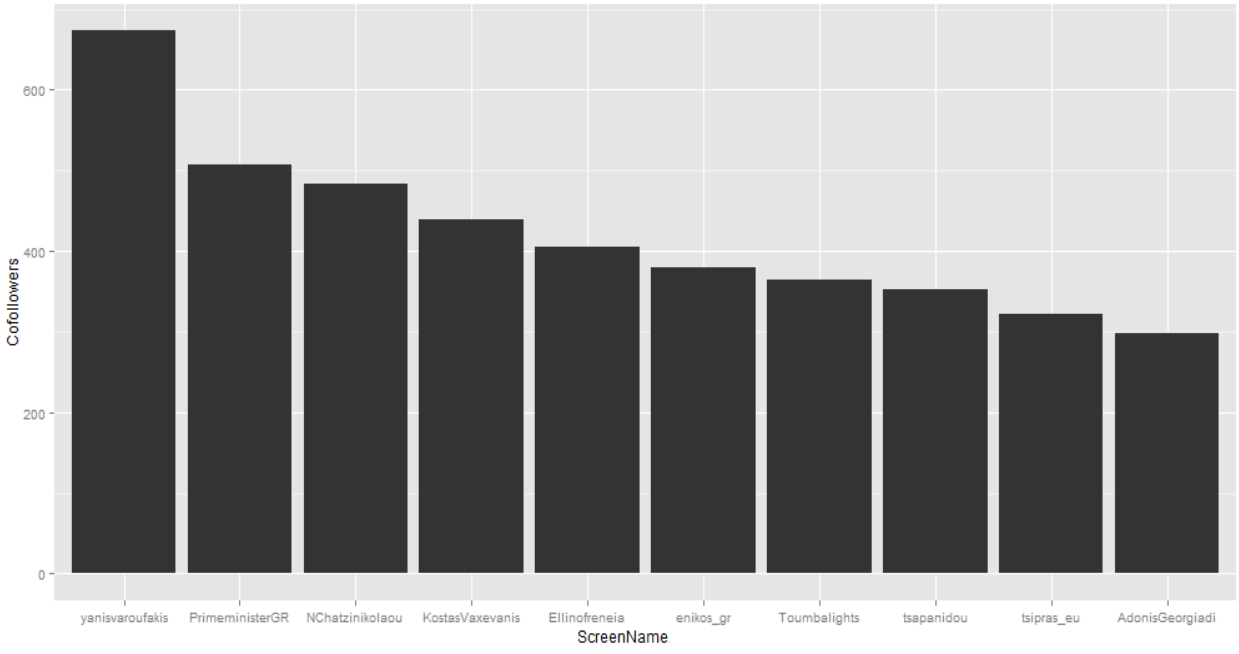
<u>Cypher Query για εύρεση των χρηστών που ακολουθούνται τις περισσότερες φορές από χρήστες που ακολουθούν ένα συγκεκριμένο χρήστη (atsipras, kmitsotakis εδώ)</u>

```
MATCH (user)<-[:FOLLOWS]-(person:’)-[:FOLLOWS]->(user2)
WHERE user.Screen_Name = ‘atsipras’
RETURN user2.Screen_Name as Other, COUNT(*) as Count
ORDER BY Count DESC
LIMIT 10
```



Εικόνα 2.9: Κορυφαίοι χρήστες που ακολουθούν οι χρήστες μαζί με τον χρήστη @kmitsotakis

(Twitter του Κυριάκου Μητσοτάκη)



Εικόνα 2.10: Κορυφαίοι χρήστες που ακολουθούν οι χρήστες μαζί με τον χρήστη @atsipras

(Twitter του Αλέξη Τσίπρα)

Cypher Query για εύρεση των κόμβων με τις περισσότερες εξερχόμενες ακμές

```
START n=node(*)
MATCH n-[r]->m
RETURN n, COUNT(r) as outdegree
ORDER BY outdegree DESC
LIMIT 5
```

Cypher Query για εύρεση των κόμβων με τις περισσότερες εισερχόμενες ακμές

```
START n=node(*)
MATCH n<-[r]-m
RETURN n, COUNT(r) as indegree
ORDER BY indegree DESC
LIMIT 5
```

2.3.4 Λειτουργίες Neo4j

Οι περισσότερες βάσεις δεδομένων σήμερα τρέχουν πάνω σε κάποιο εξυπηρετητή (server) ο οποίος είναι προσβάσιμος μέσω μιας βιβλιοθήκης του εξυπηρετούμενου(client library). Η *Neo4j* μπορεί να τρέξει τόσο σε *ενσωματωμένη λειτουργία (Embedded mode)* όσο και σε *λειτουργία εξυπηρετητή (Server mode)*. [**GDB**]

Embedded Mode: Σε αυτή τη λειτουργία, η Neo4j τρέχει στην ίδια διεργασία με την εφαρμογή μας. Η ενσωματωμένη Neo4j είναι ιδανική για να τρέχει πάνω σε συσκευές hardware, εφαρμογές προσωπικού υπολογιστή αλλά και σε ενσωμάτωση στους εξυπηρετητές των εφαρμογών μας. Το μεγάλο πλεονέκτημα που δημιουργείται λόγω της απευθείας επικοινωνίας εφαρμογής-βάσης είναι η γρήγορη απόκριση (low latency). Άλλα πλεονεκτήματα είναι η επιλογή διαφόρων APIs για δημιουργία και διάσχιση των δεδομένων (Core API, Traversal Framework, Cypher) και οι ρητές συναλλαγές (explicit transactions)

Η *Embedded* έκδοση της Neo4j μπορεί να συσταδοποιηθεί για μεγαλύτερη διαθεσιμότητα και οριζόντια κλιμάκωση ανάγνωσης, όπως ακριβώς και η Server έκδοση.

Server Mode: Το τρέξιμο της Neo4j σε λειτουργία εξυπηρετητή είναι το πιο σύνηθες σήμερα. Στην καρδιά κάθε εξυπηρετητή βρίσκεται ένα ενσωματωμένο στιγμιότυπο της Neo4j.

Κάποια από τα ***οφέλη*** της λειτουργίας *εξυπηρετητή* είναι:

REST API: Ο εξυπηρετητής εκθέτει ένα πλούσιο REST API που επιτρέπει στους εξυπηρετούμενους να στέλνουν JSON-διαμορφωμένα αιτήματα μέσω HTTP.

Οι αποκρίσεις περιέχουν JSON-διαμορφωμένα έγγραφα εμπλουτισμένα με συνδέσμους υπερμέσων που επιδεικνύουν επιπρόσθετα χαρακτηριστικά του συνόλου δεδομένων. Το REST API είναι επεκτάσιμο από τους τελικούς χρήστες και υποστηρίζει την εκτέλεση ερωτημάτων Cypher.

Ανεξαρτησία Πλατφόρμας: Επειδή η πρόσβαση γίνεται μέσω JSON-διαμορφωμένων εγγράφων που αποστέλλονται μέσω HTTP, ένας εξυπηρετητής του Neo4j μπορεί να προσεγγιστεί από εξυπηρετούμενου που τρέχει σχεδόν σε οποιαδήποτε πλατφόρμα. Το μόνο που χρειάζεται είναι μια HTTP βιβλιοθήκη εξυπηρετούμενου (client library).

Ανεξαρτησία Κλιμάκωσης(Scaling): Όταν η Neo4j τρέχει σε λειτουργία εξυπηρετητή, μπορούμε να κλιμακώσουμε τη συστάδα της βάσης δεδομένων μας ανεξάρτητα από τη συστάδα του εξυπηρετητή.

Απομόνωση από συμπεριφορές *Συλλογής Απορριμάτων (Garbage Collection-GC)*: Στην λειτουργία εξυπηρετητή, η Neo4j προστατεύεται από τυχόν GC συμπεριφορές που προκαλούνται από κομμάτι της εφαρμογής. Η Neo4j και πάλι παράγει κάποια "σκουπίδια" αλλά η επιρροή στον Garbage Collector παρακολουθείται προσεκτικά και ρυθμίζεται κατά τη διάρκεια της ανάπτυξης για να μετριαστούν τυχόν παρενέργειες.

Ένα ***μειονέκτημα*** της λειτουργίας εξυπηρετητή σε σχέση με την ενσωματωμένη λειτουργία είναι ότι ίσως χάσει σε απόδοση, λόγω τού ότι οι υπηρεσίες και ο εξυπηρετητής δεν βρίσκονται στο ίδιο μηχάνημα και έτσι απαιτούνται περισσότερες συναλλαγές για την επικοινωνία τους.

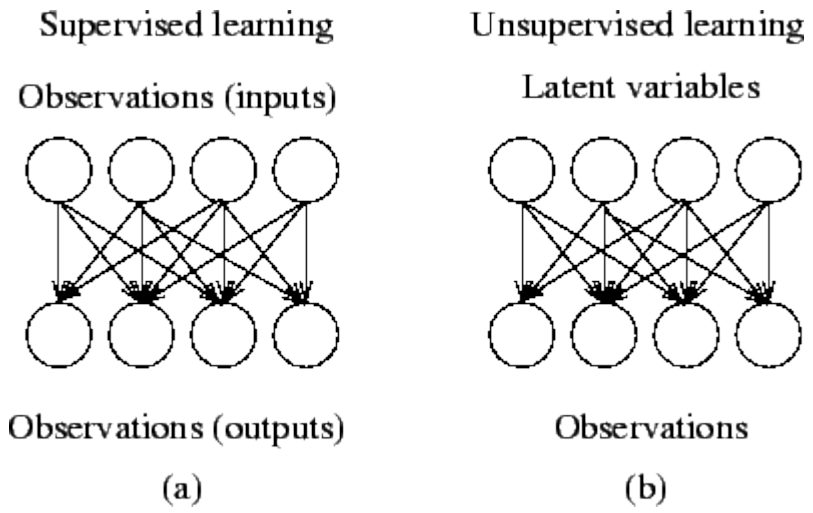
Στην εργασία μας διαλέξαμε να χρησιμοποιήσουμε τη *λειτουργία εξυπηρετητή* για να έχουμε μεγαλύτερη ευελιξία σε περίπτωση που θελήσουμε να έχουμε πρόσβαση στη βάση μας από πολλαπλά απομακρυσμένα μηχανήματα.

2.4 Μηχανική Μάθηση

Η μηχανική μάθηση, όπως περιγράφει και το όνομα της, είναι η διαδικασία που στοχεύει στή δημιουργία μηχανών ικανών να μαθαίνουν, δηλαδή ικανών να βελτιώνουν την απόδοση τους σε συγκεκριμένες διεργασίες μέσω της αξιοποίησης προγενέστερης γνώσης και εμπειρίας αλλά και με την εκμετάλλευση αυτών έτσι ώστε μπορούν να παίρνουν όσο το δυνατό σωστότερες αποφάσεις.

Η μηχανική μάθηση μπορεί να είναι είτε *Επιβλεπόμενη (Supervised Learning)* είτε *Μη-Επιβλεπόμενη (Unsupervised Learning)*. Στην *Επιβλεπόμενη Μάθηση* παρέχονται σύνολα δεδομένων που αναπαριστούν την επιθυμητή έξοδο και χρησιμοποιούνται για την εκπαίδευση της μηχανής ώστε να πάρουμε την επιθυμητή έξοδο ενώ στην *Μη-Επιβλεπόμενη Μάθηση* δεν παρέχεται κάποιο σύνολο επιθυμητής εξόδου και αντί αυτού τα δεδομένα συσταδοποιούνται σε διαφορετικές κλάσεις.

Εμείς στην παρούσα εργασία χρησιμοποιούμε τεχνικές Μη-Επιβλεπόμενης μηχανικής μάθησης.



Εικόνα 2.11: Σύγκριση Επιβλεπόμενης-Μη-Επιβλεπόμενης μάθησης

[UVSS]

2.4.1 Μη-Επιβλεπόμενη Μάθηση

Στην Μη-Επιβλεπόμενη Μάθηση η μηχανή λαμβάνει απλά ως είσοδο δεδομένα $x_1,x_2,x_3,\ldots\ldots,x_n$ χωρίς να δέχεται αναμενόμενα αποτελέσματα και ερεθίσματα από το περιβάλλον. Μπορεί να φαίνεται περίεργο για κάποιον να φανταστεί οτι μιά μηχανή μπορεί να μάθει χωρίς να δέχεται κάποια αναπληροφόρηση από το περιβάλλον της. Ωστόσο, είναι εφικτό να αναπτυχθεί κάποιος σκελετός Μη-Επιβλεπόμενης μάθησης βασιζόμενος στην αντίληψη ότι ο στόχος της μηχανής είναι η οικοδόμηση αναπαραστάσεων της εισόδου που θα μπορούν να χρησιμοποιηθούν στη λήψη αποφάσεων, στην πρόβλεψη των μελλοντικών εισόδων, στην αποτελεσματική μεταβίβαση των εισόδων σε ένα άλλο μηχάνημα, κ.λ.π.

Κατά μία έννοια, η Μη-Επιβλεπόμενη Μάθηση μπορεί να θεωρηθεί ως η *αναγνώριση προτύπων (pattern recognition*), πέρα του καθαρού αδόμητου θορύβου, μέσα στα δεδομένα **[UL]**.

Απλά παραδείγματα Μη-Επιβλεπόμενης Μάθησης είναι η *Συσταδοποίηση (Clustering)*, η *Μοντελοποίηση Θεμάτων (Topic Modeling)* και η *Μείωση Διαστάσεων (Dimensionality Reduction)* τα οποία θα δούμε παρακάτω.

2.4.2 Συσταδοποίηση Κειμένου

Η ***συσταδοποίηση (Clustering)*** γενικότερα είναι μια τεχνική μη επιβλεπόμενης μάθησης η οποία αποσκοπεί στην ομαδοποίηση αντικειμένων μεταξύ τους με βάση κάποιο δείκτη ομοιότητας έτσι ώστε τα αντικείμενα που παρουσιάζουν τη μεγαλύτερη ομοιότητα να βρίσκονται στην ίδια ομάδα (συστάδα).

Πιο συγκεκριμένα, η *συσταδοποίηση κειμένου* είναι μια από τις θεμελιώδεις λειτουργίες της εξόρυξης κειμένου. Η συσταδοποίηση κειμένου σημαίνει τη διαίρεση μιας συλλογής εγγράφων κειμένου σε ομάδες διάφορων κατηγοριών, έτσι ώστε τα έγγραφα που ανήκουν στην ίδια ομάδα να περιγράφουν το ίδιο θέμα όπως για παράδειγμα “Οικονομική Κρίση” ή “Μοντέρνα Τέχνη”. Σε αντίθεση με τη συσταδοποίηση δομημένων δεδομένων, η συσταδοποίηση κειμένου αντιμετωπίζει μια σειρά από νέες προκλήσεις με τις σημαντικότερες από αυτές να είναι: ο όγκος των δεδομένων, η διάσταση, η αραιότητα και η σημασιολογική πολυπλοκότητα.

Πρίν τη μοντελοποίηση του αρχικού κειμένου στο διανυσματικό χώρο απαιτείται πληθώρα προεπεξεργαστικών βημάτων. Τα βήματα που χρησιμοποιήθηκαν για το φιλτράρισμα των δεδομένων είναι : η αφαίρεση κάποιων εξαιρούμενων λέξεων(stopwords) που η ύπαρξη τους στα δεδομένα δεν θα απέδιδε κάποιο νόημα με βάση μία λίστα αποκλεισμού , η αφαίρεση των διπλών λέξεων από το κάθε κείμενο , η μετατροπή των επιμέρους λέξεων σε λέξεις κοινής μορφολογίας (μικρά γράμματα,όχι τονισμός) για να εξισωθούν λέξεις με κοινό νόημα σε κοινή μορφή, καθώς και η αφαίρεση λέξεων που παρουσιάζουν χαμηλή συχνότητα εμφάνισης στο λεξιλόγιο διότι λόγω της σπανιότητας τους δεν είναι ικάνές να εκφράσουν άποψη μιας ομάδας. Η σημαντικότητα ύπαρξης αυτών των προεπεξεργαστικών βημάτων είναι υψηλή καθώς μπορούν να επηρεάσουν θετικά τα αποτελέσματα και να απομακρύνουν τον λεγόμενο "θόρυβο" στο βαθμό που αυτό είναι εφικτό.

Οι αλγόριθμοι συσταδοποίησης χωρίζονται σε 2 χαρακτηριστικές κατηγορίες: αλγόριθμοι *διαχωρισμού (partitioning)* και *συσσωρευτικοί(agglomerative)* αλγόριθμοι, οι οποίοι εκπροσωπούνται από το μοντέλο *K-Means* και την *Ιεραρχική Συσταδοποίηση* αντίστοιχα. Στην περίπτωση μας εφαρμόζουμε τον αλγόριθμο σε δεδομένα κοινωνικών δικτύων μεγάλου όγκου. Η *Ιεραρχική συσταδοποίηση* είναι μια μέθοδος πού επιδιώκει την κατασκευή ιεραρχίας συστάδων. Αρχικά κάθε παρατηρηση (κείμενο) ανήκει στη δικιά της συστάδα και συγχωνεύεται με άλλες συστάδες όσο ανεβαίνει στην ιεραρχία. Η κάθε συστάδα παρουσιάζει υψηλό υπολογιστικό κόστος κατά την αύξηση των δεδομένων, αφού χρειάζεται να υπολογίζει εναν τετραγωνικό πίνακα ομοιότητας και να συγχωνεύει μικρές συστάδες κάθε φορά με χρήση συναρτήσεων σύνδεσης. Αντίθετα ο *K-Means*, είναι ένας επαναληπτικός αλγόριθμος, ο οποίος ανανεώνει τα κέντρα τον συστάδων μέσω κανονικοποίσης σε κάθε επανάληψη και επανατοποθετεί κάθε *κείμενο* στο νέο κοντινότερο κέντρο. Λόγω ταχύτητας, στην παρούσα έρευνα επιλέξαμε τον *K-Means* για τη Συσταδοποίηση Κειμένου και θα δούμε περισσότερα σχετικά με αυτόν παρακάτω. Μια σύγκριση των 2 αλγορίθμων βρίσκεται σε μελέτη για τεχνικές συσταδοποίησης εγγράφων **[Steinbach, Karypis & Kumar 2000]**.

2.4.2.1 *K-Means*

Η συσταδοποίηση K-Means είναι μια μέθοδος κβάντωσης διανυσμάτων, προερχόμενη από την επεξεργασία σημάτων, και αρκετά διαδεδομένη για ανάλυση συστάδων στην εξόρυξη δεδομένων. Η συσταδοποίηση *K-Means* στοχεύει στο διαχωρισμό *n* παρατηρήσεων σε *k*

συστάδες. Κάθε παρατήρηση λέμε ότι ανήκει σε κείνη τη συστάδα με τον πλησιέστερο μέσο. Γίνεται χρήση της ευκλείδειας απόστασης ως μετρικής ομαδοποίησης.

Η συσταδοποίηση *k-Means* αποτελεί ένα εκ των απλούστερων αλγόριθμων μη-επιβλεπόμενης μάθησης ο οποίος επιλύει το γνωστό πρόβλημα συσταδοποίησης. Η διαδικασία αποτελεί έναν απλό τρόπο ταξινόμησης ενός δοσμένου συνόλου δεδομένων εντός ορισμένου αριθμού συστάδων (έστω k) που έχει προκαθοριστεί. Η κύρια ιδέα είναι ο καθορισμός k κέντρων, ενός για κάθε μία εκ των συστάδων**. [KM]**

Αυτά τα κέντρα πρέπει να τοποθετηθούν με όσο το δυνατόν πιο σωστό τρόπο διότι διαφορετικές τοποθεσίες επιφέρουν διαφορετικά αποτελέσματα. Για αυτό το λόγο η καλύτερη επιλογή είναι να τα τοποθετήσουμε όσο μακριά γίνεται μεταξύ τους. Το επόμενο βήμα είναι να πάρουμε κάθε σημείο που ανήκει στο σύνολο δεδομένων και να το "ταιριάξουμε" με το κοντινότερο κέντρο. Όταν δεν εκκρεμεί κανένα σημείο, το πρώτο βήμα που αποτελεί την πρωταρχική ομαδοποίηση ολοκληρώνονται.

Έπειτα πρέπει να ξαναϋπολογιστούν τα *k* νέα κέντρα ως βαρύκεντρα των συστάδων που προέκυψαν από το προηγούμενο βήμα. Εφόσον πάρουμε αυτά τα k νέα κέντρα των συστάδων λοιπόν, πρέπει να γίνει ένα νέο ταίριασμα μεταξύ των σημείων δεδομένων και των πλησιέστερων νέων κέντρων των συστάδων.

Πλέον όπως γίνεται ξεκάθαρο προκύπτει ένας βρόχος. Το αποτέλεσμα που παίρνουμε από αυτόν τον βρόχο είναι η μεταβολή της τοποθεσίας των κέντρων των *k* συστάδων. Μιά αλλαγή που θα γίνεται μέχρι αυτά να σταματήσουν να βελτιώνονται οπότε και να μετακινούνται.

Τέλος ο αλγόριθμος στοχεύει στην ελαχιστοποίηση μια αντικειμενικής συνάρτησης, γνωστής και ως συνάρτησης τετραγωνικού σφάλματος η οποία δίνεται από τη σχέση:

$$J(V) = \sum_{i=1}^{c} \sum_{j=1}^{c_i} \left(\left\| x_i - v_j \right\| \right)^2$$

όπου:

$\|x_i - v_j\|$**:** η Ευκλείδεια απόσταση μεταξύ των: x_i που αντιπροσωπεύει κάθε σημείο δεδομένων και v_j όπου παριστάνει κάποιο κέντρο συστάδας.

c_i**:** ο αριθμός των σημείων δεδομένων στην *i-οστή* συστάδα.

c: ο αριθμός των κέντρων συστάδων.

[KM]

Αλγοριθμικά Βήματα Αλγόριθμου K-Means

Αν θεωρήσουμε X = $\{x_1,x_2,x_3,\ldots\ldots\ldots,x_n\}$ το σύνολο των σημείων (data points) και V=$\{v_1,v_2,\ldots\ldots\ldots,v_c\}$ το σύνολο των κέντρων συστάδων.

1. Τυχαία επιλογή των *'c'* κέντρων συστάδων.

2. Υπολογισμός της απόστασης μεταξύ κάθε σημείου και των κέντρων συστάδων.

3. Ανάθεση του σημείου (data point) στο κέντρο της συστάδας, προς το οποίο η απόσταση είναι η μικρότερη σε σχέση με κάθε άλλο κέντρο συστάδας.

4. Επανυπολογισμός του νέου κέντρου συστάδας με χρήση της σχέσης:

$$v_i = \left(\frac{1}{c_i}\right)\sum_{j=1}^{c_i}(x_i)$$

όπου το $\mathbf{c_i}$ παριστάνει τον αριθμό των σημείων (data points) της *i-στής* συστάδας.

5. Επανυπολογισμός της απόστασης μεταξύ του κάθε σημείου (data point) και των νέων κέντρων συστάδων.

6. Αν δεν έχουμε επαναπροσδιορισμό κάποιου σημείου (data point) τότε σταματάει η διαδικασία. Σε αντίθετη περίπτωση συνέχιση της διαδικασίας από το βήμα 3. **[KM]**

2.4.2.2 *Επιλογή Αριθμού Συστάδων με συνιστώσα Silhouette*

Η συνιστώσα Silhouette είναι μια μέθοδος μετάφρασης και επιβεβαίωσης της συνέπειας εντός των συστάδων. Η τεχνική αυτή μας παρέχει μια σύντομη αναπαράσταση για το πόσο καλά ανήκει το κάθε αντικείμενο στη συστάδα του.

Ας υποθέσουμε ότι τα αντικείμενα συσταδοποιούνται μέσω οποιασδήποτε τεχνικής, όπως για παράδειγμα του αλγόριθμου συσταδοποίησης K-means, σε *k* ομάδες.

Για κάθε αντικείμενο *i*, θεωρούμε ως *α(i)* τον μέσο όρο της ανομοιότητας του i με όλα τα άλλα στοιχεία που βρίσκονται εντός της ίδιας συστάδας. Μπορούμε να εμηνεύσουμε την τιμή του *α(i)* ως μια μετρική για το πόσο καλά έχει ομαδοποιηθεί το αντικείμενο *i* στη συστάδα του (όσο μικρότερη είναι η τιμή του, τόσο καλύτερη είναι η εκχώρηση που έχουμε κάνει).

Στην συνέχεια ορίζουμε το *b(i)* ως τη χαμηλότερη μέση ανομοιότητα του *i* ως προς οποιαδήποτε άλλη συστάδα, της οποίας το *i* δεν είναι μέλος. Η συστάδα που έχει την επόμενη χαμηλότερη μέση ανομοιότητα λέμε ότι είναι η "γειτονική συστάδα" του *i* διότι θα αποτελούσε την αμέσως επόμενη καλύτερη συστάδα για να ομαδοποιηθεί το αντικείμενο *i*.

Ορίζουμε τώρα το *Silhouette Score* ως:

$$s(i) = \frac{b(i)-a(i)}{max\{a(i),b(i)\}}$$

Το παραπάνω score μπορεί να πάρει τιμές από -1 εώς και 1. Το 1 δείχνει οτί σωστά το αντικείμενο ανήκει στη συστάδα που βρίσκεται. Αντίθετα το -1 δείχνει ότι θα έπρεπε να είχε τοποθετηθεί κανονικά στη γειτονική συστάδα. Τέλος μια τιμή κοντά στο ουδέτερο 0 δείχνει οτι το αντικείμενο βρίσκεται στα σύνορα των συστάδων. **[WSIL]**

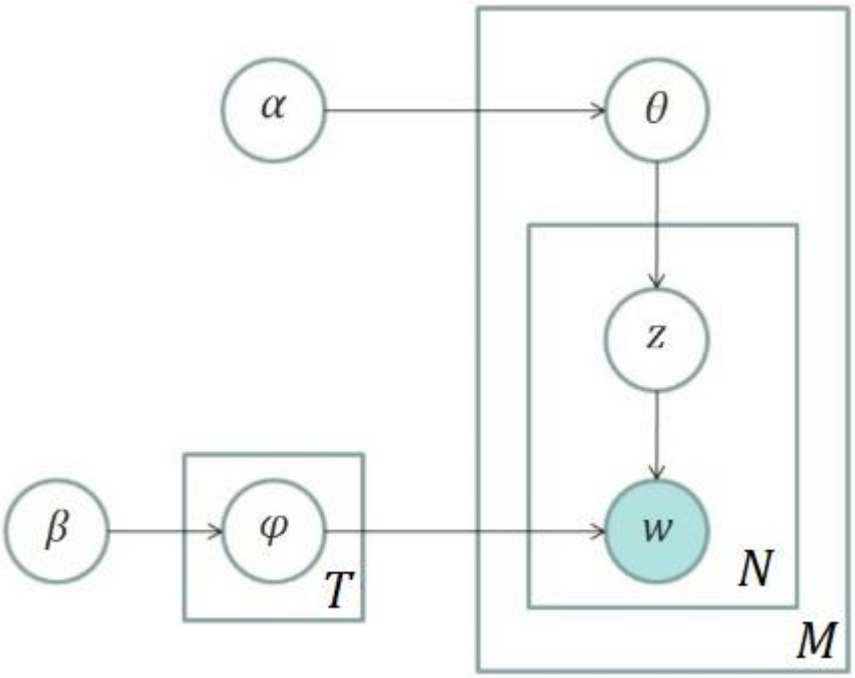
2.4.3 Μοντελοποίηση Θεμάτων Συζήτησης με LDA

Το *Latent Dirichlet Allocation (LDA)* **[Blei, et al.,2003]** είναι ενα παραγωγικό πιθανοτικό μοντέλο για συλλογές διακριτών δεδομένων όπως κείμενα. Τα δεδομένα μοντελοποιούνται ως μια κατανομή *Θεμάτων Συζήτησης (Topics)*, και κάθε *Θέμα* με τη σειρά του ως κατανομή *Λέξεων(Words)*.

Στο παρακάτω σχήμα διαφαίνεται η γραφική αναπαράσταση του μοντέλου LDA. Οι σκιασμένοι κόμβοι είναι οι παρατηρούμενες μεταβλητές και οι μη-σκιασμένοι είναι οι *λανθάνουσες (latent)* μεταβλητές. Τα βέλη αναπαριστούν τις εξαρτήσεις και τα ορθογώνια αναπαριστούν τις διαδικασίες επαναλαμβανόμενης δειγματοληψίας. Οι τιμές *M, N,* και *T* είναι αντίστοιχα: το πλήθος των *Κειμένων* στη συλλογή, το πλήθος των *Λέξεων* σε κάθε *κείμενο* και το πλήθος των *Θεμάτων Συζήτησης*.

Η τιμή *z* αντιστοιχεί στο Θέμα από το οποίο εξάγεται μια συγκεκριμένη λέξη *w*. Οι ανά-Κείμενο πολυωνυμικές κατανομές Θεμάτων δίνονται από το θ, ενώ το φ δίνει τις ανά-Θέμα πολυωνυμικές κατανομές λέξεων. Προγενέστερες (prior) κατανομές *Dirichlet* τοποθετούνται πάνω από αυτές τις κατανομές. Η Dirichlet είναι μια πολυμεταβλητή κατανομή. Αφού ο *LDA* ακολουθεί ιδέες όπως το ότι κάθε κείμενο μπορεί να αποτελείται από πολλαπλά θέματα και ότι κάθε θέμα μπορεί να αποτελείται από πολλαπλές λέξεις δημιουργούνται ανάγκες μοντελοποίησης αυτών των συσχετίσεων, και αυτές καλύπτονται από τις κατανομές Dirichlet που είναι στη φύση τους να αντιμετωπίζουν τέτοιες πολλαπλότητες. Αυτές οι Dirichlet κατανομές παραμετροποιούνται από τα α *(Alpha)* και β *(Beta)* αντίστοιχα και δεν φαίνονται από τα δεδομένα, για αυτό και τις αποκαλούμε *λανθάνουσες (latent)* ή *κρυφές (hidden)*. Όσο

πιο χαμηλές οι τιμές των Alpha και Beta τόσο πιο λίγα τα θέματα ανά κείμενο και οι λέξεις ανά θέμα αντίστοιχα.



Εικόνα 2.12: Διαδικασία LDA

[Blei, et al.,2003]

Συγκεκριμένα για N Θέματα έχουμε:

$$p(w) = \int_{\theta} \left(\prod_{n=1}^{N} \sum_{z_n=1}^{k} p(w_n|z_n;\beta)p(z_n|\theta) \right) p(\theta;a)d\theta$$

Όπου α είναι η προγενέστερη παράμετρος κατανομής ανά αντικείμενο και η β είναι η προγενέστερη Dirichlet παράμετρος κατανομής ανά θέμα. **[Blei, et al.,2003]**

Ο αλγόριθμος Latent Dirichlet Allocation είναι μια ευθεία διαδικασία και τα βήματα του περιγράφονται παρακάτω:

Αλγοριθμικά Βήματα Αλγόριθμου LDA

1. Επιλογή τιμών για τις υπερπαραμέτρους α και β και για το πλήθος των Θεμάτων Συζήτησης (Topics) Τ. Οι τιμές των α και β βασίζονται στο T και το μέγεθος του λεξιλογίου. Καλές γενικά επιλογές για τις παραμέτρους αυτές είναι $\alpha = 50/T$ και $\beta = 0.01$. **[Steyvers & Griffiths, 2007]**
2. Για κάθε Κείμενο:
   (a) Επιλογή του αριθμού των Λέξεων N.
   (b) Για κάθε Λέξη:
      i. Δειγματοληψία z από $\theta(j)$, όπου j είναι ο δείκτης του τρέχοντος κειμένου.
      ii. Δειγματοληψία w από $\varphi(z)$.

Για να γίνει η συσταδοποίηση κειμένων με τη χρήση του LDA πρέπει να βρούμε την πιθανότητα $P(z|w)$ για δεδομένα α, β και T . Γενικά, αυτό το πρόβλημα είναι δυσεπίλυτο **[Blei, et al.,2003]**; γνωστές τεχνικές προσέγγισης είναι η μεταβολλική EM (Expectation-Maximization) **[Blei, et al.,2003]** και η δειγματοληψία Gibbs **[Griffiths & Steyvers, 2004]** η οποία είναι η πιό συνηθισμένη και χρησιμοποιείται στην εργασία μας.

Εφόσον υπολογιστεί η $P(z|w)$, οι κατανομές φ και θ μπορούν να εκτιμηθούν για κάθε Θέμα και κάθε κείμενο. Οι κατανομές θ των Θεμάτων διαμορφώνουν τη βάση της μεθόδου συσταδοποίησης μας.

2.4.4 Μείωση Διαστάσεων με t-SNE

Ο κύριος σκοπός των τεχνικών *Μείωσης Διάστασης* (Dimensionality Reduction) είναι η συμπυκνωμένη αναπαράσταση της πληροφορίας που παρέχεται για περαιτέρω ανάλυση. Έτσι ιδανικά δημιουργείται ένα νέο σύνολο δεδομένων το οποίο αποτελείται από λιγότερες παρατηρήσεις αλλά περιέχει τα σημαντικότερα χαρακτηριστικά των αρχικών δεδομένων. Το νόημα χρήσης τέτοιων τεχνικών είναι η αποτελεσματική και έγκαιρη αναπαράσταση τεράστιων συνόλων δεδομένων.**[DMML]**

Ο ***t-SNE*** *(t-Distributed Stochastic Neighbor Embedding)* είναι ένας αλγόριθμος που χρησιμοποιείται για την οπτικοποίηση δεδομένων υψηλών διαστάσεων σε χώρο χαμηλών

διαστάσεων διατηρώντας τις υψηλών διαστάσεων ανά ζευγάρι ομοιότητες σε χαμηλών διαστάσεων ενσωμάτωση. **[WTSN]**

Πρακτικά ο *t-SNE* μετατρέπει τις σχέσεις των σημείων δεδομένων σε πιθανότητες. Οι σχέσεις στον πραγματικό χώρο αναπαριστώνται από *Γκαουσσιανές(Gaussian) συνδεδεμένες πιθανότητες*, ενώ οι σχέσεις στο χώρο ενσωμάτωσης αναπαριστώνται από *t-κατανομές Student*. Αυτό επιτρέπει στον *t-SNE* να είναι σχετικά ευαίσθητος στην τοπική δομή και του δίνει κάποια *πλεονεκτήματα* σε σχέση με τις άλλες τεχνικές μείωσης διαστάσεων.

Πλεονεκτήματα:

- Αποκαλύπτει τη δομή πολλών κλιμάκων σε ένα εννιαίο χάρτη.
- Αποκαλύπτει δεδομένα που κρύβονται σε πολλαπλές, διαφορετικές συλλογές ή συστάδες.
- Μειώνει την τάση συγκέντρωσης των σημείων στο κέντρο.

Ένα ***μειονέκτημα*** του t-SNE είναι ότι η συνολική δομή δεν διατηρείται ρητά.Ωστόσο αυτό το πρόβλημα μετριάζεται με την αρχικοποίηση των σημείων μέσω της στατιστικής διαδικασίας *PCA* (Principal Component Analysis).

Ο αλγόριθμος ***Barnes-Hut*** t-SNE που χρησιμοποιήσαμε εμείς, αποτελεί παραλλαγή του t-SNE και καταφέρνει την αντίστοιχη ενσωμάτωση με τον *t-SNE* σε *O(NlogN)* αντί για $O(n^2)$. Αυτό ουσιαστικά μας επιτρέπει να επιτύχουμε ενσωματώσεις συνόλων δεδομένων με εκατομμύρια, αντί για δεκάδες χιλιάδες στοιχεία. **[van der Maaten, 2013]**

3

Σχετικές Εργασίες

Μεγάλο μέρος της δημοσιευμένης έρευνας σχετικά με το Twitter επικεντρώνεται σε ερωτήσεις σχετικά με το δίκτυο που το εκφράζει και την δομή της κοινότητας που αυτό δημιουργεί. Για παράδειγμα μια έρευνα **[Krishnamurthy, Gill & Arlitt, 2008]** συγκεντρώνει γενικά χαρακτηριστικά αυτού του κοινωνικού δικτύου όπως τοπολογικές και γεωγραφικές ιδιότητες, μοτίβα ανάπτυξης και συμπεριφορές των Χρηστών του.

Κάποια άλλη έρευνα **[Java,et al., 2007]** υποστηρίζει από τη σκοπιά του δικτύου, ότι οι δραστηριότητες των χρηστών του Twitter μπορούν να θεωρηθούν ως δραστηριότητες αναζήτησης πληροφορίας, διαμοιρασμού πληροφορίας ή κοινωνικοποίησης. Για την ταυτοποίηση των διαφόρων τύπων προθέσεων των χρηστών εντός του Twitter προτάθηκε ένα πλαίσιο 2 επιπέδων για ανίχνευση της πρόθεσης των χρηστών.

Ένα άλλο, μικρότερο κομμάτι μελετών παρουσιάζει συστηματική ανάλυση του περιεχομένου των κειμένων του Twitter. Πιο πρόσφατες έρευνες εξετάζουν το περιεχόμενο των Tweets με ιδιαίτερη έμφαση σε συγκεκριμένες τεχνοτροπίες του Twitter. Για παράδειγμα μια έρευνα **[Honeycutt & Herring, 2009]** δίνει βάση στις *Επισημάνσεις Χρηστών*(@Mention), με τα σημαντικότερα ευρήματα που προέκυψαν από την ανάλυση δεδομένων/Tweets να είναι: η συγκέντρωση διαδραστικού περιεχόμενου, όπως προτροπές προς άλλους χρήστες και διαμοιρασμός περιεχομένου προς τρίτους σε Tweets με το σύμβολο @ και το εύρος φάσματος που παρουσιάζουν τα tweets που το περιέχουν ως προς το περιεχόμενο σε σύγκριση με εκείνα που δεν το περιέχουν και συνήθως απαντάνε στην απλή ερώτηση: *"Τι κάνεις;"*

Άλλη έρευνα πάλι **[Boyd, Golder, & Lotan, 2010]** που βασίζεται στις *Αναδημοσιεύσεις (Retweets)* και μετά από ανάλυση δεδομένων που προέρχονται από το API του Twitter περιγράφει τις διάφορες παραλλαγές στην αναδημοσίευση μηνυμάτων στο Twitter και τους τρόπους όπου οι ποικίλες μορφές οδηγούν σε ασάφεια σχετικά με την αρχική προέλευση, την απόδοση και την πιστότητα της ομιλίας, ειδικά όσο το περιεχόμενο μορφοποιείται κατά την διάδοση.

Μια άλλη έρευνα που είναι άξια αναφοράς **[Naaman, Boase, & Lai, 2010]**, χαρακτηρίζει το περιεχόμενο του Twitter μέσω χειρωνακτικής κατηγοριοποίησης των Tweets σε κατηγορίες ποικίλης ειδικότητας από την κατηγορία *"Διαμοιρασμός Πληροφορίας"* μέχρι την κατηγορία *"Αυτό-προώθηση"*. Σε αυτή την εργασία αναλύονται αυτές οι κατηγορίες με βάση το σύνολο δεδομένων που συλλέχθηκε, κατατάσοοντας τους χρήστες σε 2 κύριες συστάδες: αυτούς που μεταδίδουν μη-προσωπικές πληροφορίες στον κόσμο για ενημέρωση (*Informers*) και αποτελούν το 20% των χρηστών και αυτούς που δημοσιεύουν Tweets που αφορούν τον εαυτό τους (*meformers*) και αποτελούν το 80%. Η κατηγοριοποίηση των χρηστών με βάση τον τύπο των Tweets που συνήθως δημοσιεύουν, έγινε με χρήση της *ανάλυσης σύνδεσης συστάδων του Ward (Ward's Linkage Cluster Analysis)* **[WCA]** και ο εντοπισμός του ιδανικού αριθμού συστάδων που ελαχιστοποιεί τις διαφορές εντός των ομάδων και μεγιστοποιεί τις διαφορές ανάμεσα σε αυτές έγινε με *ανάλυση Kalensky (Kalensky's Analysis)*.

Μια ακόμα ενδιαφέρουσα εργασία **[Cataldi, Di Caro & Schifanella, 2010]** προτείνει μια τεχνική εντοπισμού θεμάτων που επιτρέπει την ανάκτηση των πιο αναδυόμενων θεμάτων που εκφράζονται από την κοινότητα του Twitter σε ζωντανό χρόνο. Σε αυτήν την έρευνα εξάγονται οι όροι των tweets και γίνεται μοντελοποίηση του κύκλου ζωής τους σύμφωνα με μια καινοφανή θεωρία γήρανσης που εξάγει τους αναδυόμενους, δηλαδή αυτούς που ενώ ήταν σχετικά σπάνια η εμφάνιση τους στο παρελθόν, εμφανίζονται συχνά τώρα. Έπειτα γίνεται ανάλυση των κοινωνικών σχέσεων του δικτύου με καθορισμό του βαθμού αξιοπιστίας των χρηστών, με βάση τον αλγόριθμο *Page-Rank* έτσι ώστε να επιτευχθεί ποσοτικοποίηση της σημαντικότητας κάθε αναλυόμενου όρου. Τέλος δημιουργείται ένας γράφος θεμάτων ο οποίος συνδέει τους αναδυόμενους όρους με σημασιολογικά συσχετισμένες λέξεις κλειδιά, επιτρέποντας τον εντοπισμό των αναδυόμενων θεμάτων σε καθορισμένα χρονικά παράθυρα.

Στο πλαίσιο μια άλλης έρευνας **[Weng,Lim,Jiang & He, 2010]** υλοποιείται η ταυτοποίηση των Χρηστών του Twitter που έχουν επιρροή, με βάση τον αλγόριθμο *TwitterRank(TR)* και επισημαίνεται το φαινόμενο της ομοφιλίας (homophily) στο Twitter. Ο αλγόριθμος *Latent Dirichlet Allocation* (*LDA)* εφαρμόζεται για αυτόματη αναγνώριση των Θεμάτων στα οποία ενδιαφέρεται οι χρήστες με βάση τα tweets που αυτοί δημοσιεύουν. Πιο συγκεκριμένα υπολογίζεται τη θεματική διαφορά (topical difference) μεταξύ 2 χρηστών (twitterer) του Twitter μετά τον υπολογισμό της απόκλισης Jensen-Shannon μεταξύ των 2 κατανομών πιθανοτήτων για τον κάθε χρήστη. Αποδεικνύεται ότι η μέση θεματική διαφορά των ζευγαριών χρηστών με σχέση "ακολουθίας" είναι μικρότερη από αυτών χωρίς και ότι η μέση θεματική διαφορά των ζευγαριών χρηστών με αμοιβαία σχέση "ακολουθίας" είναι μικρότερη από αυτών χωρίς. Γίνεται σύγκριση με άλλους αλγορίθμους όπως πχ. *InDegree* και *PageRank*.

Ακόμα με βάση την υλοποίηση του μοντέλου μερικώς επιβλεπόμενης μάθησης *Labeled LDA* **[Ramage, Hall, Nallapati & Manning, 2009]** έγινε απόπειρα αναπαράστασης του

περιεχόμενου του Twitter σε διαστάσεις που αντιστοιχούν στην ουσία, το στυλ, την κατάσταση και τα κοινωνικά χαρακτηριστικά των Tweets **[Ramage, Dumais, Liebling, 2010]**. Η προσέγγιση αυτή επεκτείνει τον LDA με ενσωμάτωση επίβλεψης όπου γίνεται υποθέτοντας την ύπαρξη ενός συνόλου ετικετών (labels) οι οποίες χαρακτηρίζονται από πολυωνυμική κατανομή πάνω στο σύνολο λέξεων και χρησιμοποιείται για απόδοση κάθε λέξης εντός ενός εγγράφου σε σταθμισμένο μίγμα των labels του, με τις υπόλοιπες λέξεις στο έγγραφο να βοηθάνε στην αποσαφήνιση ανάμεσα στις επιλογές ετικέτας.

Σε άλλη έρευνα **[Yu Xiao,et al., 2010]** γίνεται επισκόπηση διαφόρων τεχνικών συσταδοποίησης κειμένων. Πιο συγκεκριμένα γίνεται σύγκριση μεταξύ 2 μοντέλων που έχουν τραβήξει την προσοχή τα τελευταία χρόνια λόγω της καλής απόδοσης τους: των *Latent Dirichlet Allocation (LDA)* και *mixture of Von Mises-Fisher (moVMF).* Επισημαίνεται ότι κλασικά σύνολα δεδομένων που χρησιμοποιούνται συχνά για ερευνητικούς σκοπούς όπως για παράδειγμα σε παλαιότερες συγκρίσεις μεταξύ των παραπάνω μοντέλων περιέχουν έγγραφα όπου το καθένα αναφέρεται σε 1 θέμα. Αυτό δεν ισχύει για την πληθώρα των δεδομένων σήμερα όπως για παράδειγμα του Twitter, όπου τα έγγραφα (tweets) μπορεί να αποτελούνται από πολλαπλά θέματα. Σε τέτοιου τύπου δεδομένα ο *LDA* υπερτερεί καθώς από τη φύση του έχει φτιαχτεί για να μοντελοποιεί έγγραφα πολλαπλών θεμάτων ενώ αντίθετα ο *moVMF* δεν μπορεί να εντοπίσει παραπάνω θέματα από τον αριθμό των εγγράφων.

Η μοντελοποίηση Θεμάτων κερδίζει την προσοχή μεταξύ των κοινοτήτων εξόρυξης κειμένου. Ο αλγόριθμος *Latent Dirichlet Allocation (LDA)* **[Blei, et al., 2003]** έχει εξελιχθεί ως πρότυπο μοντελοποίησης Θεμάτων. Για αυτούς τους λόγους έχει επεκταθεί ποικιλοτρόπως και συγκεκριμένα όσον αφορά τα κοινωνικά δίκτυα, έχει προταθεί πληθώρα επεκτάσεων του LDA. Για παράδειγμα μια έρευνα **[Chang, Boyd-Graber & Blei, 2009]** αναπτύσσει το *Nubbi (Networks Uncovered By Bayesian Inference),* ένα καινοτόμο πιθανοτικό μοντέλο θεμάτων κειμένου για ανάλυση κειμένου με εξαγωγή περιγραφών των σχέσεων μεταξύ των οντοτήτων.

Μετά από εφαρμογή του σε 3 σύνολα κειμένου (Βίβλος, Wikipedia, επιστημονικά αποσπάσματα) φάνηκε ότι το *Nubbi* αποτελεί ένα μοντέρνο προφητικό (predictive) μοντέλο οντοτήτων και χρήσιμο εργαλείο εξερεύνησης για ανακάλυψη και κατανόηση δεδομένων κρυμμένων σε απλό κείμενο

Σε σχετική εργασία που στοχεύει στην ανάκτηση μίγματος Θεμάτων από μηνύματα και χρήστες που δημοσιεύουν στο Twitter **[Rosen-Zvi, Griffiths, Steyvers & Smyth, 2004],** εισήχθηκε ένα μοντέλο *Χρήστη-Θέματος (Author-Topic),* το οποίο μπορεί να μοντελοποιεί ευέλικτα τους χρήστες που δημοσιεύουν στο Twitter με τις αντίστοιχες κατανομές θεμάτων. Στο πλαίσιο της μελέτης αυτής, παρατηρήθηκε ότι το μοντέλο τους λειτουργεί καλύτερα από τον LDA στην περίπτωση όμως που έχουμε μικρό αριθμό λέξεων στο σύνολο δεδομένων μας.

Τέλος, μια άλλη μελέτη **[Phan, Nguyen & Horiguchi, 2008]** εξέτασε τη μοντελοποίηση μικρών κειμένων μέσω του LDA. Τα βήματα που ακολυθούν την ανάλυση θεμάτων είναι: επιλογή της μεθόδου μηχανικής μάθησης Maximum Entropy (MaxEnt) για ταξινόμηση, ενσωμάτωση των κρυμένων θεμάτων στο σύνολο εκπαίδευσης και τέλος εκπαίδευση του ταξινομητή πάνω στα δεδομένα εκπαίδευσης. Η μελέτη αυτή επικεντρώνεται σε εφαρμογές στη Wikipedia, χωρίς να παρέχεται κάποια πληροφορία σχετικά με άλλους τρόπους εκπαίδευσης ενός παρόμοιου μοντέλου.

Αν και πλούσιες ως προς το περιεχόμενο και τη διορατικότητα, αυτές οι έρευνες δεν παρουσιάζουν αυτόματες ολοκληρωμένες μεθόδους για οργάνωση και κατηγοριοποίηση όλου του δικτύου του Twitter.

4

Ανάλυση και Σχεδίαση

Το σύστημα που κατασκευάσαμε στοχεύει στην αυτοματοποιημένη ομαδοποίηση κειμένων και πιο συγκεκριμένα Tweets, μέσω αλγορίθμων Μηχανικής Μάθησης, σε συστάδες οι οποίες εκφράζουν Θέματα. Σκοπός είναι τόσο τα Θέματα που εξάγονται να εκφράζουν μέσω λέξεων με ρητό τρόπο κάποια άποψη, όσο και η κατάταξη των επιμέρους κειμένων σε αυτά με σωστό τρόπο. Για να επιτευχθούν τα παραπάνω απαιτείται αρχικά η συλλογή δεδομένων και στη συνέχεια η προετοιμασία των δεδομένων για τους αλγόριθμους Μηχανικής Μάθησης, η εφαρμογή των αλγορίθμων αυτών και τέλος η οπτικοποίηση των αποτελεσμάτων προς τον τελικό χρήστη για καλύτερη εξαγωγή συμπερεσμάτων.

4.1 Αρχιτεκτονική

Το σύστημά μας αποτελείται από τα ακόλουθα επιμέρους υποσυστήματα:

1. ***Συλλογή και Αποθήκευση δεδομένων*** από το Twitter στη βάση:
Για την πρόσβαση στη βάση του Twitter χρησιμοποιήθηκαν 3 λογαριασμοί για μεγαλύτερη ταχύτητα κατά τη συλλογή αλλά και για αντιμετώπιση περιορισμών που επέβαλλε το *Twitter* ανά λογαρισμό. Ο κώδικας που επικοινωνεί με το *Twitter* μέσω της βιβλιοθήκης *Tweepy* έκανε χρήση των *APIs* του *Twitter* με βάση κάποιες λέξεις κλειδιά (*Hashtags*) που εμφάνιζαν ενδιαφέρον γύρω τους. Τα Tweets αυτά αποθηκεύονται μαζί με τις σχέσεις και τα χαρακτηριστικά τους στη βάση δεδομένων μας (*Neo4j*). Η διαδικασία ***αποθήκευσης*** διευκολύνεται μέσω της βιβλιοθήκης *py2neo* που επιτρέπει την εργασία και πραγματοποίηση συναλλαγών στη βάση εντός των εφαρμογών σε *Python*.

2. ***Προεπεξεργασία*** των Tweets με χρήση διαφόρων τεχνικών.
Ουσιαστικά εδώ αφαιρέθηκε, όσο είναι δυνατό, ο θόρυβος που περιλαμβάνεται στα Tweets απομονώνοντας την πληροφορία. Οι οντότητες που θεωρήσαμε χρήσιμες είναι τα *hashtags(#)*, τα *mentions(@)* και το *απλό κείμενο ανά Tweet*. Αναφορικά έγινε αφαίρεση των σπάνιων λέξεων καθώς και κάποιων που περιλαμβάνονται σε μια λίστα αποκλεισμού και γενικότερα επιτεύχθηκε μορφοποίηση των λέξεων σε κοινό μοτίβο. Η παραπάνω προεπεξεργασία έγινε σε *Python*.

3. ***Επεξεργασία*** των Tweets που προκύπτουν από την προεπεξεργασία και μιλάνε κάθε φορά για το *Θέμα* της επιλογής μας.

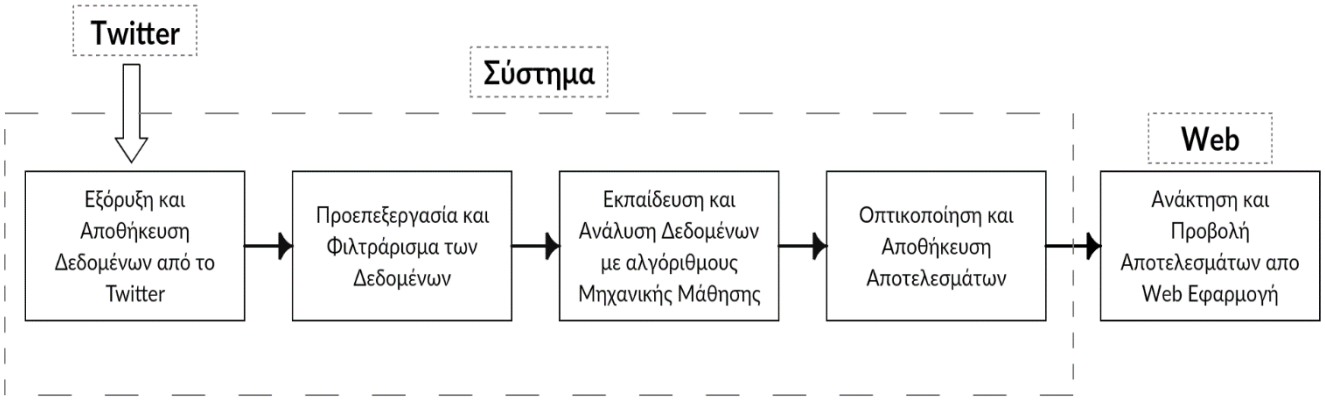
Η επεξεργασία γίνεται μέσω των αλγόριθμων *μηχανικής μάθησης K-Means* και *Latent Dirichlet Allocation* με σκοπό τον διαχωρισμό της πληροφορίας των Tweets σε συστάδες. Για τον κώδικα του *Latent Dirichlet Allocation* έγινε χρήση της βιβλιοθήκης *LDA* της *Python* και για του *K-Means* τα εργαλεία μηχανικής μάθησης που παρέχει η βιβλιοθήκη *Scikit-Learn* της *Python*. Ως είσοδο για τους παραπάνω αλγόριθμους έχουμε είτε τα επιμέρους Tweets είτε τα επιμέρους κείμενα *ανά χρήστη* (το σύνολο των Tweets του).

4. ***Οπτικοποίηση*** των αποτελεσμάτων.
Πραγματοποιείται οπτικοποίηση για τα αποτελέσματα που προκύπτουν από την επεξεργασία μέσω της βιβλιοθήκης *bokehJS* (*Javascript*) η οποία ενσωματώθηκε στην *Python*. Για την οπτικοποίηση κάνουμε πρώτα μείωση διαστάσεων των δεδομένων, όπου χρειάζεται, για αναπαράσταση στο 2-διάστατο επίπεδο.

5. ***Αποθήκευση*** των **αποτελεσμάτων** στη βάση. Τα δεδομένα αποθηκεύονται ώστε να είναι διαθέσιμα για ευκολότερη πρόσβαση στη συνέχεια.

6. Δημιουργία ***Web Εφαρμογής*** (*Flask, Html, Javascript, Css*).
Η Web Εφαρμογή δέχεται είσοδο σχετικά με τις προτιμήσεις του χρήστη πάνω στο θέμα και τον τρόπο ανάλυσης και του επιστρέφει τα αποτελέσματα με βάση τα παραπάνω βήματα τα οποία και του παρουσιάζει με κατανοητό τρόπο

Σχηματικά το σύστημα:



Εικόνα 4.1: Περιγραφή Συστήματος

4.2 Υποσύστημα Εξόρυξης και Αποθήκευσης Δεδομένων

Πρόβλημα

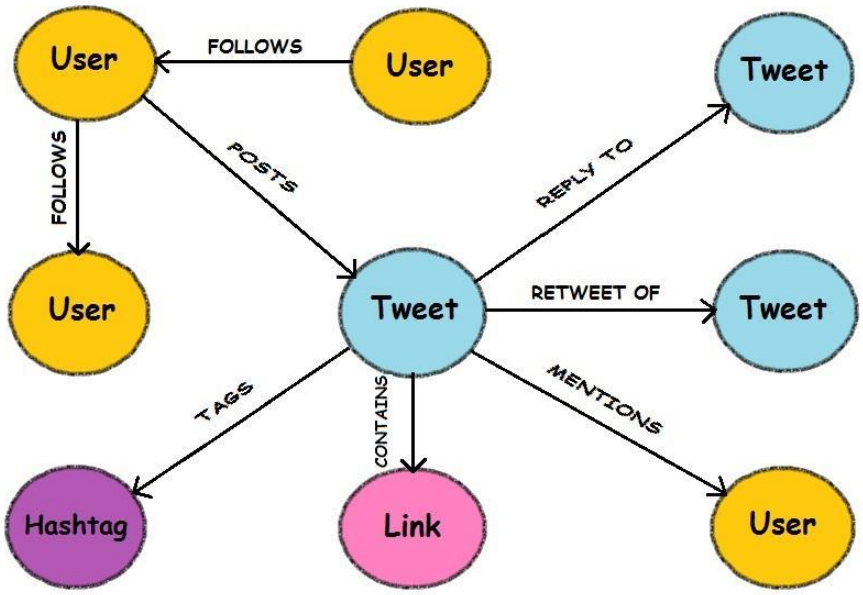
Όπως σε κάθε σύστημα ανάλυσης δεδομένων, έτσι και εδώ καλούμαστε σε πρώτη φάση να αποκτήσουμε πρόσβαση σε μιά εξωτερική βάση δεδομένων για να συλλέξουμε και στη συνέχεια να αποθηκεύσουμε το κατάλληλο υλικό πάνω στο οποίο θα εργαστούμε στη βάση δεδομένων μας. Εδώ επιλέξαμε να χρησιμοποιήσουμε τη λήψη δεδομένων από τη βάση δεδομένων του *Twitter* που είναι από τις πιο "ανοικτές" που προσφέρονται για πρόσβαση και λήψη δεδομένων και τείνει να περιλαμβάνει απόψεις από διάφορες ομάδες αποδίδοντας ολοκληρωμένη πληροφορία.

Μεθοδολογία / Μέθοδοι

Αρχικά χτίζουμε το ***μοντέλο δεδομένων*** *(data model)* με βάση το οποίο θα αποθηκευτούν τα δεδομένα στη μορφή που εμείς επιθυμούμε για επεξεργασία.

Το *μοντέλο* όπως είναι φυσικό αφού στηρίζεται στη λογική των γράφων, αποτελείται από *Κόμβους(Nodes)* και *Σχέσεις(Relationships*). Οι *Users(Χρήστες),* τα *Tweets*, τα *Hashtags* και τα *Links(Σύνδεσμοι)* αναπαριστώνται ως Κόμβοι. Αντίστοιχα οι συνδέσεις "*Follows"*, "*Posts", "Tags",* "*Mentions", "Retweet of", "Reply to", "Contains"* αποτελούν τις Σχέσεις μεταξύ των Κόμβων.

Κάθε κόμβος και ακμή στο δίκτυο είναι μοναδικός και δεικτοδοτείται με έναν αριθμό για ID. Μπορούμε να δούμε στο ακόλουθο διάγραμμα τους κόμβους και πως αυτοί δύναται να συνδέονται μεταξύ τους.



Εικόνα 4.2: Κόμβοι και Ακμές στο δίκτυο μας

Στη συνέχεια καλούμαστε να συλλέξουμε δεδομένα γύρω από κάποιο ***γεγονός***. Ο ευκολότερος τρόπος να επιτευχθεί αυτό στο Twitter είναι με επικέντρωση σε σημαντικά *hashtags(#)* που δημιουργούνται με αφορμή συμβάντα της καθημερινότητας. Επιλέγονται λοιπόν hashtags που θα φέρουν στη βάση όγκο λόγω της δημοφιλίας του εκάστοτε θέματος αλλά και διαφορετικές απόψεις από πολλαπλούς χρήστες.

Η διαδικασία λήψης δεδομένων από το Twitter απαιτεί αρχικά τη διάθεση λογαριασμού. Το Twitter αν και διαθέτει ανοικτή βάση όπως προαναφέραμε, έχει θέσει κάποιους **περιορισμούς** στο API του ως προς τον αριθμό των κλήσεων που μπορούν να γίνουν ανά λογαριασμό (προγραμματιστή) προς τη βάση στο χρονικό παράθυρο των 15 λεπτών. Εμείς για να αντιμετωπίσουμε αυτό το πρόβλημα και να συλλέξουμε όσο πιο πολλά δεδομένα γινόταν ανά

θέμα χρησιμοποιήσαμε 3 λογαριασμούς. Έτσι όσο υπήρχε αναμονή στον έναν λογαριασμό λόγω υπεράριθμων κλήσεων έτρεχε η ροή στον άλλον καλύπτοντας το κενό.

Όπως αναφέραμε και προηγουμένως η **προγραμματιστική πρόσβαση** στα δεδομένα του Twitter κατά την παρούσα μελέτη γίνεται μέσω των *REST APIs* και *Streaming API* που παρέχονται από το Twitter. Η ενσωμάτωση αυτών στα προγράμματα μας έγινε μέσω της *TweePy*, μιας βιβλιοθήκης που επιτρέπει τόσο την πιστοποίηση του χρήστη όσο και την ενσωμάτωση των συναρτήσεων που παρέχουν τα APIs για λήψη των αντικειμένων της επιλογής στον Python κώδικα μας. Η TweePy περιλαμβάνει όλες τις δυνατότητες που παρέχουν τα APIs του Twitter και προτιμήθηκε λόγω της μεγαλύτερης δημοφιλίας της που παρέχει περισσότερο υλικό αλλά και της τοποθέτησης από το ίδιο το Twitter ως προτεινόμενης επιλογής.

Μετά τη συλλογή δεδομένων θα έχει επιτευχθεί:

- Συμπερίληψη απόψεων οι οποίες βασίζονται σε συγκεκριμένα θέματα και συνοψίζονται στα κείμενα του εκάστοτε Tweet
- Κατασκευή ενός ισχυρού δικτύου που θα αποτελεί αντιπροσωπευτικό δείγμα με απορρόφηση των περιφερειακών *Οντοτήτων* που συνοδεύουν το κάθε *Tweet*. Αυτές τις *Οντότητες* μαζί με τις *Σχέσεις* που τις συνδέουν τις είδαμε στο μοντέλο δεδομένων.

Τα *Tweets* αποθηκεύονται στη βάση δεδομένων μαζί με τα αντίστοιχα *χαρακτηριστικά* και τις πιθανές *Σχέσεις* που τα συνδέουν με τους άλλους *Κόμβους* εντός του δικτύου. Για βάση δεδομένων όπως αναφέραμε και πριν χρησιμοποιήσαμε τη *Neo4j* λόγω της μορφής Γράφου που διαθέτει το Twitter ως κοινωνικό δίκτυο.

Η πρόσβαση και η **αποθήκευση των δεδομένων** σε αυτή γίνεται παράλληλα με την διαδικασία συλλογής τους. Δηλαδή κάθε *Οντότητα* και *Σχέση* που φέρνει μαζί του το κάθε Tweet με τα αντίστοιχα χαρακτηριστικά της μετουσιώνεται σε *Οντότητα* και *Σχέση* εντός του Γράφου Δεδομένων μας.

Για να επιτύχουμε αυτό επιλέξαμε τη βιβλιοθήκη *Py2neo* που επιτρέπει την πραγματοποίηση συναλλαγών με τη βάση δεδομένων *Neo4j* μέσω του περιβάλλοντος της *Python*. Έτσι επιτυγχάνεται η συνύπαρξη της συλλογής (*TweePy*) και της αποθήκευσης (*Py2neo*) δεδομένων σε προγράμματα μέσω της γλώσσας Python που αποτελέι την κοινή συνιστώσα τους.

Η διάθεση πληθώρας βιβλιοθηκών και υλικού ήταν και ένας από τους κύριους λόγους που επιλέξαμε την *Python* ως γλώσσα ανάπτυξης γιά την παρούσα διπλωματική

Η συλλογή των δεδομένων που περιστρέφονται γύρω από τα Tweets έγινε τόσο μέσω των συναρτήσεων του *REST API* όσο και των συναρτήσεων του *Streaming API*. Τα *REST APIs* χρησιμοποιήθηκαν για ανάκτηση των Tweets με τα χαρακτηριστικά τους με είσοδο λέξη και ημερομηνία ενδιαφέροντος (*Search API*), ανάκτηση χρήστη με τα γνωρίσματα του (*get_user API*) και ανάκτηση του αρχικού Tweet με τα χαρακτηριστικά του σε περίπτωση που έχουν απάντηση από συγκεκριμένο χρήστη (*get_status API*). Τώρα το *Streaming API* χρησιμοποιήθηκε για ζωντανή συλλογή δεδομένων και χαρακτηριστικών από Tweets που δημοσιεύονται τη στιγμή συλλογής, με κριτήριο τη συμπερίληψη κάποιας λέξης κλειδί της προτίμησης μας σε αυτά.

Παράλληλα προκειμένου να ισχυροποιήσουμε το δίκτυο κάναμε χρήση συναρτήσεων βασισμένων στο *REST API* οι οποίες φέρνουν δεδομένα σχετικά με χρήστες που έχουν ήδη αποθηκευτεί στη βάση δεδομένων μας από τη διαδικασία συλλογής των Tweets. Αυτά τα δεδομένα αφορούν είτε το δίκτυο *Ακολούθων (Followers)* και *Φίλων (Friends)* του κάθε χρήστη (*followers_ids, friends_ids, user_lookup APIs*) είτε το χτίσιμο σχέσεων με βάση τα Tweets που υπάρχουν στο προφίλ (Timeline) του εμπλουτίζοντας το δίκτυο σε βάθος (*user_timeline, get_user, get_status APIs*).

4.3 Υποσύστημα Προεπεξεργασίας των Δεδομένων

Πρόβλημα

Στα *Tweets* που συλλέξαμε κατά ένα μεγάλο ποσοστό περιλαμβάνεται ***"Θόρυβος"***. Ως θόρυβο θεωρούμε το κομμάτι εκείνο του κειμένου που αποτελεί άχρηστη πληροφορία. Στόχος αυτού του υποσυστήματος είναι να εξαλειφθεί η άχρηση πληροφορία και να διατηρηθεί το χρήσιμο κομμάτι των *Tweets*, σε όσο το δυνατόν πιό τυποποιημένη και έτοιμη για επεξεργασία μορφή γίνεται.

Από τις μεθόδους που θα χρησιμοποιήσουμε θα προκύψει για κάθε Tweet μια νέα οντότητα εντός της ροής του προγράμματος που αποτελεί την εξιδανικευμένη μορφή του Tweet μετά την προεπεξεργασία. Το νέο αυτό κείμενο λοιπόν ιδανικά θα περιλαμβάνει μόνο την χρήσιμη πληροφορία που αντιστοιχεί σε κάθε *Tweet* , με βάση την οποία θα προχωρήσουμε στα επόμενα βήματα που αφορούν την ανάλυση δεδομένων. Οι τίτλοι θα είναι ένα μείγμα *απλών λέξεων*,

Hashtags και *Επισημασμένων (Mentioned) Ονομάτων* που παραμένουν στο κάθε *Tweet* μετά την διαδικασία φιλτραρίσματος.

Μεθοδολογία/Μέθοδοι

Όπου αναφέρουμε τον όρο *λέξεις* παρακάτω εννοούμε είτε *Απλό Κείμενο*, είτε *Hashtags* είτε *Επισημασμένα Ονόματα Χρηστών* του *Twitter*. Η **διαδικασία** μετατροπής του κάθε *Tweet* αποτελείται από τα ακόλουθα στάδια:

- Αφαίρεση *λέξεων* που ανήκουν σε μία λίστα αποκλεισμού τις οποίες ονομάζουμε ***Stopwords*** από κάθε *Tweet*.

Στον προγραμματισμό η χρησιμοποίηση *Stopwords* σημαίνει την αφαίρεση επιλεγμένων λέξεων πριν ή μετά την επεξεργασία δεδομένων *φυσικής γλώσσας*. Οι *Stopwords* συνηθίζεται να είναι άρθρα, αντωνυμίες, σύνδεσμοι και άλλες λέξεις της εκάστοτε γλώσσας με φαινομενικά ουδέτερο νόημα. Δεν υπάρχει ιδανική λίστα για κάθε περίπτωση γι'αυτό πολλές φορές καλείται ο προγραμματιστής να προσαρμόσει τη λίστα ώστε να ανταποκρίνεται στη φυσική γλώσσα που περιλαμβάνεται στο εκάστοτε πρόβλημα. Στην περίπτωση μας έχουμε φτιάξει μία λίστα που αποτελεί ένα μίγμα προκαθορισμένων και επιλεγμένων λέξεων, γραμμάτων και συμβόλων προερχόμενων από την ελληνική την αγγλική και την ισπανική γλώσσα που δεν μπορούν να θεωρηθούν ως πληροφορία. Υπήρχαν έτοιμες λίστες με προκαθορισμένες λέξεις αλλά κρίθηκε αναγκαία η προσθήκη και άλλων όρων σε αυτές. Η τροποποίηση της λίστας προέκυψε αφενός με την προσθήκη λέξεων σχετικών με το εκάστοτε θέμα, για να αποφευχθεί η επανάληψη λέξεων που προσδίδουν το ίδιο νόημα με το *Hashtag* αναζήτησης και αφετέρου με την προσθήκη λέξεων, συμβόλων, όρων που είδαμε να εμφανίζονται συχνά στα Tweets, ίσως λόγω της απλής γλώσσας που χρησιμοποιείται στο Twitter, χωρίς να αποτελούν κάποια πληροφορία και δεν υπήρχαν ήδη στη λίστα αποκλεισμού. Η ύπαρξη έντονα χρησιμοποιούμενων λέξεων χωρίς ιδιαίτερο νόημα θα οδηγούσε σε σχηματισμό ανούσιων συστάδων από τους αλγορίθμους μηχανικής μάθησης.

- Μετατροπή κάθε λέξης στην αντίστοιχη ***χωρις Τονισμό***.

Εφόσον μιλάμε για συγκεκριμένα θέματα κάθε φορά, σπάνια θα συναντάμε δημοφιλείς λέξεις εντός αυτών με πολλαπλές έννοιες σε διαφορετικό τονισμό. Οπότε η συνήθης διαφορά στον τονισμό μεταξύ λέξεων οφείλεται στο ότι κάποιοι χρήστες πληκτρολογούν τις λέξεις με τόνους και κάποιοι άλλοι χωρίς. Η αφαίρεση τονισμού από τις λέξεις των κειμένων μας αφορά συνήθως *ελληνικές λέξεις* όταν το αντικείμενο συζήτησης αφορά την ελληνική επικαιρότητα, ενώ σε περίπτωση θεμάτων διεθνής συζήτησης συναντώνται συχνά *ισπανικές λέξεις* που χρήζουν επεξεργασίας. Η αφαίρεση τόνων και παρόμοιων συμβόλων στίξης επιτυγχάνεται με

κανονικοποίηση των επιμέρους χαρακτήρων των λέξεων μέσω της κωδικοποίησης τους (ASCII). Είναι αρκετά σημαντικό για την επεξεργασία και ανάλυση που ακολουθει, οι λέξεις με κοινό νόημα να εμφανίζονται και με κοινή μορφή έτσι ώστε να προσδίδεται μεγαλύτερη βαρύτητα σε αυτές κατά τη δημιουργία συστάδων.

- Αφαίρεση *λέξεων από* το *Λεξιλόγιο (Vocabulary)* οι οποίες παρουσιάζουν ***χαμηλή συχνότητα εμφάνισης*** στο σύνολο των *Tweets* το οποίο προκύπτει για το συγκεκριμένο *Hashtag*.

Γιά το εκάστοτε *Hashtag* προκύπτει ένα *Λεξιλόγιο* που περιλαμβάνει το περιεχόμενο όλων των *Tweets* που μιλάνε γιά αυτό. Σε αυτό το *Λεξιλόγιο* η κάθε λέξη μπορεί να εμφανίζεται πολλαπλές φορές. Αν θεωρήσουμε ότι από τις *λέξεις* που χρησιμοποιούνται προκύπτουν οι απόψεις των *Χρηστών* οδηγούμαστε στο συμπέρασμα ότι *λέξεις* που αναφέρονται ελάχιστες φορές εντός του συνόλου δεν αποτελούν πληροφορία διότι εκφράζουν μεμονωμένη άποψη, και συνεπώς δεν αντιπροσωπεύουν την κοινή γνώμη. Επομένως η συμπερίληψη τους θα αύξανε απλά τον όγκο του συνολικού κειμένου κατά την επεξεργασία. Η *αφαίρεση των λέξεων χαμηλής συχνότητας* επιτυγχάνεται με τη χρήση μια απλής λίστας συχνοτήτων για την κάθε *λέξη* που σε συνδυασμό με κάποιο όριο φορών εμφάνισης που ορίζουμε, αφαιρούν ή όχι την εκάστοτε *λέξη*.

- Αφαίρεση **πολλαπλών κοινών λέξεων** *ανά Tweet*.

Θέλουμε το τελικό *κείμενο* που θα προκύψει να μην περιέχει επαναλαμβανόμενο νόημα. Ένας *χρήστης* έχει περάσει το νόημα ότι μιλάει για κάποιο συγκεκριμένο θέμα με την πρώτη φορά χρήσης κάποιας *λέξης* στο *Tweet* του. Οπότε κρίνουμε την ύπαρξη διπλών, τριπλών κλπ. εμφανίσεων της *λέξης* πλεονασμό γι'αυτό και όταν συναντάμε τέτοιες περιπτώσεις αφαιρούμε τις περιττές *λέξεις*. Έτσι προκύπτει ότι σε κάθε τελικό κείμενο μια *λέξη* μπορεί να υπάρχει το πολύ μια φορά. Το στάδιο αυτό απαιτεί επίσης τη χρήση κάποιας λίστας συχνοτήτων (για κάθε *Tweet*) που αντίθετα με πρίν, που αφαιρούσαμε τις λιγότερο συχνές *λέξεις*, θα αφαιρεί αυτές που εμφανίζονται πάνω από το όριο της μίας φοράς.

Ουσιαστικά το φιλτράρισμα των δεδομένων αποτελεί μια προεξεργασία τους πριν αυτά περάσουν στο κύριο στάδιο επεξεργασίας και ανάλυσης και αποτελέσουν είσοδο των αλγόριθμων: *μοντελοποίησης Θεμάτων LDA* και *συσταδοποίησης Κειμένων K-Means*. Έτσι μέσω της κατάλληλης διαμόρφωσης του περιεχομένου εισόδου διασφαλίζεται η βελτιστοποίηση των αποτελεσμάτων που αυτοί εν τέλει θα παράγουν.

Οπότε τα τελικά *Tweets* στη μορφή μείγματος *Λέξεων(Words), Hashtags (#hashtag)* και *Επισημασμένων Ονομάτων Χρηστών(Mentioned Screen Names) (@screen_name)* που

παρέμειναν, θα αποτελέσουν την είσοδο για τους αλγόριθμους που χρησιμοποιήσαμε για επεξεργασία και ανάλυση των δεδομένων.

4.4 Υποσύστημα Επεξεργασίας και Ανάλυσης Δεδομένων

4.4.1 Σκοπός

Πρόβλημα

Ο **σκοπός** μας είναι η δημιουργία συστάδων όπου η καθεμία θα εκφράζει μια θέση επί του *Θέματος* συζήτησης που επιλέξαμε (μέσω *Hashtag*) , η κατάταξη των Κειμένων σε αυτές και η εξαγωγή χρήσιμων συμπερασμάτων.

Τα *Tweets* που συγκεντρώσαμε με βάση το εκάστοτε *Hashtag*, θα αναπαρασταθούν στο δισδιάστατο επίπεδο με τα πιο όμοια να βρίσκονται κοντά γεωμετρικά ενώ τα ανόμοια να παρουσιάζονται απομακρυσμένα. Με αυτόν τον τρόπο επιτρέπεται εύκολη και γρήγορη εξαγωγή αναλυτικών συμπερασμάτων από κάποιον παρατηρητή.

Μέθοδοι

Για να αναπαραστήσουμε τα όμοια *κείμενα* κοντά και πιο συγκεκριμένα σε ομάδες χρησιμοποιήσαμε μοντέλα όπως η Συσταδοποίηση *K-Means* και το παραγωγικό μοντέλο *LDA (Latent Dirichlet Allocation).* Η χρησιμοποίηση των παραπάνω μοντέλων επιτρέπει αναπαράσταση των Θεμάτων μέσω χαρακτηριστικών λέξεων και επιτυχημένη ανάθεση Θέματος στο κάθε κείμενο. Πρίν από την εκπαίδευση με βάση αυτά τα μοντέλα και μετά το προεπεξεργαστικό στάδιο που αναφέραμε προηγουμένως, μετατρέπουμε τα *κείμενα* σε διανυσματική μορφή με χρήση ενός μοντέλου διανυσματικού χώρου (*Vector Space Model - VSM)* όπως είναι το *TF-IDF*. Ουσιαστικά δημιουργείται ένας πίνακας συχνότητας εμφάνισης λέξεων ο οποίος αποτελεί και την είσοδο των αλγορίθμων μηχανικής μάθησης. Στη συνέχεια γίνεται χρήση του αλγόριθμου μείωσης διάστασης *t-SNE* για να δωθεί σε κάθε *κείμενο* μια συντεταγμένη (x,y) στο 2-διάστατο επίπεδο, έτσι ώστε να μας επιτραπεί η απεικόνιση τους σε διάγραμμα διασποράς στο επίπεδο. Στο διάγραμμα διασποράς κάθε *Κείμενο(Tweet)* αντιστοιχεί σε κάποιο σημείο με κριτήριο τον αλγόριθμο που χρησιμοποιήσαμε προηγουμένως.

4.4.2 Παραμετροποίηση

4.4.2.1 *Καθορισμός K-means*

Ο K-Means όπως είδαμε και πριν, είναι ένας δημοφιλής αλγόριθμος συσταδοποίησης ο οποίος στοχεύει στον διαχωρισμό *n* αντικειμένων σε *k* συστάδες οι οποίες παρουσιάζονται γύρω από ένα σημείο το οποίο αποκαλούμε κέντρο. Κάθε αντικείμενο λέμε ότι ανήκει στην συστάδα με το κοντινότερο σ'αυτό κέντρο.

Στην παρούσα μελέτη χρησιμοποιήσαμε τον *Mini Batch K-Means* (της βιβλιοθήκης scikit-learn για μηχανική μάθηση σε *Python*) ο οποίος αποτελεί μια ταχύτερη παραλλαγή του *K-Means* με τη διαφορά ότι η επεξεργασία των δεδομένων γίνεται αντί για μεμονωμένα, σε μικρές παρτίδες **[K]**. Ο συγκεκριμένος αλγόριθμος δεν χρησιμοποιεί όλο τον όγκο των δεδομένων σε κάθε επανάληψη αλλά ένα υποσύνολο αυτών συγκεκριμένου μεγέθους που ορίζουμε εμείς. Με τον τρόπο αυτό μειώνεται ο αριθμός των υπολογισμών αποστάσεων ανά επανάληψη και έτσι μειώνεται και το υπολογιστικό κόστος. Το κέρδος που έχουμε σε υπολογιστικό κόστος σημαίνει ωστόσο και χειρότερη ποιότητα στη συσταδοποίηση που ωστόσο στην περίπτωση μας δεν είχε ουσιαστική διαφορά.

Ενδεικτικά αναφέρουμε τους χρόνους για τους δύο αλγόριθμους όπως τους πήραμε για 20 συστάδες :

MiniBatch Kmeans: 0.24599981308 seconds

Kmeans: 24.106000185 seconds

Αρχικά θα μετατρέψουμε τα δεδομένα μας σε διανύσματα, με *tf-idf τιμές* που δείχνουν πόσο συχνά εμφανιζόμενη ή σπάνια είναι η κάθε λέξη στα δεδομένα μας, λαμβάνοντας υπόψη ολόκληρη τη συλλογή κειμένων. Με βάση αυτό επιλέγουμε να αναπαραστήσουμε κάθε κείμενο ως ένα 5000-διάστατο διάνυσμα, οι δείκτες του οποίου αντιστοιχούν στους 5 χιλιάδες πιο συχνά εμφανιζόμενους όρους στη συλλογή κειμένων μας. Η παραπάνω αναπαράσταση αποτελεί και την είσοδο του K-Means.

Όσον αφορά τις **παραμέτρους** καλούμαστε να επιλέξουμε τον αριθμό κεντροειδών οπότε και συστάδων που δημιουργούνται γύρω από αυτά. Στόχος μας είναι να πετύχουμε τον καλύτερο διαχωρισμό των Κειμένων σε Συστάδες.

Για την επιλογή του αριθμού Συστάδων *k* που θα δημιουργηθούν χρησιμοποιήσαμε μια μετρική που μας παρέχει η ίδια βιβλιοθήκη, την *μέση συνιστώσα Silhouette (Silhouette Coefficient)*.

Αυτή η μετρική μας πληροφορεί για το πόσο καλά ανήκει το κάθε αντικείμενο στη συστάδα του.

Ο αλγόριθμος τρέχει για μερικές εκατοντάδες επαναλήψεις μέχρι τα κεντροειδή να μην βελτιώνονται άλλο. Τότε παίρνουμε έναν πίνακα με διαστάσεις *#Κειμένων x #Συστάδων* που μας παρέχει την απόσταση του κάθε *Κειμένου* προς κάθε κέντρο. Από δω μπορούμε να καταλήξουμε ποιό είναι το κοντινότερο κέντρο για κάθε κείμενο, στο οποίο θα λέμε ότι ανήκει.

Για κάθε μια από τις *k* Συστάδες επιλέγουμε να εμφανισουμε τις 8 κορυφαίες λέξεις. Αυτό μας επιτρέπει να αποκτήσουμε καλύτερη αντίληψη για το τι περιέχεται σε κάθε Συστάδα και επομένως τι αυτή εκφράζει.

Σε δοκιμές με παρόμοιες αρχικές συνθήκες παρατηρήσαμε σταθερότητα στα αποτελέσματα του αλγόριθμου, κάτι το οποίο δείχνει ότι τα δεδομένα μας συσταδοποιούνται καλά με τον K-Means.

4.4.2.2 *Καθορισμός LDA*

Για μοντελοποίηση Θεμάτων Συζήτησης (Topic Modeling) κάναμε χρήση του μοντέλου *Latent Dirichlet Allocation* ή αλλιώς *LDA* (μέσω της βιβλιοθήκης *lda* της Python), αρχικά για να εντοπίσουμε τα Θέματα Συζήτησης που προκύπτουν από τα Tweets που συλλέξαμε και έπειτα για να βρούμε τις κατανομές των *Θεμάτων* (Topics) σε κάθε *κείμενο* και να ομαδοποιήσουμε τα όμοια εξ'αυτών. Η συγκεκριμένη υλοποίηση κάνει χρήση της καταρρεσμένης δειγματοληψίας Gibbs **[Griffiths & Steyvers, 2004]**.

Αρχικά μετατρέπουμε τα δεδομένα μας σε διανύσματα, αναπαριστώντας κάθε κείμενο ως ένα 5000-διάστατο διάνυσμα οι δείκτες του οποίου αντιστοιχούν στους 5 χιλιάδες πιο συχνά εμφανιζόμενους όρους στη συλλογή κειμένων μας.

Έπειτα τροφοδοτούμε τον πίνακα με διαστάσεις *#Κειμένων x 5,000* που προέκυψε από τον *LDA* για να αναγνωρίσει τα κρυμμένα (Latent) Θέματα που βρίσκονται στα δεδομένα μας. Όσον αφορά τις **παραμέτρους**, όπως μας επιτρέπει ο *LDA*, επιλέγουμε τον αριθμό των θεμάτων που θέλουμε να ανακαλύψουμε εξ' αρχής καθώς και τον αριθμό των επαναλήψεων που θα τρέξει ο αλγόριθμος. Σχετικά με τον αριθμό των προς ανακάλυψη *Θεμάτων* δοκιμάσαμε και εδώ ποικίλες τιμές και είδαμε ότι: μικρές τιμές (<10) οδηγούν σε ετερογενές σύνολο *λέξεων*, ενώ με μεγάλες τιμές διαχέεται η πληροφορία και παρουσιάζονται πολλά Θέματα με το ίδιο νόημα. Με βάση τα παραπάνω, ο *LDA* δείχνει να παρουσιάζει αρκετά καλά αποτελέσματα για 10 Θέματα και πάνω, οπότε θα επιλέξουμε αριθμό Θεμάτων ανάλογο με τον αριθμό συστάδων που διαλέξαμε στον *K-Means* για να διευκολύνουμε την σύγκριση. Όσον αφορά την επιλογή του αριθμού των επαναλήψεων που επιθυμούμε να τρέξει ο *LDA*, παρατηρήσαμε οτι δεν προκύπτει κάποια βελτίωση της σύγκλισης μετά τις 2,000 επαναλήψεις οπότε και επιλέξαμε

αυτές ως όριο. Άλλες κύριες παράμετροι που έχει ο LDA είναι η *α (Alpha)* και η *β (Beta)*. Η παράμετρος *Alpha* καθορίζει πόσα Θέματα ενδέχεται να έχει κάθε κείμενο. Όσο πιο χαμηλή η τιμή της τόσο πιο λίγα τα θέματα στα κείμενα. Η παράμετρος *Beta* με τη σειρά της καθορίζει τον αριθμό των *λέξεων ανά θέμα*. Όπως και στην *Alpha* έτσι και δω, όσο χαμηλότερη η τιμή της, τόσο πιο λίγες οι *λέξεις ανά θέμα*. Εφόσον έχουμε να κάνουμε με Tweets, θεωρούμε ότι ο κάθε χρήστης έχει περιορισμένο αριθμό Θεμάτων για τα οποία μιλάει, ως εκ τούτου ορίσαμε την *Alpha* με μικρη τιμή 0.1 ενώ τη *Beta* την αφήσαμε στην προεπιλεγμένη τιμή (0.01).

Ως αποτέλεσμα αναγνωρίζουμε τα Θέματα που επιθυμούμε και παίρνουμε έναν πίνακα με διαστάσεις *#Κειμένων x #Θεμάτων που δείχνει* την κατανομή των θεμάτων στα κείμενα μας. Από αυτόν τον πίνακα μπορούμε να δούμε και σε ποιο Θέμα κατατάσσεται το κάθε κείμενο (αυτό με το μεγαλύτερο ποσοστό).

Για κάθε ένα από τα *n Θέματα* επιλέγουμε να εμφανίσουμε τις 8 κορυφαίες λέξεις, δηλαδή τις πιο σχετικές ανά *Θέμα*. Αυτό μας επιτρέπει να αποκτήσουμε καλύτερη αντίληψη για το τι περιέχεται σε κάθε Θέμα και επομένως τι αυτό εκφράζει.

4.4.2.3 *Οπτικοποίηση*

Για την οπτικοποίηση κάναμε χρήση της διαδραστικής βιβλιοθήκης *Bokeh* που βασίζεται σε Python με στόχο την παρουσίαση ενός διαγράμματος διασποράς που θα απεικονίζει γεωμετρικά την απόσταση των επιμέρους αντικειμένων, με πρόσθετη πληροφορία το χρώμα τους που θα δείχνει σε ποια *Συστάδα* ή ποιο *Θέμα*, αντίστοιχα για *K-Means* και *LDA* ανήκουν.

Παρακάτω βλέπουμε την παλέτα 20 χρωμάτων που χρησιμοποιήσαμε για το διαχωρισμό σε συνδυασμό με τον αριθμό Συστάδας/Θέματος που αντιστοιχεί:

Εικόνα 4.3: Παλέτα Χρωμάτων

Για τα επόμενα βήματα επιδεικνύουμε αποτελέσματα που βασίζονται σε Tweets που περιέχουν το Hashtag ***#dimopsifisma*** και επομένως επικεντρώνονται σε απόψεις γύρω από αυτό.

Για να πάρουμε μια ιδέα σχετικά με τις Δημοφιλείς *λέξεις* εντός του συνόλου *Κειμένων* μας γύρω από το προαναφερθέν *Hashtag*, παρουσιάζουμε παρακάτω το ***Σύννεφο Λέξεων (Word Cloud)*** που παράχθηκε από αυτό*:*

Εικόνα 4.4: Σύννεφο Λέξεων από τα Tweets με Hashtag #dimopsifisma

4.4.3 Ομαδοποίηση των Tweets

Σε αυτό το κομμάτι τα επιμέρους κείμενα με βάση τα οποία προχωρήσαμε στην ομαδοποίηση είναι τα Tweets τα οποία έχουν περάσει το προεπεξεργαστικό στάδιο.

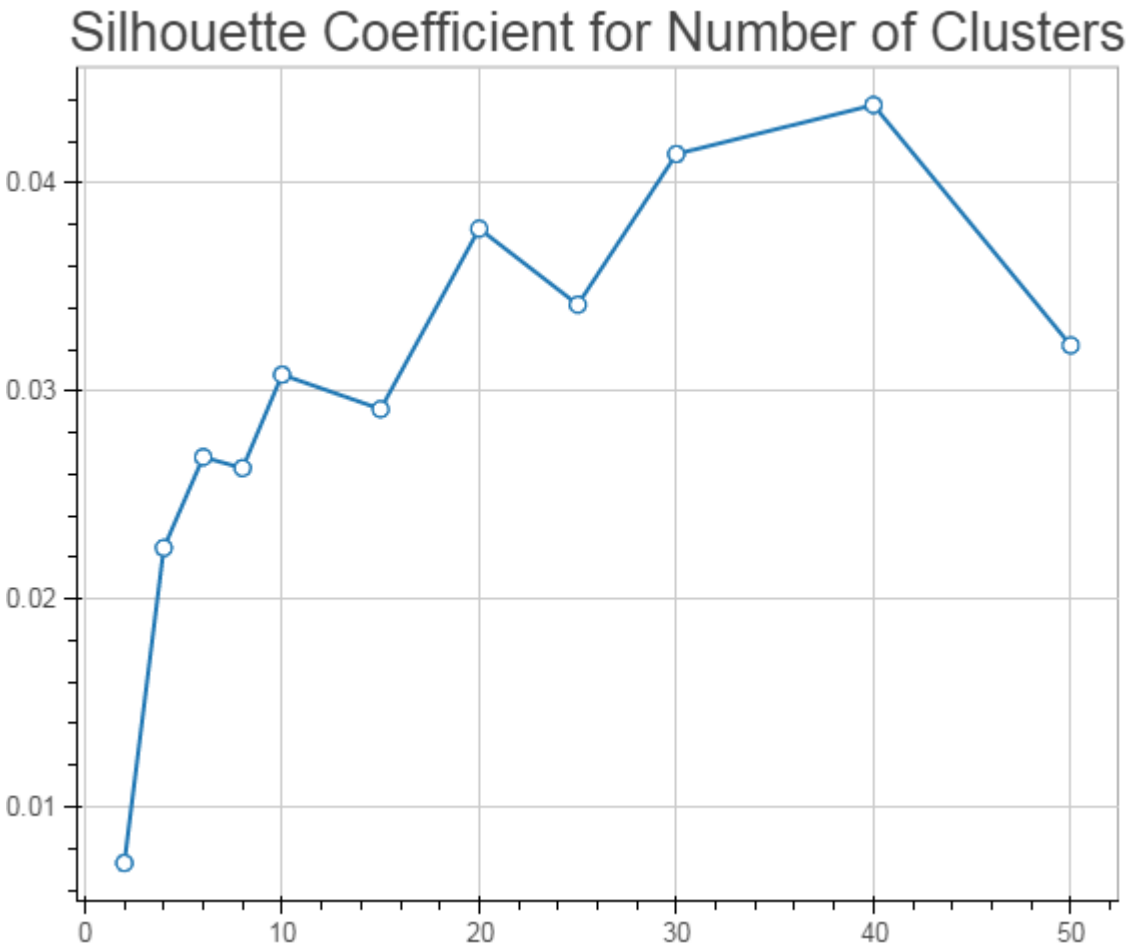
Μεθοδολογία

4.4.3.1 *Συσταδοποίηση Κειμένου K-Means*

Θα δοκιμάσουμε κατά σειρά διαφορετικό πλήθος κέντρων/συστάδων k, με τις τιμές του k να ξεκινάνε από 2 και να φτάνουν έως και 30 έτσι ώστε να επιλέξουμε τον κατάλληλο αριθμό συστάδων.

Γι'αυτό το σκοπό όπως αναφέραμε και παραπάνω χρησιμοποιήσαμε τη συνιστώσα *Silhouette* την οποία και υπολογίσαμε για τους ποικίλους αριθμούς συστάδων: 2,4,6,8,10,15,20,25,30.

Εμείς όπως βλέπουμε και στο παρακάτω **διάγραμμα** παίρνουμε τιμές που είναι αρκετά κοντά στο 0. Αυτό συμβαίνει διότι οι συστάδες που έχουν δημιουργηθεί παρουσιάζουν μεταξύ τους αρκετές κοινές λέξεις κλειδιά οι οποίες συγκεντρώνουν αρκετά μεγάλο όγκο δεδομένων οπότε αρκετά Tweets κινούνται μεταξύ αυτών και τελικά καταλήγουν σε αυτό που κλίνουν λίγο πιο πολύ.



Εικόνα 4.5: Silhouette Coefficient για μεταβλητό αριθμό συστάδων στη συσταδοποίηση Tweets

Παρατηρούμε οτι για αριθμό συστάδων 20 και πάνω πετυχαίνουμε τις ψηλότερες δυνατές τιμές. Θα επιλέξουμε 20 συστάδες για να αποφύγουμε όσο γίνεται τις επαναλήψεις συστάδων με το ίδιο νόημα (*overfitting*).

Όπως είδαμε και προηγουμένως μετά το τρέξιμο του K-Means μας παρέχεται για κάθε κείμενο, που σε αυτή την περίπτωση είναι ένα tweet, το κοντινότερο κέντρο συστάδας και τόσο η απόσταση προς αυτό όσο και προς όλα τα άλλα κέντρα.

Για να καταλάβουμε καλύτερα τι βρίσκεται σε κάθε συστάδα παίρνουμε τα 8 χαρακτηριστικά (λέξεις) για κάθε μία από τις **20 συστάδες** μας:

Συστάδα	**Λέξεις**
0	*greececrisis elladaoramhden ελληνες μερες greece enikos eurogroup οπως*
1	*xreokopia grexit nai onlinemega greekmediapropaganda naistinellada greececrisis martinschulz*
2	*οχι oxi ελλαδα tsipras ναι ατμ ευρω grexit*
3	*grexit greececrisis tellmedad istandwithgreece oxi tsipras οχι vouli*
4	*skai_xeftiles mega_xeftiles europe yes greferentum ant1_xeftiles calm gr*
5	*σαμαρας samaras αντωνης παραιτηθηκε οχι νδ oxi oxi2015*
6	*syriza tsipras grexit oxi οχι syriza_xeftiles συντροφοι kke*
7	*ερευνα διαδικτυακη greekcrisis 2015 απθ πανεπιστημιο ιουλιου θεσσαλονικης*
8	*oxi2015 oxi οχι lemeoxi team_oxi syntagma greececrisis tsipras*
9	*αποφαση δυσκολη oxi σωστη παρουμε συνεπειες λαθος nai*
10	*τσιπρα προταση συμφωνια γιουνκερ θεσμων νεα οχι tsipras*
11	*vouli κκε οχι tsipras ζωη τσιπρας αλεξη grexit*

12	*yeseurope oxi yesgreece oxi2015 nai ναι μνημονιακε παπια*
13	*grefenderum oxi2015 oxi greekreferendum οχι samaras greececrisis ηπιαμε*
14	*oxi οχι freedomordeath πανευρωπαικο υπογραψε πλευρο χιλιαδες λιτοτητα*
15	*πισω γκρεμος ρεμα μπρος βημα οχι πορτα oxi*
16	*nai oxi grineuro οχι oxi2015 naistinellada xreokopia πειραμα*
17	*menoumeevropi syntagma nai οριστικα οχι τελος menoume_malakes tsipras*
18	*capitalcontrols grexit greececrisis bankrun eurogroup eurozone atm europe*
19	*κυβερνηση οχι συριζα σταυρος εθνικης αντιπολιτευση λαο παραιτηθει*

Εικόνα 4.6: Οι 8 επικρατέστερες λέξεις των Συστάδων που προέκυψαν από τα Tweets με K-Means

Όπως φαίνεται υπάρχουν δείγματα σωστού διαχωρισμού ανάμεσα στις συστάδες μας.

Για παράδειγμα μπορούμε να διακρίνουμε κάποιες απόψεις/τάσεις που διαμορφώνονται στις παραπάνω συστάδες:

Συστάδα 1: Πέρα των γενικών αναφορών δείχνει να στηρίζει το ΝΑΙ μέσω αναφορών όπως *naistinellada,nai* αλλά και καταστάσεων που είχαν συνδυαστεί με τα αποτελέσματα σε αντίθετη περίπτωση όπως *grexit* και *xreokopia*.

Συστάδα 4: Εκφράζει τάση κατά των καναλιών με επιμέρους λέξεις κλειδιά όπως *skai_xeftiles mega_xeftiles, ant1_xeftiles*.

Συστάδες 8,14: Σχετίζεται με το ΟΧΙ μέσω λέξεων κλειδιά που το περιλαμβάνουν (*oxi2015, oxi, οχι, lemeoxi, team_oxi* κλπ) με αναφορές στη διαδήλωση που έγινε στο Σύνταγμα υπέρ του όχι (*syntagma, Σύνταγμα*) και στα μνημόνια.

Συστάδα 18: Γενικότερη αναφορά στην κατάσταση με συγκέντρωση λέξεων όπως *capitalcontrols, greececrisis, atm, eurogroup.*

Ας δούμε πως έγινε η **κατάταξη** σε **δείγμα** των **10 Tweets**:

#	Tweet	Συστάδα
1	*ψηφισες «ναι» #mnimonio3 #lafazanis #vouli #NAI #OXI*	15
2	*εικονα σημερινης οχι μπει σπιτι να καταλαβουν λαος νικαει φοβο*	2
3	*ευχηθουμε αυριο ακυρωσει φαρσα διχαζει #ant1_news*	2
4	*1000 greeks #democracy #Grefenderum #austerity #OXI @tsipras_eu @SyrizaLondon @Arhsx*	13
5	*liberation νομπελ οικονομιας οχι @zappasaspa @FREEDYBRUNA*	2
6	*εκανα αναλυση twitter facebook βρισκουν γκαλοπ δειχνω #OXI #OXI #NAI*	15
7	*keep calm #no #yes #Europe #euro #Greferentum #gr*	4
8	*βρισκω βαρουφακης βγει μεση*	2
9	*liberation νομπελ οικονομιας οχι @olympiada*	2
10	*ζωη αρθρο 44 #vouli*	11

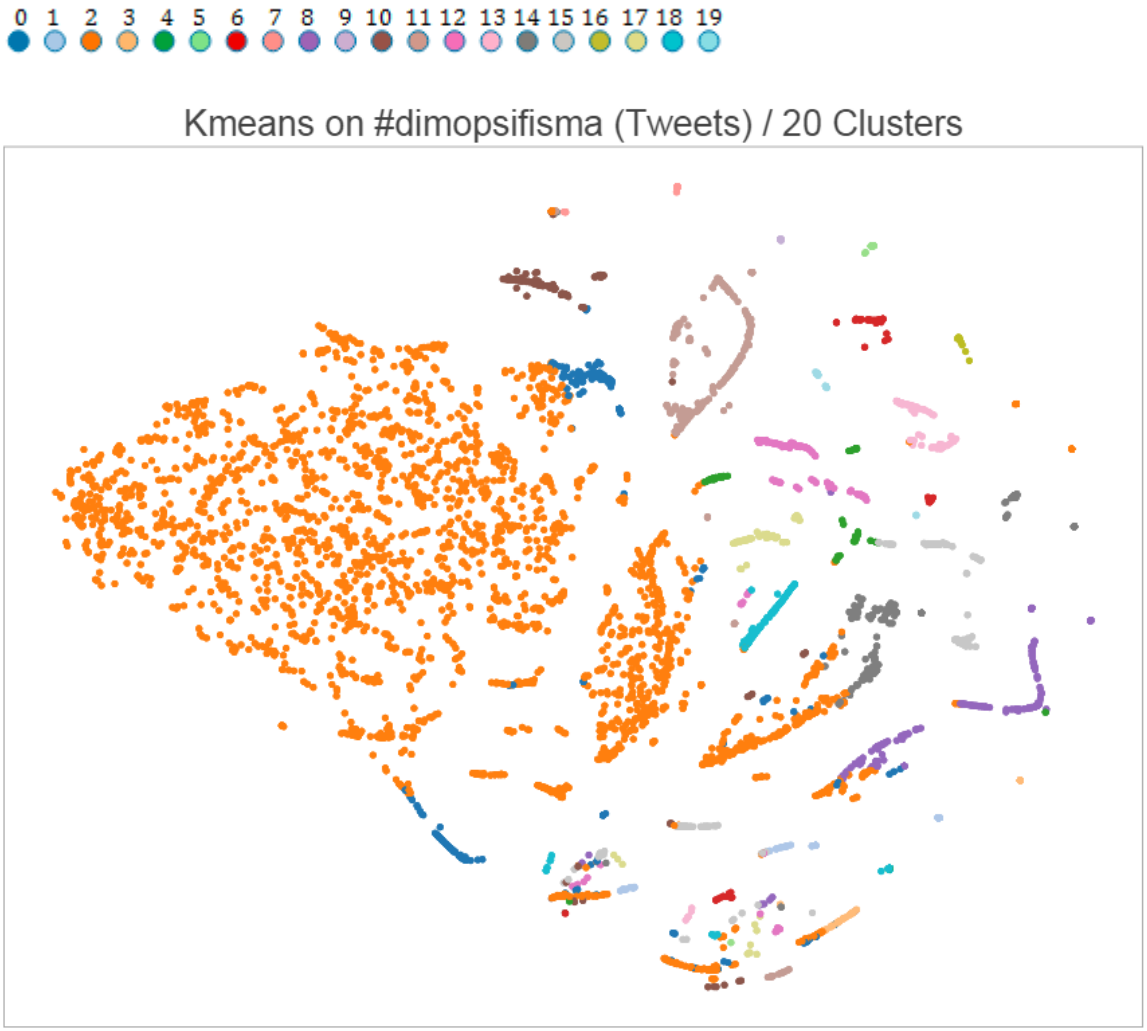
Εικόνα 4.7: Δείγμα Κατάταξης 10 Tweets με το μοντέλο K-Means

Όσον αφορά το κομμάτι της **οπτικοποίησης** παίρνουμε δείγμα 5,000 tweets και εφόσον έχουμε 20 συστάδες για να οπτικοποίησουμε την παραπάνω συσχέτιση απαιτείται κάποια μείωση διάστασης.

Γιά αυτό το σκοπό έγινε χρήση του t-SNE για μείωση της διάστασης από 20 (που ήταν ο αριθμός συστάδων/κέντρων) σε 2 για να δούμε στο 2-διάστατο επίπεδο το αποτέλεσμα.

Για να φαίνεται καλύτερα ο διαχωρισμός μεταξύ συστάδων, θα χρωματίσουμε κάθε tweet ανάλογα με τη συστάδα στην αυτό οποία ανήκει.

Η οπτικοποίηση λοιπόν των tweets, με βάση την απόσταση τους από τα κέντρα των *k* συστάδων μας φαίνεται παρακάτω:



Εικόνα 4.8: Διάγραμμα διασποράς των tweets για 20 συστάδες που προκύπτει με τη συσταδοποίηση K-Means

Ο **διαχωρισμός** δείχνει να είναι καλός. Όπως ήταν αναμενόμενο παρουσιάζονται βέβαια κάποιες επικαλύψεις στα tweets διαφορετικών συστάδων και αυτό αποδίδεται στο γεγονός ότι παρατηρούνται αρκετές κοινές λέξεις κλειδιά. Γενικά παρατηρείται να χτίζονται περιοχές από tweets γύρω από λέξεις (κειμένου, hashtag, mentions) των οποίων έγινε εκτεταμένη χρήση από τους χρήστες και έτσι συγκέντρωσαν μεγάλο όγκο δεδομένων.

4.4.3.2 *Μοντελοποίηση Θεμάτων LDA*

Όπως αναφέραμε και προηγουμένως επιλέγουμε κατά την μοντελοποίηση Θεμάτων Συζήτησης που θα κάνουμε με τον LDA να διατηρήσουμε τον αριθμό των Θεμάτων σε 20 για να υπάρχει σύγκριση με τα αντίστοιχα αποτελέσματα που προέκυψαν από τον K-Means.

Μετά λοιπόν τις 2,000 επαναλήψεις στον αλγόριθμο έχουμε πάρει έναν πίνακα με διαστάσεις *#Κειμένων x #20 που δείχνει* την κατανομή των θεμάτων στα κείμενα μας και θα χρησιμοποιηθεί ύστερα για την οπτικοποίηση.

Για να δούμε πιο λεπτομερώς τι εμπεριέχεται σε κάθε ένα από τα **20 Θέματα Συζήτησης (Topics)** που εντοπίστηκαν από τον LDA, βλέπουμε τις 8 πιο σχετικές λέξεις για το κάθε ένα:

Θέμα	**Λέξεις**
0	*oxi europe yes greek euro grexit greececrisis greece*
1	*oxi skai_xeftiles mega skai οχι μμε προπαγανδα mega_xeftiles*
2	*oxi nai oxi2015 ελλαδας ευρωπαιοι λιτοτητα χιλιαδες πλευρο*
3	*οχι ευρω ευρωπη ναι ελλαδα yeseurope nai δραχμη*
4	*live ελλαδα ελληνες γιουνκερ μερκελ δντ τσιπρα eurogroup*
5	*vouli οχι αντε χρονια παιδια oxi xreokopia μαλακα*
6	*οχι oxi oxi2015 συνταγμα σημερα ευρωπη οχι2015 ελλαδα*
7	*vouli tsipras οχι αλεξη ζωη τελος syriza atsipras*
8	*τραπεζες λαος οχι αυριο κυβερνηση τσιπρας κλειστες δευτερα*
9	*οχι μνημονιο vouli κυβερνηση συμφωνια τελικα συριζα τσιπρας*
10	*ατμ atm capitalcontrols λεφτα ευρω grexit παω ουρες*

11	*syriza_xeftiles οχι yeseurope syriza syriza_apateones grexit συριζα χωρα*
12	*2015 greekcrisis ερευνα διαδικτυακη youtube oxi 15 30*
13	*οχι λαο υπαρχει δημοψηφισματος ενημερωση θεση εναν χωρα*
14	*γιουνκερ προταση enikos βαρουφακης τσιπρας συμφωνια υπαρχει capitalcontrol*
15	*οχι κκε elladaoramhden enikos vouli κυριακη απλα αυριο*
16	*grexit xreokopia greececrisis capitalcontrols eurogroup nai yeseurope eurozone*
17	*oxi oxi2015 nai team_oxi skai_xeftiles yeseurope mega_xeftiles lemeoxi*
18	*οχι ελληνες ψηφισουν χρονια θελουν πανε βγει αλλοι*
19	*samaras grefenderum oxi οχι σαμαρας νδ σαμαρα oxi2015*

Εικόνα 4.9: Οι 8 επικρατέστερες λέξεις των Θεμάτων που προέκυψαν από τα Tweets με LDA

Η διάκριση των απόψεων στα επιμέρους Θέματα είναι πιό ξεκάθαρη από αυτή που είδαμε στις συστάδες πρίν. Για παράδειγμα ας δούμε ορισμένα Θέματα:

Θέματα 1,17: Εκφράζουν τάση κατά των καναλιών με επιμέρους λέξεις κλειδιά όπως *skai_xeftiles, skai, mega_xeftiles, mega,,* αλλά και πιο γενικά *προπαγάνδα.* Δείχνει να συνδυάζεται και με το ΟΧΙ (*oxi,oxi2015,lemeoxi*) το οποίο είναι πολύ λογικό.

Θέματα 3,16:Πέρα των γενικών αναφορών δείχνει να στηρίζει το ΝΑΙ μέσω αναφορών όπως ναι,*nai,yeseurope* αλλά και αρνητικών καταστάσεων που είχαν συνδυαστεί με τα αποτελέσματα σε αντίθετη περίπτωση όπως δραχμή *grexit* και *xreokopia*.

Θέμα 4: Μιλά για τις εξωτερικές εξελίξεις με αναφορές σε ξένους ηγέτες όπως*: Γιούνκερ, Μέρκελ* καθώς και στο *eurogroup* και *δντ.*

Θέμα 6: Τάση υπέρ του *ΟΧΙ* με λέξεις όπως*: oxi, oxi2015, team_oxi, οχι* καθώς και αναφορές στον προθυπουργό *(tsipras),* στην υπέρ του ΟΧΙ διαδήλωση *(syntagma)* και στη *λιτότητα.*

Θέματα 8,10: Προβληματισμοί σχετικά με τις τράπεζες που φαίνονται μέσω λέξεων όπως *τραπεζες,atm,λεφτα, κλειστές, capitalcontrols.*

Θέμα 11: Τάση κατά του Σύριζα *(syriza_xeftiles, syriza_apateones)*

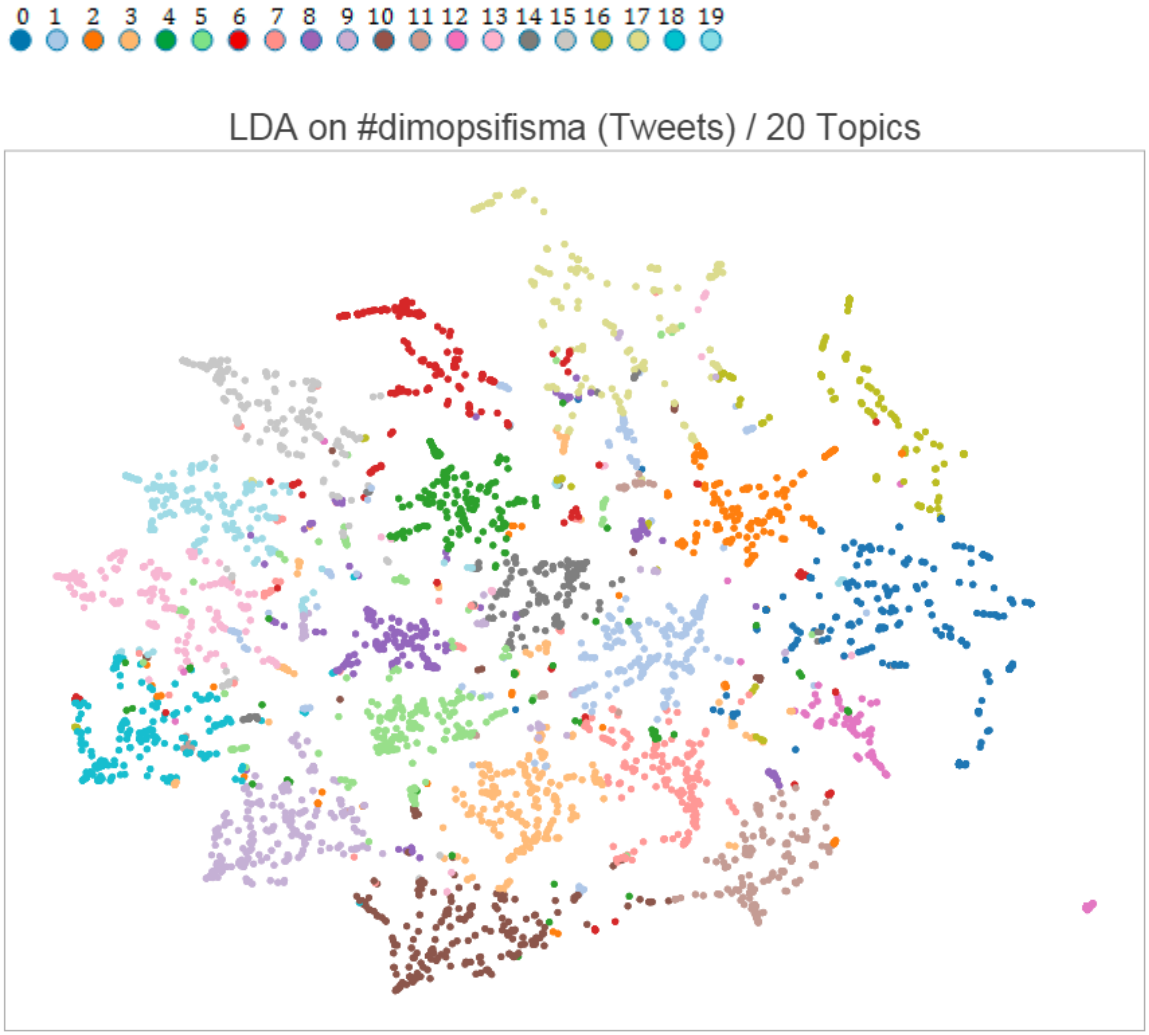
Θέμα 19: Αναφέρεται κυρίως στο κόμμα της Νέας Δημοκρατίας με αναφορές στον πρόεδρο της (*samaras, σαμαρας, σαμαρα*) αλλά και στο ίδιο το κόμμα (*νδ*) με συμπερίληψη λέξεων υπέρ του ΟΧΙ (*οχι,oxi,oxi2015*)

Ας δούμε πως έγινε η **κατάταξη** σε **δείγμα** των **10 Tweets**:

#	Tweet	Θέμα
1	*ψηφισες «ναι» #mnimonio3 #lafazanis #vouli #NAI #OXI*	2
2	*εικονα σημερινης οχι μπει σπιτι να καταλαβουν λαος νικαει φοβο*	18
3	*ευχηθουμε αυριο ακυρωσει φαρσα διχαζει #ant1_news*	8
4	*1000 greeks #democracy #Grefenderum #austerity #OXI @tsipras_eu @SyrizaLondon @Arhsx*	0
5	*liberation νομπελ οικονομιας οχι @zappasaspa @FREEDYBRUNA*	11
6	*εκανα αναλυση twitter facebook βρισκουν γκαλοπ δειχνω #OXI #OXI #NAI*	5
7	*keep calm #no #yes #Europe #euro #Greferentum #gr*	0
8	*βρισκω βαρουφακης βγει μεση*	9
9	*liberation νομπελ οικονομιας οχι @olympiada*	11
10	*ζωη αρθρο 44 #vouli*	7

Εικόνα 4.10: Δείγμα Κατάταξης 10 Tweets με το μοντέλο LDA

Για την **οπτικοποίηση** των αποτελεσμάτων αντίστοιχα εργαζόμαστε και εδώ με χρήση του αλγόριθμου *t-SNE* για μείωση διαστάσεων από 20 σε 2 για να δούμε τη συσχέτιση του δείγματος των 5,000 tweets στο 2-διάστατο επίπεδο.



Εικόνα 4.11: Διάγραμμα διασποράς των tweets για 20 θέματα συζήτησης που προκύπτουν με τη μοντελοποίηση LDA

Εδω πέρα οι περιοχές που δημιουργούνται δείχνουν να είναι πιο καλά **διαχωρισμένες** χωρίς να υπάρχουν πολλές επικαλύψεις μεταξύ tweets επιτρέποντας στα Θέματα συζήτησης να διακριθούν καλύτερα. Ακόμα παρατηρούμε ότι τα Tweets έχουν διαμοιραστεί πιο δίκαια ανά ομάδα και δεν παρατηρείται η κατάταξη πολλών μαζί σε μιά γενική μεγάλη συστάδα όπως πρίν.

Τέλος βλέποντας καλά το παραπάνω διάγραμμα διασποράς ότι δεν παρατηρούνται να είναι κοντά μόνο παρόμοια tweets αλλά και παρόμοια Θέματα (Topics). Για παράδειγμα τα Θέματα

3,10,11 βρίσκονται κοντά γεωμετρικά και ταυτόχρονα δείχνουν να παίρνουν και παρόμοια θέση. Πιο συγκεκριμένα συγκεντρώνουν κοντά θέσεις υπέρ της παραμονής στην Ευρώπη, ανησυχίες σχετικά με την έκβαση σε περίπτωση ΟΧΙ και τάση κατά της κυβέρνησης αντίστοιχα. Δηλαδή απόψεις που λογικά ταυτίζονται τείνουν να συγκεντρώνονται σε γειτονικά "νησάκια".

4.4.4 Ομαδοποίηση Χρηστών

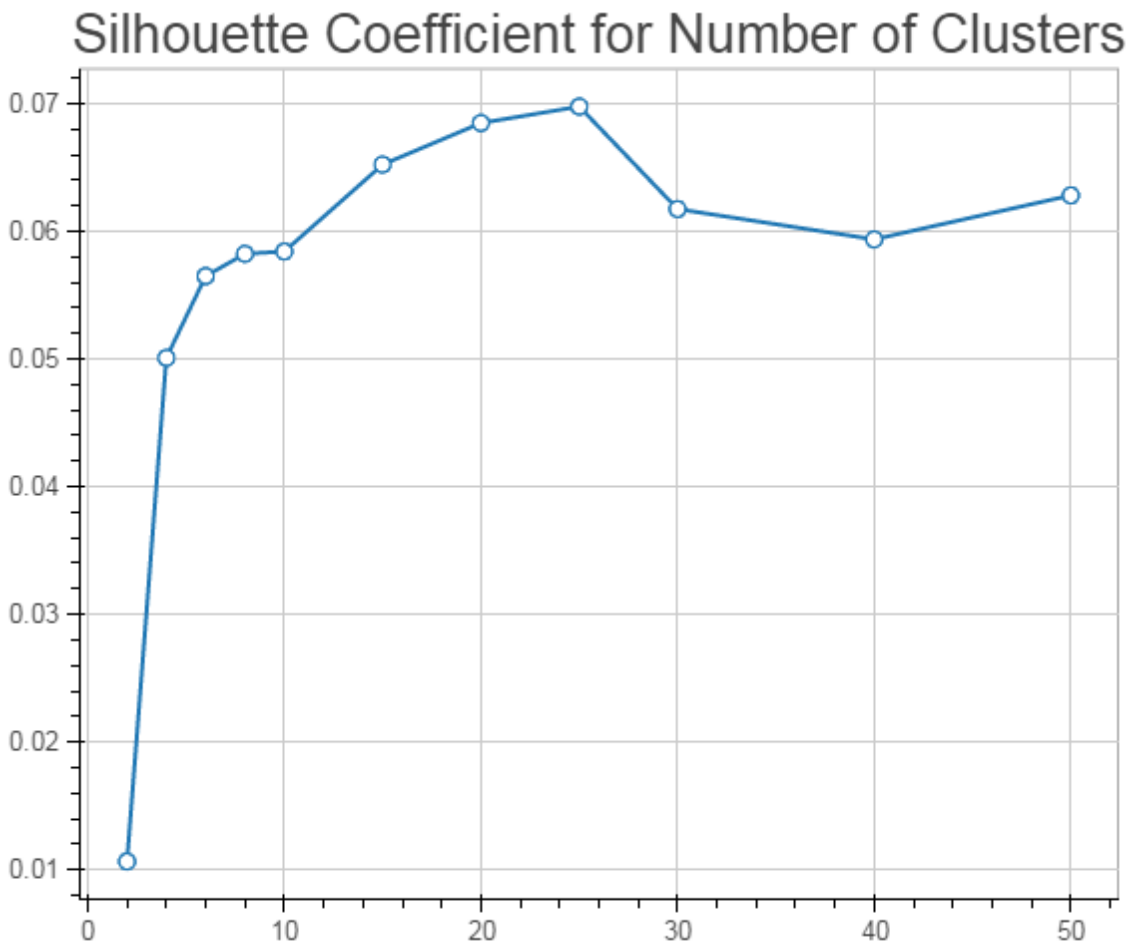
Εδώ τα επιμέρους αντικείμενα με βάση τα οποία προχωρήσαμε στην ομαδοποίηση είναι οι χρήστες. Και επειδή πάντα μιλάμε για κείμενα, στον κάθε χρήστη αντιστοιχεί ένα σύνολο από Tweets τα οποία αυτός έχει δημοσιεύσει, έχουν περάσει το προεπεξεργαστικό στάδιο και μαζί συνθέτουν τη συλλογή του (αντικείμενο).

4.4.4.1 *Συσταδοποίηση Κειμένου K-Means*

Και εδώ όπως και προηγουμένως κάναμε χρήση της συνιστώσας *Silhouette* για αριθμούς συστάδων: 2,4,6,8,10,15,20,25,30.

Όπως φαίνεται και στο παρακάτω **διάγραμμα** παίρνουμε και πάλι τιμές που είναι σχετικά κοντά στο 0 αφού οι λέξεις κλειδιά παραμένουν στις συστάδες αλλά με περίπου διπλάσιο score για τη συνιστώσα Silhouette σχέση με πρίν.

Αυτό συμβαίνει διότι ενώ το ένα tweet που είδαμε πριν ως αντικείμενο έχει περισσότερες πιθανότητες να είναι ουδέτερης κατεύθυνσης και να κατατάσσεται ανάμεσα σε 2 συστάδες, τα ενδεχομένως περισσότερα του ενός tweets που αντιστοιχούν στον κάθε χρήστη μπορούν να οδηγήσουν σε μεγαλύτερη προσέγγιση κάποιων κέντρων συστάδων.



Εικόνα 4.12: Silhouette Coefficient για μεταβλητό αριθμό συστάδων για συσταδοποίηση Χρηστών

Παρατηρούμε οτι για αριθμό συστάδων 20 και 25 πετυχαίνουμε τις ψηλότερες δυνατές τιμές με μικρή διαφορά μεταξύ τους. Επιλέγουμε πάλι τις 20 συστάδες για να αποφύγουμε όσο γίνεται τις επαναλήψεις συστάδων με το ίδιο νόημα.

Οι **20 συστάδες** που παίρνουμε με τα 8 χαρακτηριστικά (λέξεις) που επιλέξαμε να εμφανίσουμε είναι:

Συστάδα	**Λέξεις**
0	*αυριο οχι σημερα τραπεζες παρελθον τιμωρησεις επιλεγεις καναλια*
1	*oxi tsipras_eu syntagma grecia οχι oxi2015 nai tsipras*

2	*διαδικτυακη ερευνα greekcrisis 2015 απθ τρεμουν οχι μπειτε*
3	*primeministergr δραχμη vote referendum yes greek επρεπε people*
4	*oxi2015 oxi οχι grefenderum syntagma newsit γελαμε lemeoxi*
5	*game million unemployed living poverty people ellada tsipras_eu*
6	*παιδια atm ατμ οχι oxi λεφτα 20 μωρο*
7	*ατμ tsipras oxi ελλαδα κυριακη ευρωπη οχι σημερα*
8	*greececrisis grexit xreokopia oxi capitalcontrols οχι yeseurope tsipras*
9	*grexit vouli tsipras oxi capitalcontrols οχι atm ατμ*
10	*vouli οχι tsipras κκε ζωη kke τσιπρας αλεξη*
11	*menoumeevropi τελος menoume_malakes ευρωπη αναρτηση καλο ηρθε ζηταει*
12	*νομπελ οικονομιας liberation μνημονιακε παπια οχι freedybruna olympiada*
13	*greekreferendum grefenderum oxi oxi2015 greececrisis greek greekcrisis οχι*
14	*nai oxi ναι οχι menoumeevropi xreokopia grineuro naistinellada*
15	*yeseurope team_oxi oxi oxi2015 οχι xreokopia grexit nai*
16	*capitalcontrols syriza xreokopia syriza_xeftiles atm syriza_apateones tsipras οχι*
17	*αλεξη alexistsipras mnimonio3 ζορι μαζι βερολινο ερπη vouli*
18	*οχι ναι ευρωπη oxi κυριακη σημαινει κκε ελλαδα*
19	*υπογραψε πανευρωπαικο πλευρο χιλιαδες λιτοτητα ευρωπαιοι ελλαδας oxi*

Εικόνα 4.13: Οι 8 επικρατέστερες λέξεις των Συστάδων που προέκυψαν από τα κείμενα Χρηστών με K-Means

Ας δούμε εδω πως "σπάνε" οι απόψεις στις συστάδες και στην περίπτωση της ομαδοποίησης *Χρηστών*. Εδώ βλέπουμε κάποιες συστάδες που ξεχωρίζουν για τη θέση τους ωστόσο αρκετές περιλαμβάνουν ανάμεικτα μηνύματα.

Ας δούμε μερικές παρακάτω:

Συστάδα 1: Συνδυασμός Τσίπρα (*tsipras_eu,tsipras*) με το ΟΧΙ (*οχι,oxi*) και τη διαδήλωση στο Σύνταγμα(*syntagma*).

Συστάδα 4: Ξεκάθαρη τάση υπέρ του ΟΧΙ με αρκετές λέξεις κλειδιά μαζεμένες να το αποδεικνύουν (*oxi2015, oxi, οχι, team_oxi, lemeoxi*)

Συστάδα 8: Γενικότερες ανησυχίες για την κατάσταση με αναφορές στα capital controls (*capitalcontrols*) το ενδεχόμενο εξόδου της Ελλάδας από την Ευρώπη(*grexit*) και την κρίση (*greececrisis,xreokopia*).

Συστάδα 14: Δείχνει μια τάση υπέρ του ΝΑΙ μέσω αναφορών σ'αυτό (ναι,*nai)* καθώς και στην παραμονή της Ελλάδας στην Ευρώπη *(menoumeevropi,grineuro)* αλλά και αρνητικών καταστάσεων που είχαν συνδυαστεί με τα αποτελέσματα σε αντίθετη περίπτωση όπως: *xreokopia*.

Συστάδα 16: Τάση κατά του Σύριζα *(syriza_xeftiles, syriza_apateones) σε συνδυασμό με αρνητικές καταστάσεις όπως: capitalcontrols,atm,xreokopia.*

Ας δούμε πως έγινε η **κατάταξη** σε **δείγμα από** κείμενα των **10 Χρηστών** :

#	Χρήστης	Κείμενα	Συστάδα
1	**KsidisX**	*θελω βγαζει παγκαλος 60 ευρω*	7
2	**Imperator_Lex**	*κασιδιαρης παρελθει #oxi2015 #ΟΧΙ*	4

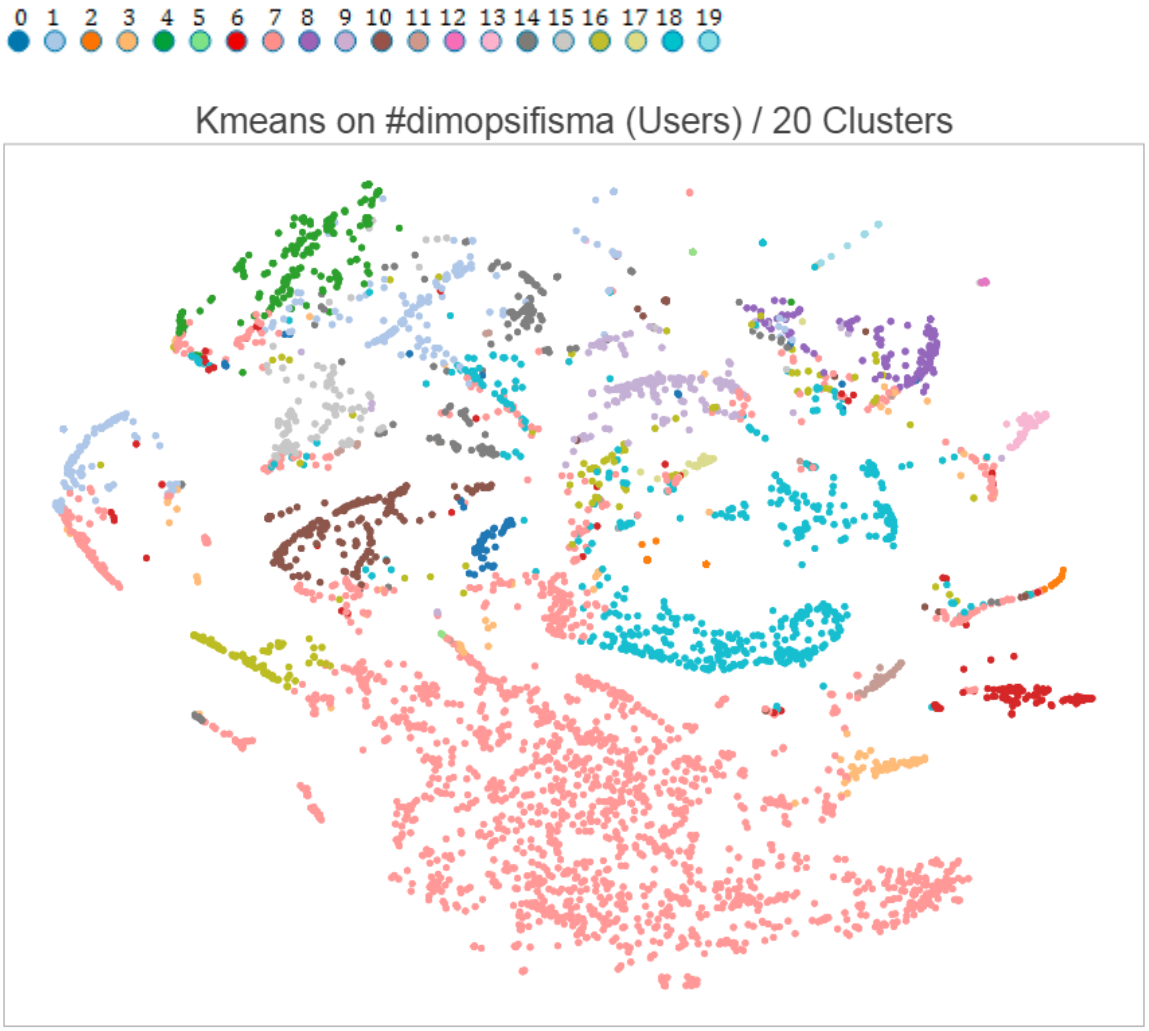
3	**PrimeministerGR**	*δευτερα ιουλιου δυναμη λαικης συνεχισουμε προσπαθειες #vouli μηνες ελλαδα υφισταται εκβιασμο αποδεχτει υφεσιακα μετρα λαος αποφασισει ελευθερα βεβαιος ελληνικος λαος στειλει μηνυμα δημοκρατιας αξιοπρεπειας ευρωπη #vouli σημερα ευρωπη ακουμπα ελλαδα προσβλεπει οχι αξιοπρεπειας ελληνικου λαου #vouli απολυσεων εργοδοτικου #vouli ισχυρο ισχυρη κυβερνηση διαπραγματευση επομενης μερας #ΟΧΙ εντολη δινετε ρηξης ευρωπη ενισχυσης διαπραγματευτικης δυναμης αποφαση ελληνικος λαος ρηξης ευρωπη #vouli δικιο υπερασπιζομαστε δημοκρατια ελλαδα ευρωπη μονη δυνατοτητα πετυχουμε περισσοτερα βγουμε φαυλο κυκλο κρισης ερωτηθει ελληνικος λαος δικαιωμα δικη επιλογη υπογραφη βαλουμε ταφοπλακα δημοκρατιας τοπο επισης ευρωπαικο κεκτημενο διαπραγματευσεων #vouli αποφαση ρηξης πρακτικες ευρωπη οχι διαπραγματευτικη δυνατοτητα χωρας #vouli φτασει λαος καλπη υπαρχουν σημεια δανειστες δημοψηφισματος εργαστουμε αμεση συμφωνια πουμε ψεμματα πολιτες μπορουμε τελειωσει περιπετεια #ΟΧΙ συλλαλητηριο υποβαθμιστηκε αντιδεοντολογικα ιδιωτικους τηλεοπτικους σταθμους #ΟΧΙ επιτεθουμε μισθωτους συνταξιουχους #vouli αρχες αξιες υπερασπιζομαστε δημοκρατια ισοτητα αλληλεγγυη αξιοπρεπεια κοινωνικα μεγαλο αποφαση κυβερνησης απορριψει τελεσιγραφο δημοκρατικη παραδοση δυνατοτητα εναν ολοκληρο λαο παρεμβει διαδικασια διαπραγματευσης κρισιμη επιμονη κυριως δντ προτασεις φορολογηση πλουτου παραλληλα σαφες εσοδα ελεγχο πλουτου #vouli οχι συμφωνησουμε παραδωσουμε πολιτικη αξιοπρεπεια #vouli καταλαβουν ελλαδα προκειται παραδοθει παιχνιδι τελειωσε #vouli φπα τουρισμο ξενοδοχεια εστιαση 5% 23% #vouli σημερινο νικητες αποτελει μεγαλη νικη ακομα δυσκολες συνθηκες δημοκρατια εκβιαζεται αποτελει κυριαρχη αξια επιλογη*	10
4	**neverlandfw**	*βαζει φιλη vs βοριδη αφηνει #enikos @NChatzinikolaou φωτα #GreeceCrisis ανοιξε φιλος κομμα προσφερει νεα προταση γιουνκερ κλεινει σημειο #enikos πλοιο ειχα πιει τοση νομιζα νιωθω σταυρο #enikos εικονα μμε δημοκρατιας κορεα πολλου χρεοκοπουν τραπεζες ντροπη χρεοκοπεις εθνος γιουνκερ εδωσα παντα μανταμ suis*	7
5	**mattsnucci**	*storia alla voto #Grecia #IoStoConLaGrecia @altraeuropa atm 13 #referendumgr*	6
6	**meropitzoufi**	*τοποθετηση ολομελεια #syriza #vouli*	10
7	**Nikos_Oikonomop**	*ελληνικε λαε παρε παραπανω φορους θελαμε οχι θεσμοι χρηματοδοτηση ορους κουγκι*	7

8	**Freddos**	*αιτηση esm δηλωσης νομου καναμε saturday σημερα βουλη αποτελεσμα παναθηναικος ολυμπιακος 42 40 εναλλακτικη ερωτηση περιπτωση τελικα μπορω ντρεπομαι οχι*	7
9	**tasoskritos**	*2015 διαδικτυακη ερευνα #GreekCrisis*	2
10	**redslicedtomato**	*παω ψηφισω νικη αδελφια #oxi2015 #team_OXI*	15

Εικόνα 4.14: Δείγμα Κατάταξης των Κειμένων 10 Χρηστών με το μοντέλο K-Means

Η **οπτικοποίηση** στο 2-διάστατο επίπεδο έγινε όπως προηγουμένως λαμβάνοντας δείγμα 5,000 Χρηστών με τη διαφορά ότι το κάθε σημείο εδώ αντιπροσωπεύει κάθε χρήστη (όχι απλά Tweet), δηλαδή το σύνολο κειμένων αυτού. Εδώ η διάσταση μειώθηκε από 5000 x 20 συστάδες σε 5000 x 2 μέσω του t-SNE.

Το διάγραμμα διασποράς στην ομαδοποίηση χρηστών για 20 συστάδες που προκύπτει λοιπόν με τη συσταδοποίηση K-Means προκύπτει όπως φαίνεται παρακάτω:



Εικόνα 4.15: Διάγραμμα διασποράς των χρηστών(συλλογή tweets του καθενός) για 20 συστάδες που προκύπτει με τη συσταδοποίηση K-Means

Απο τα παραπάνω βλέπουμε ότι επιτυγχάνεται κάποιος **διαχωρισμός** για τα *Κείμενα* (σύνολο *Tweets*) του κάθε χρήστη φέρνοντας αρκετές κοινές απόψεις μαζί, ωστόσο εδώ υπάρχουν και αρκετές συστάδες με συνδυασμό αντίθετων απόψεων. Ακόμα παρατηρούνται λιγότερες επικαλύψεις σχέση με τον *K-Means* στην ομαδοποίηση *Tweets* και αυτό είναι λογικό καθώς η πιθανή ύπαρξη πολλαπλών *Tweets ανά χρήστη* μπορεί να ορίσει καλύτερα την άποψη αυτού. Γενικά και δω όπως και σε κάθε περίπτωση παρατηρείται το χτίσιμο περιοχών γύρω από κάποιες δημοφιλείς λέξεις κλειδιά. Ακόμα παρατηρούμε για τον *K-Means* σε σχέση με τον *LDA* ότι αποτυγχάνει να φέρει "νησίδες" κοινής σημασίας κοντά.

4.4.4.2 *Μοντελοποίηση Θεμάτων LDA*

Τα **20 Θέματα** που προκύπτουν εδώ είναι, με τις 8 σχετικότερες λέξεις τους είναι:

Θέμα	**Λέξεις**
0	*greececrisis tsipras οχι capitalcontrols grexit eurogroup syntagma eurozone*
1	*οχι οπως τελικα υπαρχει βγει παντως μαζι παιδια*
2	*οχι live ναι ελλαδα tsipras δημοψηφισματος ευρω vouli*
3	*grefenderum samaras σαμαρα οχι σαμαρας greekreferendum πες mega*
4	*οχι τραπεζες χρονια κοσμο ελληνες λεφτα προπαγανδα ψηφισουν*
5	*2015 greekcrisis ερευνα διαδικτυακη antireport 06 απθ 07*
6	*skai_xeftiles oxi mega_xeftiles skai enikos ant1_xeftiles greekmediapropaganda mega*
7	*grexit xreokopia capitalcontrols capitalcontrol yeseurope οχι greececrisis default*
8	*oxi greek eu greece people europe democracy tsipras_eu*
9	*nai oxi yeseurope menoumeevropi ναι naistinellada grineuro ευρωπη*
10	*oxi euro europe yes gr vouli calm greferentum*
11	*τσιπρα τσιπρας ελλαδα γιουνκερ συμφωνια ευρωπη μερκελ προταση*
12	*syriza yeseurope tsipras syriza_xeftiles syriza_apateones topotami τραπεζες οχι*
13	*οχι κκε ερωτημα προταση ευρω γιουνκερ τελικα κυβερνηση*
14	*ατμ atm παω 60 capitalcontrols αυριο ευρω παιδια*
15	*οχι ευρωπαιοι ελλαδας λιτοτητα χιλιαδες πλευρο πανευρωπαικο υπογραψε*

16	*vouli ζωη τσιπρας tsipras αλεξη κκε enikos πουμε*
17	*ελληνες οχι atsipras λαος ευρωπη μεσω δημοκρατια εξωτερικου*
18	*οχι oxi oxi2015 αυριο οχι2015 συνταγμα ναι youtube*
19	*oxi oxi2015 team_oxi lemeoxi yeseurope nai syntagma grefenderum*

Εικόνα 4.16: Οι 8 επικρατέστερες λέξεις των Θεμάτων που προέκυψαν από τα κείμενα χρηστών με LDA

Η διάκριση των απόψεων στα επιμέρους Θέματα και σε αυτή την περίπτωση εκτέλεσης ομαδοποίησης από τον *LDA* είναι πιο ξεκάθαρη από αυτή που είδαμε στις Συστάδες του *K-Means* πρίν. Για παράδειγμα ας δούμε ορισμένα Θέματα:

Θέματα 0,7: Γενικότερη αναφορά στις ανησυχίες του κόσμου σχετικά με την κατάσταση και τις εξελίξεις με συγκέντρωση λέξεων όπως *grexit, xreokopia, capitalcontrols, greececrisis, eurogroup,eurozone.*

Θέμα 3: Θέμα που περιστρέφεται γύρω από τον αρχηγό της ΝΔ, Σαμαρά με πολλαπλές αναφορές στο όνομα του *(samaras,σαμαρα,σαμαρας).*

Θέμα 6: Θέμα συζήτησης που δείχνει την τάση κατά των καναλιών με πολλαπλές λέξεις κλειδιά να το επιβεβαιώνουν όπως: *greekmediapropaganda,skai_xeftiles, mega_xeftiles,ant1_xeftiles,skai,mega,enikos,καναλια* με το ενδιαφέρον στοιχείο εδώ να είναι ο συνδυασμός με το ΟΧΙ *(oxi).*

Θέμα 9: Θέμα όπου η άποψη υπέρ του ΝΑΙ σε συνδυασμό με παραμονή της Ελλάδας στην Ευρώπη φαίνεται ξεκάθαρα *(nai, menoumeevropi, naistinellada, yeseurope, ναι, grineuro, ευρωπη*).

Θέμα 11: Θέματα που περιλαμβάνουν αναφορές στον πρωθυπουργό Αλέξη Τσίπρα (*tsipras,τσιπρας,αλεξη*) με κάποιες αναφορές στην Ευρώπη και στην πρόταση των ξένων (*πρόταση,Ευρώπη,γιούνκερ,μέρκελ*).

Θέμα 12: Τάση κατά του Σύριζα *(syriza_xeftiles, syriza_apateones)* που δείχνει να συνδυάζεται με την επιθυμία παραμονής στην Ευρώπη *(yeseurope)*. Όπως είναι λογικό εδώ, η αναφορά στον Σύριζα και στον πρωθυπουργό Τσίπρα *(syriza,tsipras)* γίνεται για αρνητικό σκοπό.

Θέμα 16: Αναφέρεται στον πρωθυπουργό (*τσιπρασ,tsipras,αλεξη*) κυρίως αλλά και με πράγματα που μπορεί να συμβαίνουν στη βουλή (*vouli,ζωη,κκε*).

Θέματα 18,19: Απόψεις που συγκεντρώνουν τάση υπέρ του ΟΧΙ τόσο με λέξεις κλειδιά που το περιλαμβάνουν (*oxi,oxi2015,οχι,team_oxi,lemeoxi*) σε συνδυασμό με αναφορές στο Σύνταγμα (*syntagma,συνταγμα*) όπου και έγινε η διαδήλωση υπέρ του.

Ας δούμε πως έγινε η **κατάταξη** σε **δείγμα από** κείμενα των **10 Χρηστών** :

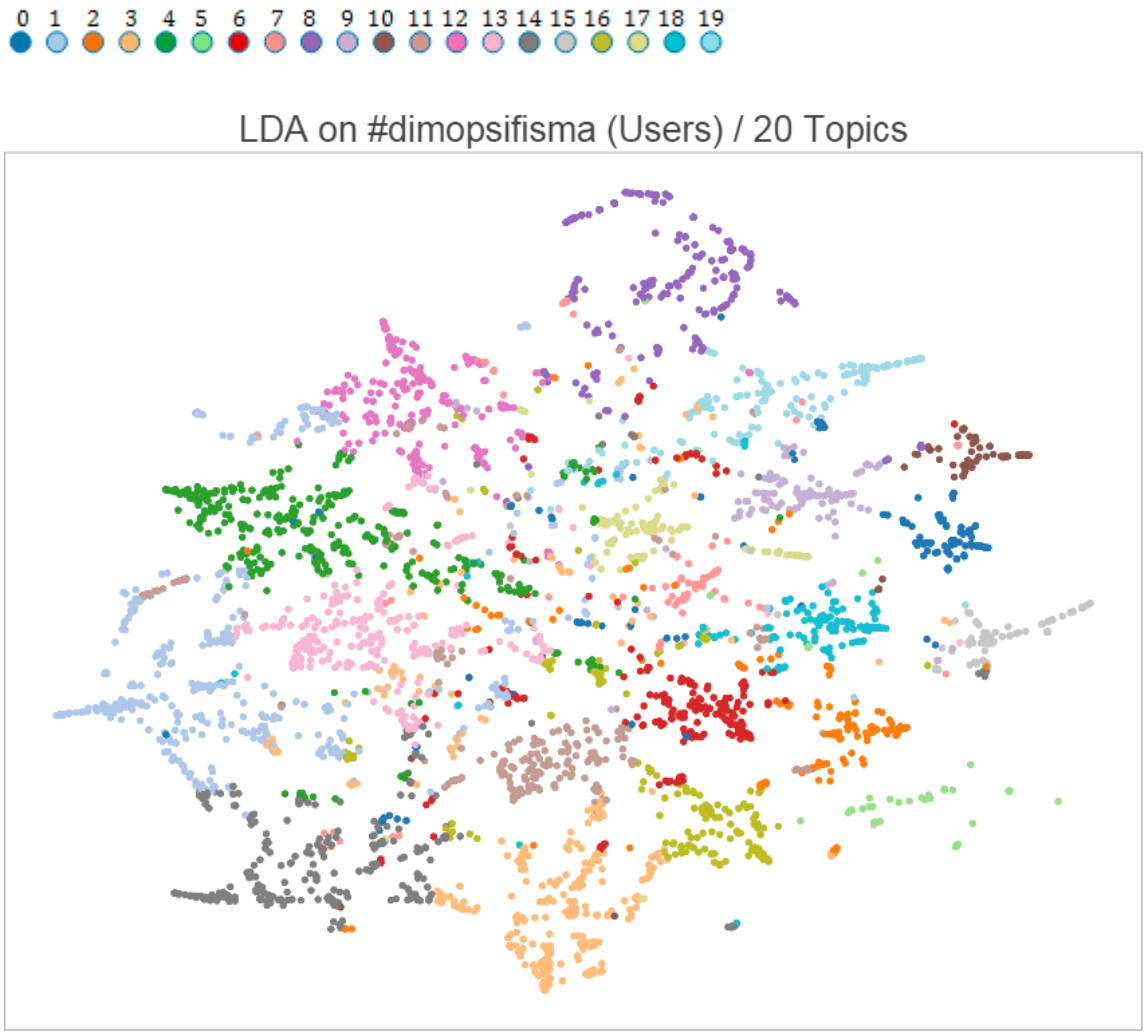
#	Χρήστης	Κείμενα	Θέμα
1	**KsidisX**	*θελω βγαζει παγκαλος 60 ευρω*	14
2	**Imperator_Lex**	*κασιδιαρης παρελθει #oxi2015 #ΟΧΙ*	19
3	**PrimeministerGR**	*δευτερα ιουλιου δυναμη λαικης συνεχισουμε προσπαθειες #vouli μηνες ελλαδα υφισταται εκβιασμο αποδεχτει υφεσιακα μετρα λαος αποφασισει ελευθερα βεβαιος ελληνικος λαος στειλει μηνυμα δημοκρατιας αξιοπρεπειας ευρωπη #vouli σημερα ευρωπη ακουμπα ελλαδα προσβλεπει οχι αξιοπρεπειας ελληνικου λαου #vouli απολυσεων εργοδοτικου #vouli ισχυρο ισχυρη κυβερνηση διαπραγματευση επομενης μερας #ΟΧΙ εντολη δινετε ρηξης ευρωπη ενισχυσης διαπραγματευτικης δυναμης αποφαση ελληνικος λαος ρηξης ευρωπη #vouli δικιο υπερασπιζομαστε δημοκρατια ελλαδα ευρωπη μονη δυνατοτητα πετυχουμε περισσοτερα βγουμε φαυλο κυκλο κρισης ερωτηθει ελληνικος λαος δικαιωμα δικη επιλογη υπογραφη βαλουμε ταφοπλακα δημοκρατιας τοπο επισης ευρωπαικο κεκτημενο διαπραγματευσεων #vouli αποφαση ρηξης πρακτικες ευρωπη οχι διαπραγματευτικη δυνατοτητα χωρας #vouli φτασει λαος καλπη υπαρχουν σημεια δανειστες δημοψηφισματος εργαστουμε αμεση συμφωνια πουμε ψεμματα πολιτες μπορουμε τελειωσει περιπετεια #ΟΧΙ συλλαλητηριο υποβαθμιστηκε αντιδεοντολογικα*	17

		ιδιωτικους τηλεοπτικους σταθμους #OXI επιτεθουμε μισθωτους συνταξιουχους #vouli αρχες αξιες υπερασπιζομαστε δημοκρατια ισοτητα αλληλεγγυη αξιοπρεπεια κοινωνικα μεγαλο αποφαση κυβερνησης απορριψει τελεσιγραφο δημοκρατικη παραδοση δυνατοτητα εναν ολοκληρο λαο παρεμβει διαδικασια διαπραγματευσης κρισιμη επιμονη κυριως δντ προτασεις φορολογηση πλουτου παραλληλα σαφες εσοδα ελεγχο πλουτου #vouli οχι συμφωνησουμε παραδωσουμε πολιτικη αξιοπρεπεια #vouli καταλαβουν ελλαδα προκειται παραδοθει παιχνιδι τελειωσε #vouli φπα τουρισμο ξενοδοχεια εστιαση 5% 23% #vouli σημερινο νικητες αποτελει μεγαλη νικη ακομα δυσκολες συνθηκες δημοκρατια εκβιαζεται αποτελει κυριαρχη αξια επιλογη	
4	**neverlandfw**	*βαζει φιλη vs βοριδη αφηνει #enikos @NChatzinikolaou φωτα #GreeceCrisis ανοιξε φιλος κομμα προσφερει νεα προταση γιουνκερ κλεινει σημειο #enikos πλοιο ειχα πιει τοση νομιζα νιωθω σταυρο #enikos εικονα μμε δημοκρατιας κορεα πολλου χρεοκοπουν τραπεζες ντροπη χρεοκοπεις εθνος γιουνκερ εδωσα παντα μανταμ suis*	6
5	**mattsnucci**	*storia alla voto #Grecia #IoStoConLaGrecia @altraeuropa atm 13 #referendumgr*	10
6	**meropitzoufi**	*τοποθετηση ολομελεια #syriza #vouli*	3
7	**Nikos_Oikonomop**	*ελληνικε λαε παρε παραπανω φορους θελαμε οχι θεσμοι χρηματοδοτηση ορους κουγκι*	13
8	**Freddos**	*αιτηση esm δηλωσης νομου καναμε saturday σημερα βουλη αποτελεσμα παναθηναικος ολυμπιακος 42 40 εναλλακτικη ερωτηση περιπτωση τελικα μπορω ντρεπομαι οχι*	13
9	**tasoskritos**	*2015 διαδικτυακη ερευνα #GreekCrisis*	5
10	**redslicedtomato**	*παω ψηφισω νικη αδελφια #oxi2015 #team_OXI*	15

Εικόνα 4.17: Δείγμα Κατάταξης των Κειμένων 10 Χρηστών με το μοντέλο LDA

Η **οπτικοποίηση** ακολουθεί ακριβώς τη διαδικασία που είδαμε στον K-Means για ομαδοποίηση Χρηστών.

Το διάγραμμα διασποράς για 20 Θέματα που προκύπτει λοιπόν με τη μοντελοποίηση LDA προκύπτει όπως φαίνεται παρακάτω:



Εικόνα 4.18: Διάγραμμα διασποράς των χρηστών (συλλογή tweets του καθενός) για 20 Θέματα Συζήτησης που προκύπτουν με τη μοντελοποίηση LDA

Όπως και στην προηγούμενη περίπτωση της ομαδοποίησης Tweets έτσι και εδώ πετυχαίνουμε καλύτερα αποτελέσματα με τον LDA από κάθε άποψη. Οι περιοχές των Θεμάτων στο 2-διάστατο δείγμα δείχνουν να είναι καλύτερα **διαχωρισμένες** με λιγοστές επικαλύψεις μεταξύ Χρηστών. Το δείγμα των Χρηστών διαμοιράζεται σε ομάδες που έχουν σχετικά παρόμοιο πλήθος στοιχείων.Τέλος απ'το διάγραμμα παρατηρούμε οτι παρόμοια Θέματα, των οποίων το περιεχόμενο φαίνεται παραπάνω, τείνουν να βρίσκονται κοντά γεωμετρικά με τον LDA. Για παράδειγμα τα Θέματα 11,16 στο διάγραμμα μας τα οποία ασχολούνται με τον πρωθυπουργό βρίσκονται σε γειτονικά "νησάκια", ενώ δίπλα τους εμφανίζεται και το Θέμα που εκφράζει την

τάση κατά των καναλιών(Θέμα 6). Αυτό είναι ένα σημαντικό πλεονέκτημα του LDA έναντι του K-Means διότι πέρα από τον επιτυχή διαχωρισμό απόψεων σε διαφορετικά Θέματα καταφέρνουμε να δούμε κατά μία έννοια πόσο και πως αυτά συσχετίζονται μεταξύ τους.

4.5 Σύγκριση Αποτελεσμάτων

Η σύγκριση των μοντέλων που χρησιμοποίησαμε αποτελεί ουσιαστικά ένα δείγμα σύγκρισης της ***Συσταδοποίησης*** *(K-Means)* με την ***Κατάταξη*** *(LDA)*. Τόσο ο *K-Means* όσο και ο *Latent Dirichlet Allocation(LDA)* είναι αλγόριθμοι *μη-επιβλεπόμενης Μηχανικής Μάθησης,* στους οποίους ο προγραμματιστής πρέπει να καθορίζει εκ των προτέρων τον αριθμό *Συστάδων (k)* και τον αριθμό *Θεμάτων (n)* αντίστοιχα.

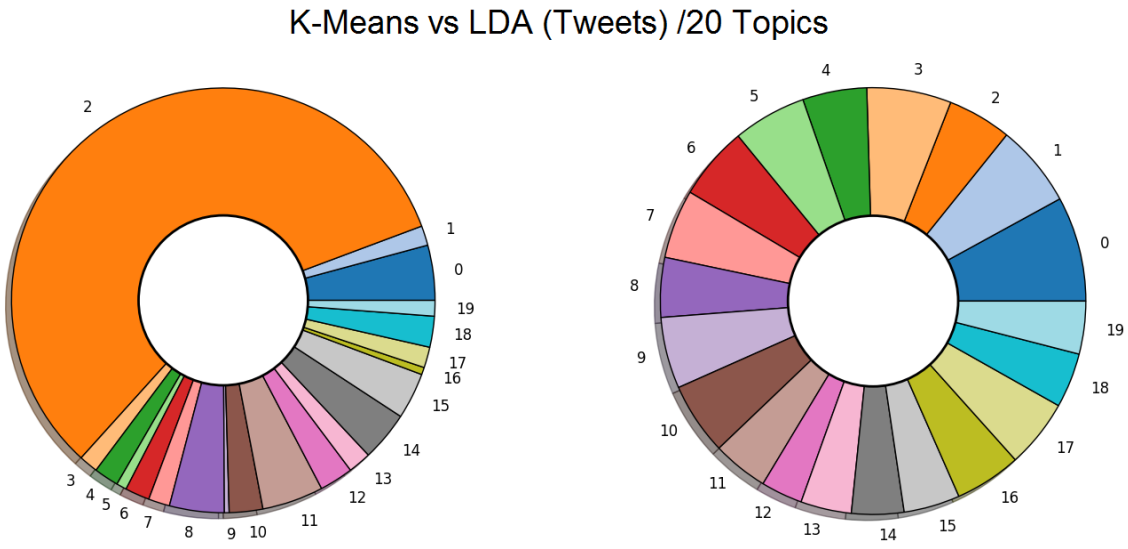
Κατά την εφαρμογή και των 2 με σκοπό την ανάθεση *k (αντίστοιχα n)* Θεμάτων σε ένα σύνολο *N Κειμένων*, η πιό μεγάλη διαφορά στα αποτελέσματα τους οφείλεται στο ότι ο *K-Means* χωρίζει τα *N* κείμενα σε *k* ξένες μεταξύ τους *Συστάδες(Θέματα)*, δηλαδή σε *Θέματα* χωρίς κοινά στοιχεία. από την άλλη, ο *LDA* αναθέτει το κάθε *κείμενο* σε μια *μίξη Θεμάτων*. Επομένως στον *LDA* κάθε *κείμενο* χαρακτηρίζεται από 1 ή περισσότερα *Θέματα* με τα αντίστοιχα βάρη. Επιπροσθέτως, ο *LDA* μας δίνει τα *Θέματα* ως πιθανοτικές κατανομές επί των *λέξεων*. Για παράδειγμα ένα *κείμενο* μπορεί να ανήκει *60%* στο *Θέμα Α, 30%* στο *Θέμα Β* και *10%* στο *Θέμα Γ*.

Για αυτό το λόγο ο LDA δίνει πιο ρεαλιστικά αποτελέσματα από τον K-Means κατά την ανάθεση των Θεμάτων.

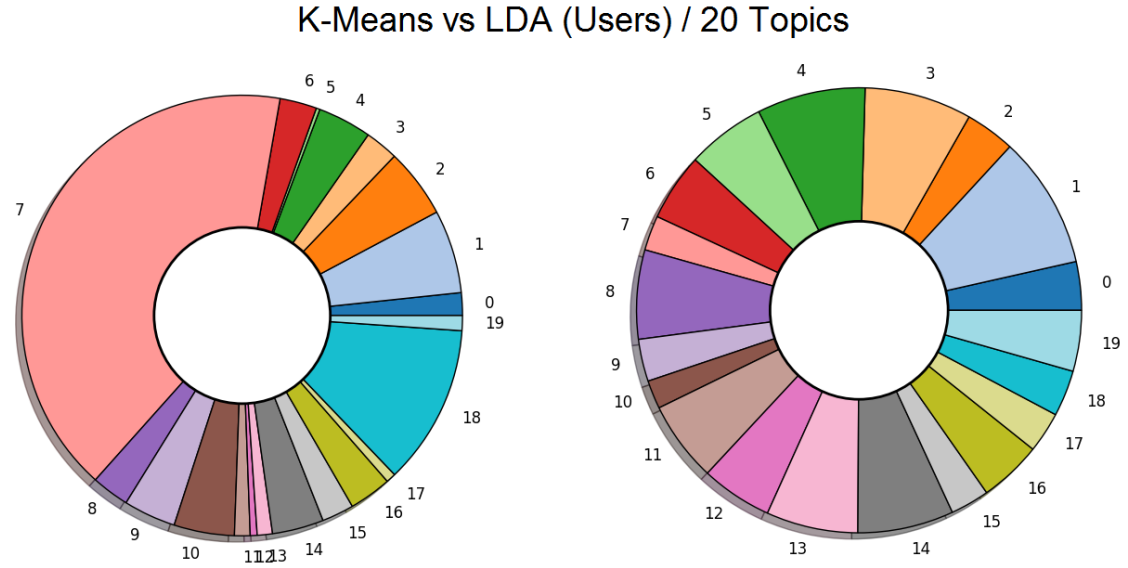
Πιο **συγκεκριμένα στη μελέτη μας** τόσο με τον *K-Means* όσο και με τον *LDA*, πετυχαίνουμε ***ικανοποιητικό εντοπισμό Θεμάτων***, με *λέξεις* που εκφράζουν κατά ένα μεγάλο ποσοστό είτε κάποια θέση είτε σωστά συγκεντρωμένες απόψεις. Βέβαια δεν έχει αποφευχθεί η ύπαρξη μερικών συστάδων που περιέχουν λέξεις που εκφράζουν ανάμεικτα νοήματα κυρίως με τη συσταδοποίηση K-Means. Τα *κείμενα* στο παράδειγμα μας, αφού προέρχονται από το *Twitter*, που είναι ένα γενικής φύσεως κοινωνικό δίκτυο, μπορούν να μιλάνε για οποιοδήποτε συνδυασμό *Θεμάτων Συζήτησης*. Για αυτό ο *LDA* ως μια *soft clustering* τεχνική, σε σχέση με τον *K-Means* πού είναι *hard clustering* τεχνική πετυχαίνει πιό ικανοποιητικό εντοπισμό. Βέβαια σε κάθε περίπτωση υπάρχουν μερικά Θέματα με λέξεις που δείχνουν να έχουν ταιριάξει

λάθος. Αυτό είναι λογικό καθώς τα κείμενα μπορεί να προέρχονται από θεωρητικά ουδέτερες πηγές (πχ. δημοσιογράφους) στα κείμενα ή να περιλαμβάνουν παράγοντες όπως είναι η ειρωνεία.

Επίσης παρατηρούμε διαφορά στον ***αριθμό των στοιχείων ανά Θέμα***. Οι συστάδες στον K-Means δεν είναι ίδιας τάξης πλήθους στοιχείων με τις συστάδες του LDA όπως φαίνεται και στα παρακάτω Σχήματα για Tweets και χρήστες αντίστοιχα:



Εικόνα 4.19: Διάγραμμα Πίτας διαμοιρασμού των Tweets ανά Θέμα(Συστάδα) κατά την εκτέλεση των μοντέλων μας



Εικόνα 4.20: Διάγραμμα Πίτας διαμοιρασμού των Χρηστών ανά Θέμα (Συστάδα) κατά την εκτέλεση των μοντέλων μας

Αυτό το φαινόμενο συμβαίνει διότι ο K-Means στην προσπάθεια του να ελαχιστοποιήσει το άθροισμα των τετραγώνων εντός των επιμέρους συστάδων, καταλήγει να δίνει περισσότερο βάρος στις μεγαλύτερες συστάδες. Πρακτικά, αυτό σημαίνει ότι ο K-Means αφήνει μια μικρή συστάδα να καταλήξει μακριά από οποιοδήποτε κέντρο, ενώ χρησιμοποιεί αυτά τα κέντρα για το διαχωρισμό μιας πολύ μεγαλύτερης Συστάδας. Ο LDA αντίθετα πετυχαίνει διαχωρισμό των Κειμένων σε συστάδες παραπλήσιου μήκους και αυτό οφειλεται στη καλύτερη μοντελοποίηση του πρόβληματος που προσφέρει η λογική ότι ένα αντικείμενο μπορεί να ανήκει σε πολλά διαφορετικά Θέματα.

Ακόμα όσον αφορά την ***κατάταξη των επιμέρους Κειμένων*** στα *Θέματα* που δημιουργήθηκαν ο *LDA* υπερέχει σε σχέση με τον *K-Means*. Αυτή η διαφορά μεταξύ τους παρατηρείται λόγω των *Θεμάτων* που δημιουργήθηκαν πριν. Ανάμεικτα *Θέματα* που έχουν δημιουργηθεί κυρίως στον *K-Means* μπορεί να "τραβάνε" κείμενα που περιέχουν ένα από τα μπλεγμένα σε αυτά νοήματα, μη αποδίδοντας έτσι πληροφορία.

Επιπλέον στην περίπτωση ομαδοποίησης *ανά χρήστη* πετυχαίνουμε καλύτερα αποτελέσματα λόγω του ότι το κάθε κείμενο ενδέχεται να έχει περισσότερες *λέξεις* για να ταιριάξει με τις αντίστοιχες των Θεμάτων.

Άρα συνδυαστικά θα λέγαμε ότι πετυχαίνουμε ***καλύτερη κατάταξη*** με τον αλγόριθμο *LDA* με ομαδοποιημένα *ανά χρήστη κείμενα*.

Τέλος, είδαμε ότι σε κάθε περίπτωση μοντέλου, ο ***διαχωρισμός των Κειμένων ανά χρήστη*** επιφέρει λιγότερες επικαλύψεις σχέση με τον διαχωρισμό *ανά Tweet* και αυτό είναι λογικό καθώς η πιθανή ύπαρξη πολλαπλών *Tweets ανά χρήστη* μπορεί να ορίσει καλύτερα την άποψη αυτού. Ουσιαστικά επιτυγχάνεται κατά μία έννοια αντιμετώπιση του μειονεκτήματος του περιορισμού χαρακτήρων (μέχρι 140) που υπάρχει *ανά Tweet*. Προϋπόθεση βέβαια η ύπαρξη πολλαπλών *Tweets ανά χρήστη*.

5

Υλοποίηση

5.1 Περιγραφή Βασικών Συναρτήσεων

Παρακάτω θα περιγράψουμε τις βασικές κλάσεις/συναρτήσεις που υλοποιούν αυτά που παρέχει το σύστημα, όπως τα αναφέραμε παραπάνω.

5.1.1 Συναρτήσεις Συλλογής Δεδομένων από Twitter

Σε κάθε περίπτωση θα **επικοινωνούμε με τη βάση** οπότε πρίν την κύρια ροή πρέπει να ανοίξουμε τη σύνδεση μας με αυτή. Για την δημιουργία *Οντοτήτων* στον *Γράφο* κάνουμε χρήση της ***merge_one()*** που τρέχει επί του Γράφου *(Graph)* με ορίσματα: τον *τύπο οντότητας* (πχ. Tweet, User) και το αναγνωριστικό χαρακτηριστικό του *(ID).* Για την ενημέρωση των χαρακτηριστικών *Κόμβου* χρησιμοποιούμε την ***properties.update()*** επι του *Κόμβου* με ορίσματα τα *χαρακτηριστικά* με τις αντίστοιχες τιμές τους. Τέλος για την δημιουργία *Σχέσεων* μεταξύ *Κόμβων* της ***create_unique()*** επι του *Γράφου* που δέχεται ως όρισμα τη *Σχέση* που προηγουμένως πρέπει να έχουμε ορίσει μέσω της ***Relationship()*** η οποία με τη σειρά της έχει ορίσματα τους 2 *Κόμβους* που θα συνδέσουμε με τον αντίστοιχο *τύπο σύνδεσης (πχ.Posts, Follows).* Όλες οι παραπάνω αποτελούν συνάρτησεις της βιβλιοθήκης *Py2neo* για εκτέλεση εντολών της βάσης *Neo4j* εντός του *Python* προγράμματος μας.

Για τη σύνδεση στα *APIs* του *Twitter* απαιτείται η ταυτοποίηση μέσω των *κλειδιών* που αντιστοιχούν σε κάθε λογαριασμό και επιτρέπουν την ανάκτηση και αποστολή δεδομένων προς αυτό.

Διαδικασία χειρισμού Tweets

Για **κάθε αντικείμενο Tweet** που συλλέγουμε αρχικά ελέγχουμε ότι αυτό δεν υπάρχει ήδη στη βάση με βάση το μοναδικό ID του και στη συνέχεια δημιουργούμε οντότητα τύπου Tweet με τα εκάστοτε χαρακτηριστικά του(*Κείμενο, Ημερομηνία, αριθμός retweets, αριθμός favourites*). Εκτός αυτού οφείλουμε να δημιουργήσουμε στη βάση και τις υπόλοιπες οντότητες υποχρεωτικές/προαιρετικές που μπορεί να το συνοδεύουν. Πιο αναλυτικά δημιουργούμε την οντότητα του *User*(*Χρήστη*) που το δημοσίευσε, αν δεν υπάρχει ήδη, με τα χαρακτηριστικά της (*Όνομα, Περιγραφή, Εικόνα, Ακόλουθοι κλπ.*) και την συνδέουμε με το *Tweet* μέσω της σχέσης *POST*.

Τώρα προαιρετικά για το Tweet που φέραμε ανάλογα την περίπτωση, δημιουργούμε οντότητες *Hashtag*, *Url*(συνδέσμου) τις οποίες παίρνουμε εφαρμόζοντας τις συναρτήσεις **entities.get('hashtags')** και **entities.get('urls')** της βιβλιοθήκης *TweePy* πάνω στο αντικείμενο Tweet αντίστοιχα. Δημιουργούμε τις παραπάνω *οντότητες* στη βάση μας με τα αντίστοιχα χαρακτηριστικά τους και τις συνδέουμε με το *Tweet* μέσω των *Σχέσεων TAGS* για το *Hashtag* και *CONTAINS* για τον *σύνδεσμο* αντίστοιχα.

Ακόμα σε περίπτωση που το συγκεκριμένο *Tweet επισημαίνει (Mentions)* κάποιους *Users (Χρήστες)* τους λαμβάνουμε μέσω της *εφαμοργής της* ***entities.get('user_mentions')*** *επι του Tweet* και τους συνδέουμε με αυτό μέσω της σχέσης *MENTIONS*.

Τέλος ενδέχεται το συγκεκριμένο *Tweet* να αποτελεί *Retweet (Αναδημοσίευση)* ενός άλλου ή *Reply (Απάντηση)* σε κάποιο άλλο. Σε περίπτωση που είναι *Retweet* δημιουργούμε τον κόμβο του αυθεντικού *Tweet* μέσω της οντότητας *retweeted_status* που περιλαμβάνεται στο *Tweet* που φέραμε με τα χαρακτηριστικά της και τα συνδέουμε μεταξύ τους μέσω της *Σχέσης `RETWEET OF`*. Ακόμα δημιουργούμε την *οντότητα* για τον *User* που δημοσίευσε το αρχικό *Tweet μέσω της συνάρτησης του REST API:* ***get_user()*** με όρισμα το όνομα *χρήστη* του. Αντίστοιχα στην περίπτωση που αποτελεί *Reply* άλλου λαμβάνουμε το αρχικό *Tweet* μέσω της συνάρτησης του REST API: ***get_status()*** με όρισμα το ID του αυθεντικού το οποίο έχουμε λάβει από το όρισμα *in_reply_to_status_id* του *Tweet* που φέραμε, το δημιουργούμε ως *οντότητα* στη βάση και το συνδέουμε τοTweet μας με αυτό μέσω της *σχέσης `REPLY TO`*. Οι συναρτήσεις *get_status(), get_user()* είναι επίσης συναρτήσεις του *TweePy* και επικοινωνούν με τη βάση του *Twitter*.

5.1.1.1 *get_tweets.py*

Το module αυτό χρησιμοποιεί το *Search* του *Twitter REST API* με το οποίο επιστρέφει ένα σύνολο από *Tweets* τα οποία πληρούν κάποια κριτήρια όπως συμπερίληψη κάποιας λέξης κλειδί (για παράδειγμα κάποιο *Hashtag*) σε αυτά ή συγκεκριμένη ημερομηνία δημοσίευσης. Στη συνέχεια ακολουθεί η διαδικασία που περιγράψαμε παραπάνω για τον χειρισμό του κάθε Tweet που φέρνουμε.

5.1.1.2 *get_live_tweets.py*

Για την συλλογή δεδομένων (*Tweets*) πραγματικού χρόνου από το *Twitter το module αυτό* χρησιμοποεί το *Filter Stream* του *Streaming API* του *Twitter* με κριτήριο αναζήτησης το *Hashtag* για το οποίο γίνεται ένα *Tweet* (πχ. *#Dimopsifisma, #Davos*). Στη συνέχεια ακολουθεί η διαδικασία που περιγράψαμε παραπάνω για τον χειρισμό του κάθε Tweet που φέρνουμε.

5.1.1.3 *get_user_timeline.py*

Αυτό το module αυτή έχει σκοπό την περαιτέρω εξερεύνηση των Users που υπάρχουν ήδη στη βάση μας λόγω κάποιου *Tweet* τους. Φέρνει λοιπόν τα Ονόματα τους από τη βάση εφόσον δεν έχουν εξερευνηθεί ήδη μέσω κατάλληλου *Cypher* ερωτήματος. Για κάθε όνομα που επιστρέφεται εκτελούμε τη συνάρτηση ***user_timeline()*** *του Twitter REST API* με όρισμα: το *Όνομα χρήστη* και την *ημερομηνία* που μας ενδιαφέρει. Η συνάρτηση αυτή επιστρέφει ένα σύνολο από *Tweets,* το καθένα από τα οποία θα ακολουθήσει τη διαδικασία που περιγράψαμε παραπάνω.

5.1.1.4 *get_user_network.py*

Το module αυτό έχει σκοπό την ανάκτηση του δικτύου *Followers (Ακόλουθοι)* και *Friends (Άτομα που αυτός ακολουθεί)* για τους *Users* αφού φέρει τα ονόματα τους από τη βάση και εφόσον δεν έχουν ήδη εξερευνηθεί μέσω κατάλληλου *Cypher* ερωτήματος. Τα δίκτυα Ακολούθων και Φίλων φέρνουμε μέσω των συνάρτησεων του Twitter API: **followers_ids()** και **friends_ids()** που δέχονται ως όρισμα του κάθε όνομα χρήστη. Στη συνέχεια, και για κάθε περίπτωση με τη συνάρτηση **lookup_users()** φέρνουμε όλα τα χαρακτηριστικά του κάθε *χρήστη,* δημιουργούμε την *Οντότητα* τη συνδέουμε με τον αρχικό *χρήστη*.

5.1.2 Συναρτήσεις Προεπεξεργασίας Δεδομένων

5.1.2.1 *tweet_filter.py*

Δέχεται ως είσοδο την εκάστοτε λίστα *Tweets,* η οποία προέκυψε από την εξόρυξη με βάση κάποιο συγκεκριμένο *Hashtag*, πάνω στην οποία θα εφαρμοστεί η ανάλυση καθώς και μία Λίστα Αποκλεισμού λέξεων σχετικών με το συγκεκριμένο *Hashtag.* Με βάση την παραπάνω είσοδο, εντός αυτής της συνάρτησης συντονίζεται η κλήση όλων των επιμέρους συναρτήσεων που φαίνονται παρακάτω και αφορούν την προεπεξεργασία του κάθε Tweet.

5.1.2.2 *remove_stopwords.py*

Χρησιμοποιείται για την αφαίρεση προκαθορισμένων και επιλεγμένων λέξεων και γραμμάτων (*Stopwords*) από τα *Tweets*. Τρέχει για κάθε ένα Tweet, το οποία και δέχεται ως είσοδο μαζί με τη *λίστα αποκλεισμού* που περιλαμβάνει τις *Stopwords*. Έτσι για κάθε Tweet ανατρέχουμε στα csv αρχεία που έχουμε αποθηκευμένες τις λίστες αποκλεισμού και όποια *λέξη* του *Tweet* ανήκει σε αυτές την αφαιρούμε.

5.1.2.3 *strip_symbols.py*

Παρομοίως με την παραπάνω μόνο που εδώ αφαιρούμε συγκεκριμένα σύμβολα όπως συνδέσμους(http:/…), τα σύμβολα για *Retweet*(RT), *Mention*(@), *Hashtags (#) κλπ.* αφού η πληροφορία τους περιέχεται στις *οντότητες* που δημιουργήσαμε και συνοδεύουν το κάθε *Tweet.*

5.1.2.4 *strip_accents.py*

Αυτή η συνάρτηση δέχεται μια συμβολοσειρά που αντιπροσωπεύει μια *λέξη* του *Tweet* και την μετατρέπει στην αντίστοιχη *χωρις Τονισμό*. Για να το πετύχει αυτό προγραμματιστικά κάνει κανονικοποίηση των επιμέρους χαρακτήρων των λέξεων μέσω της κωδικοποίησης τους (*ASCII*).

5.1.2.5 *cut_low_frequency.py*

Η συνάρτηση αυτή δέχεται στην είσοδο ως λίστα το *Λεξιλόγιο* μας και το επιστρέφει έχοντας αφαιρέσει τις *λέξεις* οι οποίες εμφάνιζαν στο σύνολο του *συχνότητα* εμφάνισης μικρότερη από το καθορισμένο *κατώφλι*, η τιμή του οποίου ρυθμίζεται παραμετρικά. Αυτό γίνεται

προγραμματιστικά με χρήση μια λίστας συχνοτήτων για τις λέξεις που σε συνδυασμό με το *κατώφλι* που ορίζουμε, οδηγεί ή όχι στην αφαίρεση της εκάστοτε *λέξης*.

5.1.2.6 *uniquify_string.py*

Η συνάρτηση δέχεται ως είσοδο σε μορφή συμβολοσειράς τα περιεχόμενα του *Tweet* που έχουν παραμείνει από τις υπόλοιπες συναρτήσεις προεπεξεργασίας και τα επιστρέφει σε παρόμοια μορφή χωρίς την ύπαρξη πολλαπλών λέξεων. Για το λόγο αυτό απαιτείται η χρήση κάποιας λίστας συχνοτήτων (για κάθε *Tweet*) με βάση την οποία θα αφαιρούνται οι λέξεις που εμφανίζονται πάνω από μία φορά στο *Tweet*.

5.1.3 Συναρτήσεις Ανάλυσης και Οπτικοποίησης Δεδομένων

Οι παρακάτω συναρτήσεις εφαρμόζονται πάνω σε σύνολα δεδομένων (Tweets), τα οποία φέρνουμε από τη βάση ανά περίπτωση με κατάλληλο ερώτημα στη Neo4j βάση μας μέσω του Py2neo. Τα δεδομένα αφού περάσουν και από το προεπεξεργαστικό στάδιο που περιγράψαμε παραπάνω τροφοδοτούν τους αλγόριθμους επεξεργασίας.

5.1.3.1 *kmeans_tweets.py*

Το module αυτό περιλαμβάνει τις συναρτήσεις που παράγουν το διάγραμμα διασποράς που αντιστοιχεί στην εφαρμογή του αλγόριθμου συσταδοποίησης Κειμένου *K-Means* πάνω στο σύνολο των *Tweets*.

Αρχικά μετατρέπουμε τα δεδομένα μας σε *tf-idf (βιβλιοθήκη Scikit-Learn)* διανύσματα ορίζοντας στις παραμέτρους στην εντολή: **vectorizer=*TfidfVectorizer*()** ώστε το κάθε κείμενο να εκφραστεί ως ένα 5000-διάστατο διάνυσμα, οι δείκτες του οποίου αντιστοιχούν στους 5 χιλιάδες πιο συχνά εμφανιζόμενους όρους στη συλλογή κειμένων μας μετά την εκπαίδευση**:** ***vectorizer.fit_transform(...)***. Η παραπάνω αναπαράσταση αποτελεί και την είσοδο του K-Means. Πιο συγκεκριμένα χρησιμοποιήσαμε τον *Mini Batch K-Means* της βιβλιοθήκης *Scikit-Learn (*μηχανική μάθηση σε Python) με τον αριθμό των συστάδων να δίνεται παραμετρικά. Το μοντέλο ορίζεται ως: ***kmeans_model = MiniBatchKMeans(...)*** με τις παραμέτρους που

αναφέραμε παραπάνω και τα διανύσματα προσαρμόζονται σε αυτό με τη συνάρτηση: ***kmeans_model.fit(...)*** που τα δέχεται ως είσοδο. Όταν η θέση των κεντροειδών δεν βελτιώνεται άλλο σταματάει η εκπαίδευση του μοντέλου και λαμβάνουμε τον πίνακα με διαστάσεις *k x Αριθμός_Κειμένων* που μας παρέχει την απόσταση του κάθε Κειμένου προς κάθε κέντρο απ'τον οποίο βρίσκουμε και σε ποιο κέντρο αυτά ανήκουν.Για κάθε μια από τις *k Συστάδες* επιλέγουμε να εμφανισουμε τις 8 κορυφαίες λέξεις.

Προκειμένου να αναπαραστήσουμε τα αποτελέσματα στο 2-διάστατο επίπεδο γίνεται χρήση του *αλγόριθμου Μείωσης Διαστάσεων t-SNE* ο οποίος δέχεται τις αποστάσεις που προέκυψαν από τον *K-Means* για τις *k* συστάδες και τις επιστρέφει σε 2-διάστατη μορφή (*2 x Αριθμός_Κειμένων*). Το μοντέλο ορίζεται μέσω της ***tsne_model = TSNE(...)*** με βασική παράμετρο τον αριθμό τον διαστάσεων και οι *αποστάσεις* του *K-Means* προσαρμόζονται μέσω της εντολής: ***tsne_kmeans = tsne_model.fit_transform(...)*** όντας παράμετροι για την προσαρμογή.

5.1.3.2 *kmeans_users.py*

Το module αυτό περιλαμβάνει τις συναρτήσεις που παράγουν το διάγραμμα διασποράς που αντιστοιχεί στην εφαρμογή του αλγόριθμου συσταδοποίησης Κειμένου *K-Means* πάνω σε ένα σύνολο από Users, όπου τον καθένα αντιπροσωπεύει ένα κείμενο με όλα του τα *Tweets*.

Η διαδικασία ανάλυσης είναι ακριβώς η ίδια με του *kmeans_tweets.py* που είδαμε παραπάνω μόνο που εδώ τα επιμέρους *κείμενα* κάθε *χρήστη* αποτελούν τα αντικείμενα που περνάνε απ'όλα αυτά τα στάδια.

5.1.3.3 *lda_tweets.py*

Αυτό το module περιλαμβάνει τις συναρτήσεις που παράγουν το διάγραμμα διασποράς που αντιστοιχεί στην εφαρμογή του αλγόριθμου μοντελοποίησης Θεμάτων *LDA* πάνω σε ένα σύνολο από *Tweets*.

Αρχικά ορίζουμε με την εντολή: ***cvectorizer=CountVectorizer(...)*** με παράμετρο τους 5 χιλιάδες πιο συχνά εμφανιζόμενους όρους στη συλλογή κειμένων μας, το μοντέλο με το οποίο θα διαμορφώσουμε ως διανύσματα τα *κείμενα* μας. Η μετατροπή γίνεται μέσω της: ***cvectorizer.fit_transform(...)***. Η λειτουργία αυτή προσφέρεται επίσης από στο πακέτο της βιβλιοθήκης *για μηχανική μάθηση* σε *Python, Scikit-Learn.*

Ως αποτέλεσμα παίρνουμε για κάθε κείμενο ένα 5000-διάστατο διάνυσμα σε μορφή πίνακα με διαστάσεις: *Αριθμός_Κειμένων x 5,000. Αυτός ο πίνακας αποτελεί την είσοδο του* αλγόριθμου

*LDA (*της βιβλιοθήκης *lda* της *Python*). Το μοντέλο του *LDA* ορίζεται: ***lda_model=lda.LDA(...)*** με βασικές παραμέτρους τον αριθμό *n* των *Θεμάτων* και τον *αριθμό επαναλήψεων* για τερματισμό εκπαίδευσης του μοντέλου. Η προσαρμογή των διανυσμάτων στον LDA γίνεται με τη συνάρτηση **lda_model.fit_transform(...)** που τα δέχεται ως είσοδο και επιστρέφει *ένα* πίνακα με διαστάσεις *Αριθμός_Κειμένων x Αριθμός_Θεμάτων.* Αυτός μας δίνει την κατανομή των *Θεμάτων* στα *κείμενα* και παράλληλα την κατάταξη κάθε Κειμένου σε κάποιο Θέμα (σε αυτό που εμφανίζει μεγαλύτερο ποσοστό). Για κάθε ένα από τα *n Θέματα* επιλέγουμε να εμφανιστούν οι 8 κορυφαίες (πιο σχετικές) λέξεις.

Στη συνέχεια θα μειώσουμε τις διαστάσεις όπως ακριβώς πράξαμε και με τον *K-Means* μέσω του t-SNE ο οποίος δέχεται τον παραπάνω πίνακα και επιστρέφει τον αντίστοιχο με διαστάσεις *2 x Αριθμός_Κειμένων*.

5.1.3.4 *lda_users.py*

Το module αυτό περιλαμβάνει τις συναρτήσεις που παράγουν το διάγραμμα διασποράς που αντιστοιχεί στην εφαρμογή του αλγόριθμου μοντελοποίησης Θεμάτων *LDA* πάνω σε ένα σύνολο από Users, όπου τον καθένα αντιπροσωπεύει ένα κείμενο με όλα του τα *Tweets*. Η διαδικασία ανάλυσης είναι ακριβώς η ίδια με του *lda_tweets.py* που είδαμε παραπάνω μόνο που εδώ τα αντικείμενα που περνάνε από τα επιμέρους στάδια είναι τα *ανά χρήστη κείμενα*.

5.1.3.5 *plots.py*

Το module αυτό περιλαμβάνει τις συναρτήσεις που απαιτούνται για την οπτικοποίηση των δεδομένων μας.

Για τη δημιουργία του *διάγραμματος διασποράς* μέσω της συνάρτησης **bokeh.plotting.*figure.scatter(...)*** της βιβλιοθήκης *bokehJS* της *Python*, δίνουμε ως είσοδο τις συντεταγμένες x,y που προέκυψαν από τους αλγόριθμους *K-Means ,LDA* των κλάσεων που αναφέραμε πρίν μετά τη μείωση διαστάσεων *t-SNE*. Τα σημεία, πού αντιπροσωπεύουν είτε τα επιμέρους *Tweets* είτε τα *κείμενα* των επιμέρους *χρηστών,* απεικονίζονται με κατάλληλο χρωματισμό ανάλογο της συστάδας στην οποία ανήκουν.

Ακόμα περιλαμβάνονται οι συναρτήσεις ***bar_chart()*** (βιβλιοθήκη *bokehJS*) και ***pie_chart()*** (βιβλιοθήκη *matplotlib*) οι οποίες δεχόμενες τον αριθμό των στοιχείων ανά *Θέμα/Συστάδα,* απεικονίζουν σε κάθε περίπτωση σε *ραβδόγραμμα* και *διάγραμμα ''πίτας''* αντίστοιχα την κατανομή των στοιχείων ανά *Συστάδα*.

Τέλος στην κλάση περιλαμβάνεται η ***silhouette_coefficient(),*** μέσω της οποίας απεικονίζεται σχηματικά η *συνιστώσα Silhouette* που προκύπτει από τον *K-Means*, σε συνάρτηση με τον αριθμό των *Θεμάτων* για τα οποία προέκυψε.

5.1.3.6 *wordcloud_tweets.py*

Η συνάρτηση αυτή παράγει ένα *Wordcloud* (*Σύννεφο Λέξεων*) που οπτικοποιεί τη συχνότητα εμφάνισης για τις *λέξεις* στο σύνολο των *Tweets*. Όπως και οι υπόλοιπες κλάσεις δέχεται τα προεπεξεργασμένα *Tweets* που προέρχονται από κάποιο συγκεκριμένο *Θέμα*. Η διαφορά είναι ότι τα λαμβάνει ως είσοδο στη μορφή μιας μοναδικής συμβολοσειράς. Ορίζουμε τα χαρακτηριστικά της εικόνας όπως φόντο, μέγεθος κλπ. μέσω των παραμέτρων της: **wordcloud = WordCloud(...)** όπου *Wordcloud* βιβλιοθήκη της *Python,* και προσαρμόζουμε το συνολικό κείμενο στην εικόνα δίνοντας το ως είσοδο στη συνάρτηση: **wordcloud.generate(...)**. Στην εικόνα με μεγαλύτερο μέγεθος φαίνονται οι *λέξεις* που έχουν μεγαλύτερη συχνότητα εμφάνισης.

5.2 Πλατφόρμες και Προγραμματιστικά Εργαλεία

5.2.1 Python

Η *Python* είναι μια γλώσσα υψηλού επιπέδου γενικού σκοπού. Χαρακτηρίζεται από το απλό συντακτικό της και τις λίγες γραμμές κώδικα που απαιτούνται για υλοποίηση λειτουργιών σε σχέση με άλλες γλώσσες προγραμματισμού. Τα παραπάνω καθιστούν την *Python* εύκολη στη χρήση και τον κώδικο που παράγεται αναγνώσιμο και συντηρήσιμο. Ένα ακόμα γνώρισμα κλειδί της γλώσσας είναι οι πολλές βιβλιοθήκες που διαθέτει, για κάθε πιθανή λειτουργία που μπορεί να φανταστεί κανείς. Αυτές δίνουν στην *Python* τη δυνατότητα να απορροφά χαρακτηριστικά άλλων γλωσσών με ένα απλό φόρτωμα μια βιβλιοθήκης, καθιστώντας την ένα ακόμα πιο οργανωμένο εργαλείο διευκολύνοντας ακόμα περισσότερο τον προγραμματιστή.

Αυτή η ύπαρξη μεγάλου αριθμού βιβλιοθηκών οφείλεται κυρίως στο γεγονός ότι η *Python* αναπτύσσεται ως ανοιχτό λογισμικό(open-source) για αυτό είναι πιο απλό για κάποιον να συνεισφέρει σε αυτήν. Συγκεκριμένα στην εργασία μας χρησιμοποιήσαμε την *Python 2* (έκδοση 2.7) την οποία και προτιμήσαμε από την *Python 3* λόγω του ότι αρκετές βιβλιοθήκες ενδέχεται να μην είναι διαθέσιμες στην πρόσφατη έκδοση. Στην εργασία μας χρησιμοποιήσαμε πληθώρα βιβλιοθηκών όπως οι *Tweepy, Py2neo, Scikit-Learn* κλπ. Η πλατφόρμα στην οποία στηρίχθηκε μεγάλο κομμάτι της ανάπτυξης του κώδικα είναι η το *PyCharm* της *JetBrains*, ένα Ολοκληρωμένο Περιβάλλον Ανάπτυξης (IDE) που παρέχει κάθε "άνεση" στον προγραμματιστή(Ανάλυση κώδικα, Debugging κλπ.) χωρίς να είναι ιδιαίτερα "βαριά" εφαρμογή. Έγινε χρήση της Professional Έκδοσης που δικαιούμαστε δωρεάν σαν φοιτητές.

5.2.2 Tweepy

Η *Tweepy* είναι μια από τις πιο ολοκληρωμένες βιβλιοθήκες της Python που προσφέρονται για σύνδεση του προγραμματιστή με τα *APIs* του Twitter. Η σύνδεση στα APIs γίνεται μέσω κλειδιών ταυτοποίησης που αντιστοιχούν στο λογαριασμό κάθε χρήστη (developer) του Twitter. Επιτρέπεται λοιπόν η πρόσβαση τόσο στα REST APIs όσο και στο Streaming API μέσω των εφαρμογών μας στην Python. Έτσι γίνεται δυνατές, εύκολα και απλά η συναλλαγές με το κοινωνικό δίκτυο του Twitter μέσω ενός περιβάλλοντος που ενδείκνυται για τροποποίηση αυτών και των δεδομένων που περιλαμβάνουν .

5.2.3 Py2neo

Η *py2neo* είναι μια βιβλιοθήκη στη μεριά του εξυπηρετούμενου και αποτελεί σημαντικό εργαλείο για εκμετάλλευση των δυνατοτήτων της Neo4j και της γλώσσας ερωτημάτων Cypher τόσο για αποθήκευση όσο και ανάκτηση δεδομένων από τη βάση μέσω εφαρμογών σε Python. Σε συνδυασμό με το Tweepy αποτελούν ένα ολοκληρωμένο σύστημα συλλογής δεδομένων από μία εξωτερική βάση όπως είναι το Twitter και αποθήκευση τους στην προσωπική μας βάση Neo4j.

5.2.4 Scikit-Learn

Η βιβλιοθήκη *scikit-learn* (ή *sklearn*) είναι η πιο διαδεδομένη για την εφαρμογή αλγορίθμων μηχανικής μάθησης σε Python. Αποτελει ένα σύνολο από απλά και αποτελεσματικά εργαλεία

για εξόρυξη και ανάλυση πάνω σε δεδομένα. Έχει χτιστεί με βάση τις "διάσημες" βιβλιοθήκες της Python *NumPy, SciPy* και *Matplotlib*. Εμείς κάνουμε χρήση των συναρτήσεων για συσταδοποίηση K-Means που αυτή παρέχει.

5.2.5 LDA library

Η βιβλιοθήκη *LDA* υλοποιεί τον αλγόριθμο μοντελοποίησης Θεμάτων (*Topic Modeling*) Latent Dirichlet Allocation (LDA) με χρήση καταρρεούμενης δειγματοληψίας *Gibbs* (*collapsed Gibbs sampling*). Ο αλγόριθμος της βιβλιοθήκης είναι γρήγορος και ανεξάρτητος λειτουργικού συστήματος εφαρμογής. Η διεπαφή ακολουθεί τις συμβάσεις που έχουν γίνει στη Scikit-Learn που είδαμε παραπάνω.

5.2.6 BokehJS

Η *Bokeh* είναι μια διαδραστική βιβλιοθήκη οπτικοποίησης της Python η οποία αποσκοπεί στην αναπαράσταση δεδομένων σε σύγχρονους web browsers. Καταφέρνει να παρέχει κομψή και περιεκτική κατασκευή πρωτότυπων γραφικών στο στύλ της D3.js (Διαδραστική βιβλιοθήκη οπτικοποίησης δεδομένων σε JavaScript) σε συνδυασμό με διαδραστικότητα υψηλής-απόδοσης πάνω σε πολύ μεγάλα σύνολα δεδομένων. Η *Bokeh* είναι ιδανική για δημιουργία εφαρμογών που αποσκοπούν στην οπτικοποίηση δεδομένων, κάνοντας την πληροφορία πιο εμφανή στον τελικό χρήστη.

5.2.7 Διεπαφή Neo4j-Cypher

Αποτελεί την εφαρμογή περιηγητή του Neo4j η οποία τρέχει πάνω στη βάση. Διαθέτει ένα ωραίο περιβάλλον για ανάπτυξη και τρέξιμο ερωτημάτων Cypher. Αυτό που το κάνει να ξεχωρίζει είναι ότι εκτός της κλασικής παρουσίασης των αποτελεσμάτων των ερωτημάτων σε πίνακες , τα οπτικοποιεί ως γράφους με κόμβους και ακμές.

5.2.8 HTML, Javascript, CSS

Οι HTML, JavaScript και CSS είναι 3 γλώσσες που χρησιμοποιούνται από τους web browsers, και λειτουργούν βέλτιστα μόνο όταν συνδυάζονται μεταξύ τους γι'αυτό και τις παρουσιάζουμε στο ίδιο κεφάλαιο. Η κάθε μία από τις 3 είναι υπεύθυνη για συγκεκριμένες λειτουργίες των ιστοσελίδων. Η *HTML(Hyper Text Markup Language)* είναι η κύρια γλώσσα σήμανσης στις ιστοσελίδες και χρησιμοποιείται για αναπαράσταση περιεχομένου σε αυτές το οποίο περιεχόμενο μπορεί να είναι κείμενο, εικόνα, βίντεο κλπ. Η JavaScript είναι μια ελαφριά γλώσσα προγραμματισμού σεναρίων (scripting language) που χρησιμοποιείται για έλεγχο της συμπεριφοράς και της ροής των ιστοσελίδων. Κύριο χαρακτηριστικό της είναι η δυναμική μεταβολή του περιεχόμενου μιας ιστοσελίδας και η εμφάνιση αυτής χωρίς ιδιαίτερη επιβάρυνση του εξυπηρετητή. Τέλος η CSS (Cascading Style Sheets) είναι μια γλώσσα φύλλων στύλ που χρησιμοποιείται για να ελέγχεται και να προσαρμόζεται η εμφάνιση ενός εγγράφου που έχει γραφτεί σε HTML. Αναφορικά κάποιοι από τους τομείς που παρεμβαίνει η CSS είναι το χρώμα, το στύλ, το μέγεθος και το σχήμα των στοιχείων της ιστοσελίδας.

5.2.9 Flask

Το *Flask* είναι ένα μικρο-πλαίσιο (microframework) για web εφαρμογές γραμμένο σε *Python* και βασισμένο στην εργαλειοθήκη *Werkzeug* και τη μηχανή προτύπων *Jinja2*. Κατηγοριοποιείται ως μικρό-πλαίσιο διότι δεν απαιτεί ή αναγκάζει τον προγραμματιστή στη χρήση συγκεκριμένων εργαλείων ή βιβλιοθηκών. Ωστόσο υποστηρίζει επεκτάσεις και να λειτουργήσουν σαν να ήταν εξ'αρχής ανεπτυγμένες για το *Flask*.

6

Ανάπτυξη Web-Based Εφαρμογής

Αναπτύξαμε μια Web Εφαρμογή μέσω της οποίας ο χρήστης μπορεί να διαλέξει μέσα από μια λίστα Θεμάτων που έχουμε στη βάση μας, αυτό που αυτός επιθυμεί καθώς και τη μέθοδο ανάλυσης που θα χρησιμοποιηθεί. Έτσι γίνεται πιο εύκολο να παρουσιαστούν τα αποτελέσματα και να τρέξει κάποιος σενάρια χωρίς να έχει ιδιαίτερη γνώση.

Τα συγκεκριμένα θέματα που συλλέξαμε στηρίζονται σε συμβάντα όπως είναι : η συνάντηση για το Διεθνές Οικονομικό Forum που έγινε το 2016 στο Davos για την οποία συλλέξαμε Tweets μέσω του hashtag ***#WEF16***, τα βραβεία μουσικής Grammys του 2016 μέσω του hashtag ***#Grammys2016***, τα βραβεία κινηματογράφου Oscar μέσω του hashtag ***#Oscars***, ο 50ος τελικός του Αμερικάνικου Ποδοσφαίρου (Superbowl) μέσω του hashtag ***#SuperBowl*** και το δημοψήφισμα που έγινε τον Ιούλιο του 2015 στην Ελλάδα μέσω του hashtag ***#dimopsifisma.***

Συνολικά στη βάση έχουμε συλλέξει δεδομένα για: *889076 Tweets*, *254835 Χρήστες*, *50811 Hashtags*, 124446 *συνδέσμους (URLs)* και *3542403 Σχέσεις* διάφορων τύπων, που είδαμε παραπάνω και συνδέουν τις οντότητες. Ο συνολικός χώρος που καταλαμβάνεται είναι περίπου 2.5 GB

6.1 Αρχιτεκτονικό Μοτίβο Σχεδίασης

Η εφαρμογή μας ακολουθεί το αρχιτεκτονικό μοτίβο σχεδίασης **Model-View-Controller(MVC).**

Στο ***Model*** το οποίο είναι υλοποιημένο σε *Python (models.py),* υλοποιούμε τις συναρτήσεις που επιτελούν τις λειτουργίες της εφαρμογής και διαχειρίζονται τα δεδομένα και τη λογική της. Οι συναρτήσεις αυτές τρέχουν στο παρασκήνιο ανεξαρτήτως τι βλέπει ο χρήστης. Δεχόμενες την είσοδο από τον *Controller* παρέχουν την έξοδο προς το *View*.

Στις συναρτήσεις του ***View,*** που είναι επίσης υλοποιημένες σε *Python (views.py),* ρυθμίζεται η έξοδος και η αναπαράσταση της πληροφορίας γραφικά προς τον χρήστη με βάση την είσοδο του.

Τέλος το τελευταίο μέρος, ο ***Controller***, είναι υπεύθυνος να δέχεται την είσοδο του χρήστη και να τη μετατρέπει σε εντολές προς τα άλλα 2 συστατικά μέρη. Αυτό το υλοποιούμε σε *HTML* σελίδες που εφαρμόζουν λειτουργικότητα *Javascript* όπου κρίνεται αναγκαίο.

6.2 Περιγραφή Βασικών Συναρτήσεων και Σελίδων

6.2.1 Modules

6.2.1.1 *models.py*

Σε αυτό το module ορίζεται η πρόσβαση στη βάση μας (Neo4j) καθώς και κάποια ερωτήματα *Cypher* προς αυτήν, τα οποία χειριζόμαστε με τη μορφή συναρτήσεων. Ενδεικτικά αναφέρουμε τη συνάρτηση ***hashtag_cooccurence()*** η οποία δέχεται ως είσοδο από το χρήστη ένα συγκεκριμένο *Hashtag* και του επιστρέφει τα *Hashtags* που συνυπάρχουν πιο πολλές φορές μαζί με αυτό σε *Tweets*. Το συγκεκριμένο ερώτημα παρουσιάστηκε σαν κώδικας στο Κεφάλαιο 2.

6.2.1.2 *views.py*

Είναι το module το οποίο είναι υπεύθυνο για το χειρισμό της ροής των σελίδων που βλέπει ο τελικός χρήστης. Για την κάθε οθόνη συνδέει τις αποκρίσεις του χρήστη σε αυτή με τις ενέργειες που τους αντιστοιχούν. Ενδεικτικά μπορεί να οδηγήσει σε μετάβαση προς άλλη οθόνη ή να εμφανίσει το ζητούμενο περιεχόμενο στην τρέχουσα. Για παράδειγμα στην σελίδα που επιλέγει ο χρήστης τις παραμέτρους *Συσταδοποίησης Κειμένου* η κλάση *views.py* έχει συνάρτηση που τις δέχεται και επιστρέφει την αντίστοιχη μοντελοποίηση πίσω σε αυτόν, ελέγχοντας έτσι την ροή.

6.2.2 Σελίδες

6.2.2.1 *layout.html*

Εδώ ορίζεται ο γενικός σχεδιασμός και η δομή όλων των επιμέρους σελιδών. Αυτό επιτυγχάνεται με ορισμό πάγιων συστατικών, ορισμό του πλαισίου που θα εμπεριέχεται το περιεχόμενο των άλλων σελίδων αλλά και με χρήση βιβλιοθηκών της Javascript για εμφάνιση των συστατικών της (*bootstrap*). Εκτός των άλλων ορίζεται η καρτέλα περιήγησης προς όλες τις σελίδες. Οι υπόλοιπες σελίδες λέμε ότι κάνουν επέκταση(extend) αυτης.

6.2.2.2 *index.html*

Σελίδα που εμφανίζεται κατά την είσοδο στην εφαρμογή, με το αντίστοιχο μήνυμα και κάποιες γενικές πληροφορίες προς τον χρήστη.

6.2.2.3 *analytics.html*

Σε αυτή τη σελίδα παρουσιάζονται στο χρήστη κάποιες επιλογές ανάλυσης των δεδομένων στη βάση μας. Μετά την επιλογή της επιθυμίας του, του επιστρέφονται τα αποτελέσματα εντός αυτής.

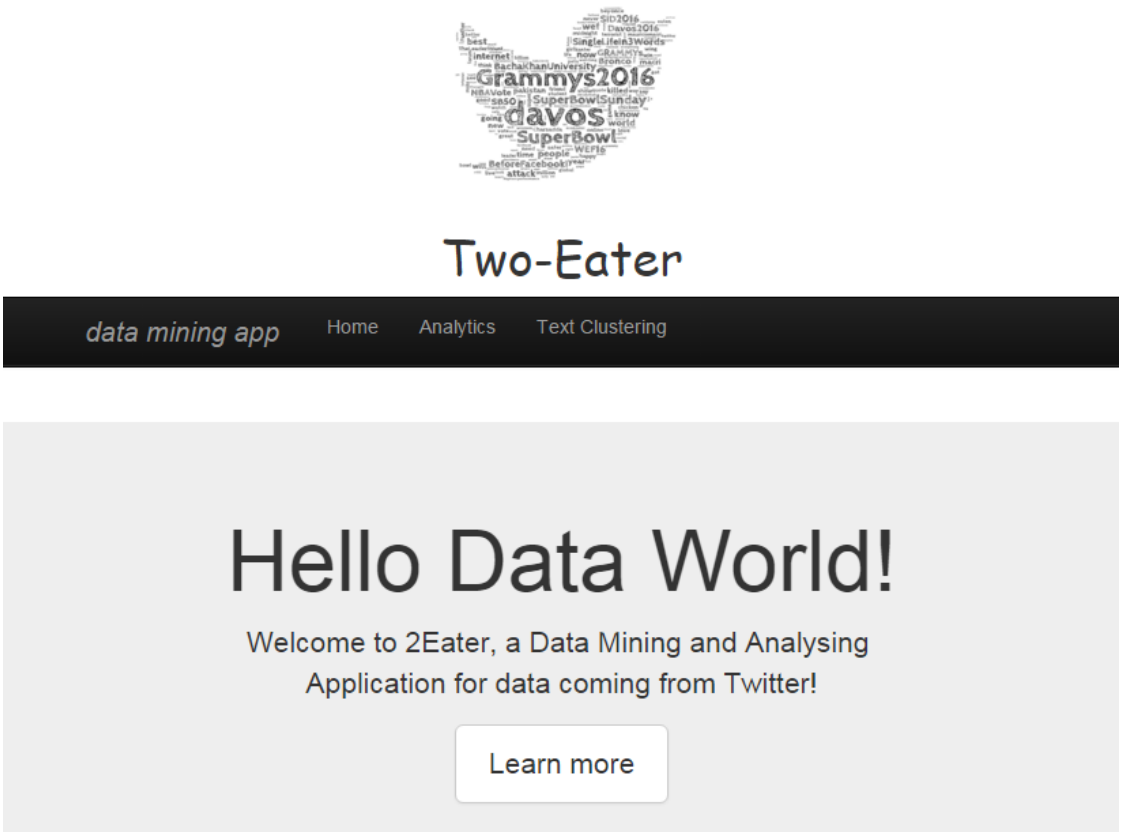
6.2.2.4 *models.html*

Στη σελίδα αυτή παρουσιάζεται και το κύριο ενδιαφέρον της μελέτης μας. Δίνεται στο χρήστη η δυνατότητα να εφαρμόσει στο Θέμα της επιλογής του κάποια μέθοδο *Συσταδοποίησης Κειμένου (Text Clustering)* μέσω των αλγορίθμων *K-Means* και *LDA*. Ο χρήστης ακόμα

καλείται να επιλέξει αν τα κείμενα ως προς τα οποία θα γίνει η ανάλυση θα είναι χωρισμένα *ανά Tweet* ή *ανά Tweets* Χρηστών του κοινωνικού δικτύου. Τέλος ο χρήστης δίνει παραμετρικά και τον αριθμό συστάδων/θεμάτων. Μετά τον χειρισμό της εισόδου του χρήστη του επιστρέφεται η οθόνη με τα αποτελέσματα του μοντέλου της επιλογής του που έχουν προκύψει σε διαδραστιό διάγραμμα διασποράς σε *Javascript* μέσω της βιβλιοθήκης *Bokeh* της *Python*, όπως αναφέραμε και προηγουμένως.

6.3 *Σενάρια Χρήσης Web-Based Εφαρμογής*

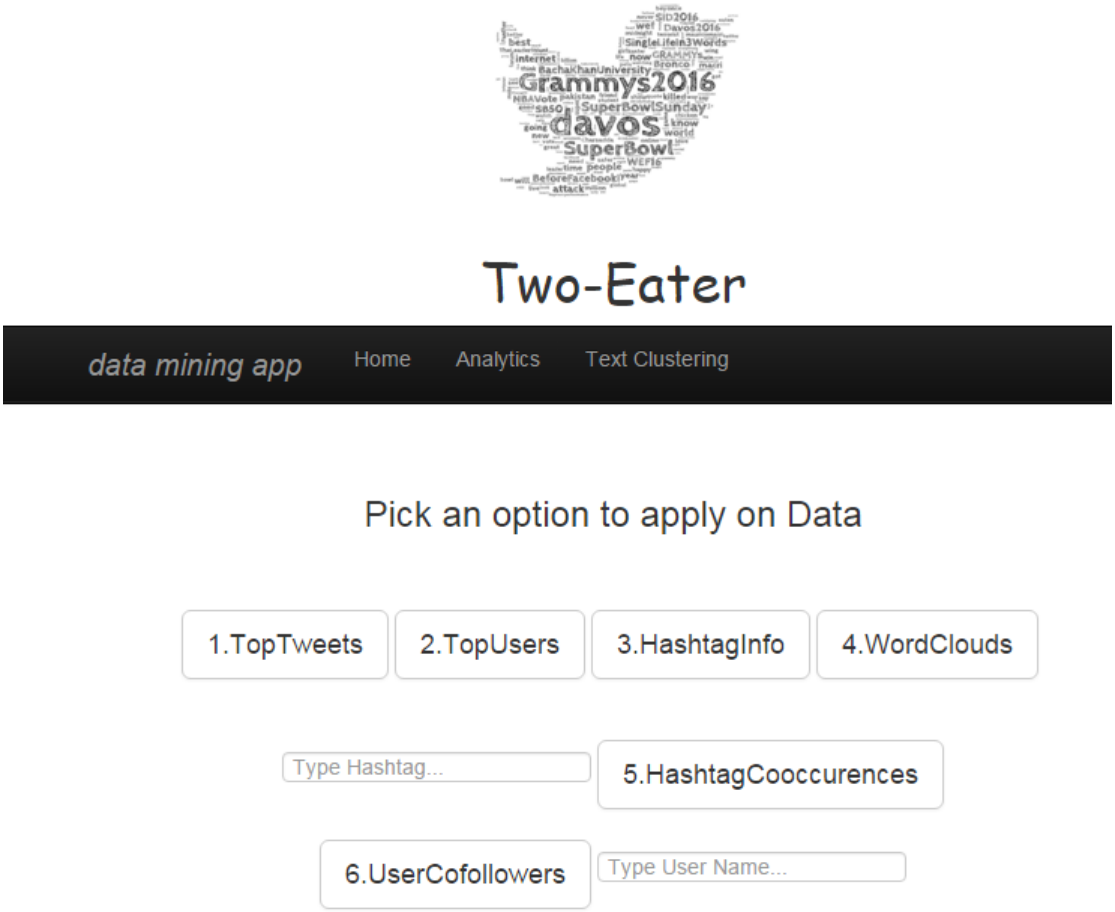
Αρχικά ο χρήστης του διαδικτύου με την είσοδο του στην ιστοσελίδα, θα βρεθεί στην αρχική μας οθόνη (***Home***). Η αρχική σελίδα πέρα από πληροφορίες σχετικά με τις δυνατότητες της εφαρμογής, παρέχει τη δυνατότητα περιήγησης στις επιμέρους σελίδες της και τις αντίστοιχες λειτουργίες τους. Η αρχική οθόνη φαίνεται παρακάτω:



Εικόνα 6.1: Αρχική οθόνη web εφαρμογής

Όπως είναι ευδιάκριτο από την παραπάνω εικόνα είναι δυνατή η μετάβαση στη σελίδα ***Analytics****(analytics.html)* και στη σελίδα ***Text Clustering****(models.html).*

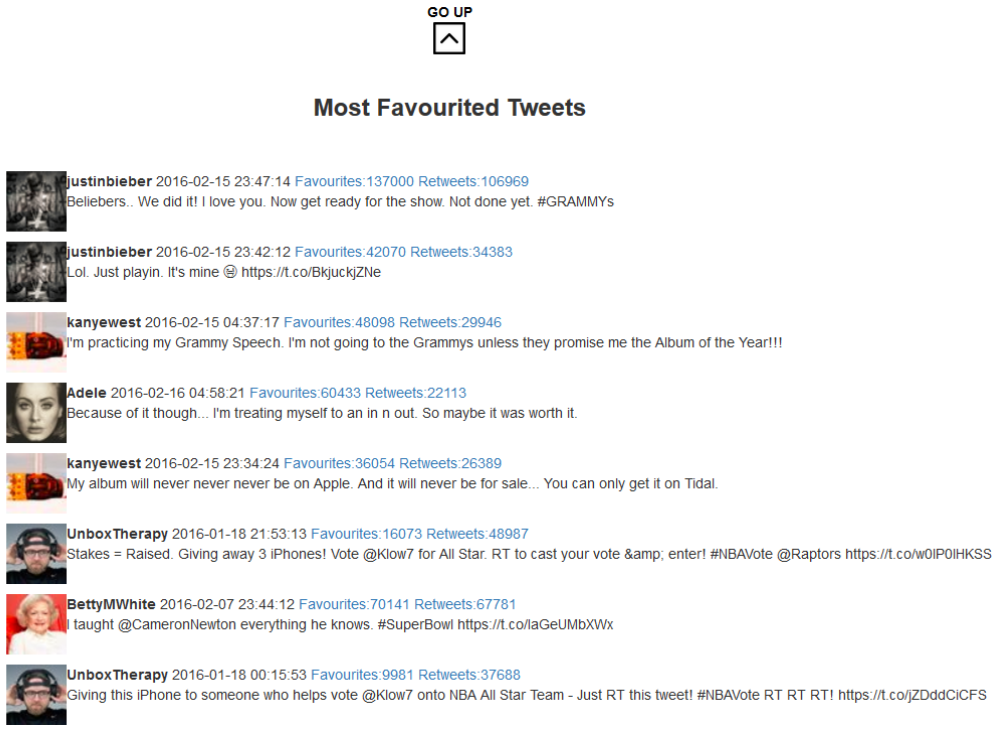
Έστω ότι επιλέγεται η μετάβαση στην σελίδα ***Analytics*** η οποία φαίνεται παρακάτω. Εκει παρέχεται πληθώρα δυνατοτήτων ανάλυσης των δεδομένων μας όπως : εύρεση των αγαπημένων *Tweets* και *Χρηστών*, αναπαράσταση των ανά *Hashtag* δεδομένων σε σύννεφα Λέξεων(*Word Clouds*) κλπ.



Εικόνα 6.2: Οθόνη Analytics web εφαρμογής

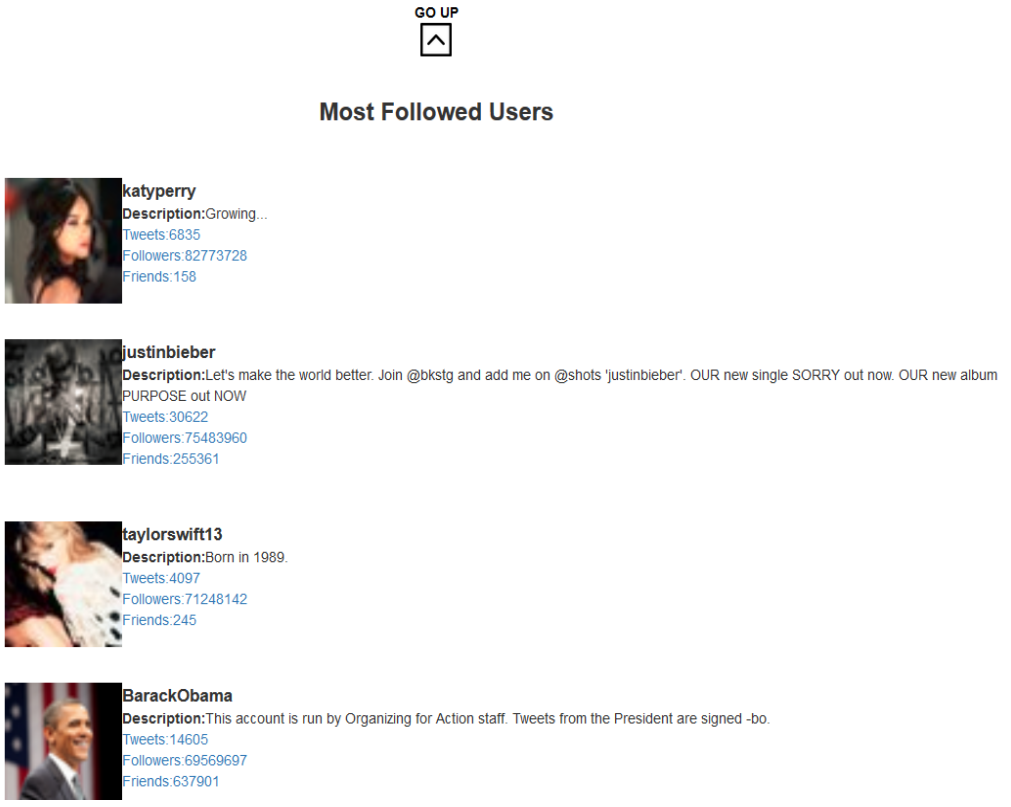
Παρακάτω θα δούμε τις οθόνες που μπορούν να προκύψουν ανάλογα την επιλογή μας στην οθόνη ***Analytics***. Αυτές είναι:

Αν επιλέξουμε την καρτέλα ***TopTweets*** μας επιστρέφεται λίστα με τα Tweets σε φθίνουσα σειρά *Favourites* δηλαδή από το πιο αγαπημένο στο λιγότερο αγαπημένο εντός της βάσης μας.



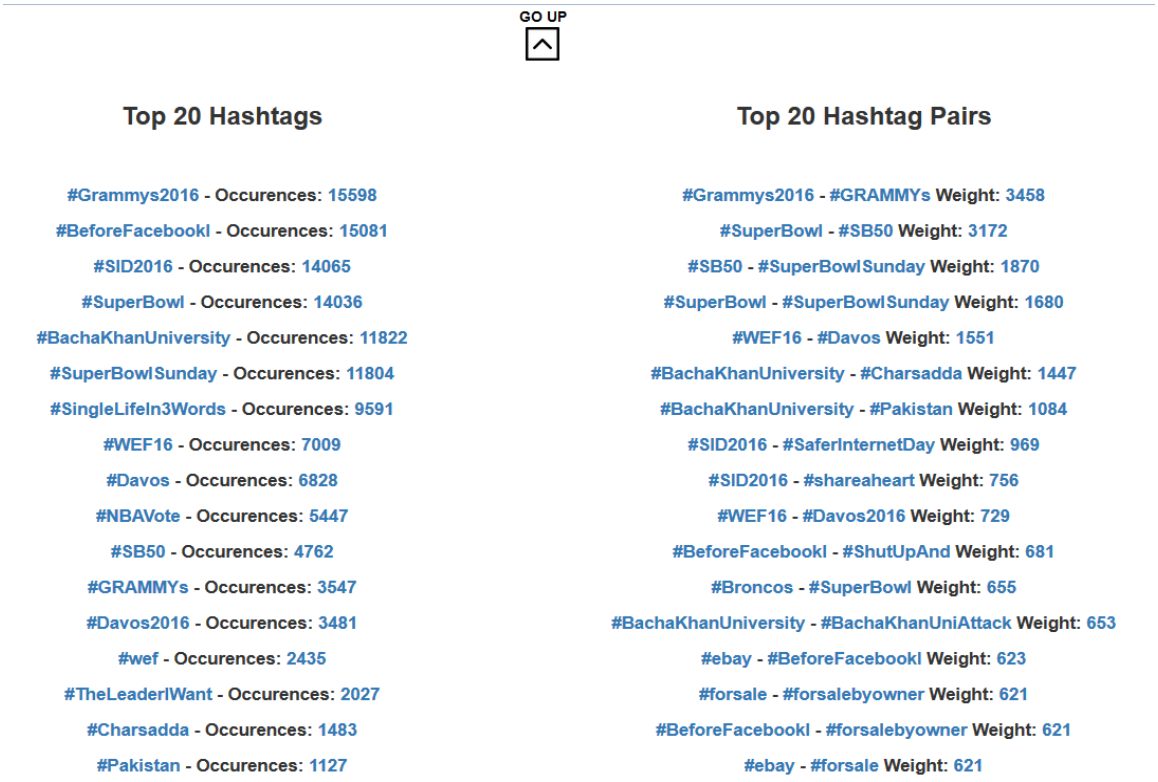
Εικόνα 6.3: Οθόνη Εμφάνισης Δημοφιλών Tweets

Αν η επιλογή είναι ***TopUsers*** τότε εμφανίζεται λίστα με τους πιο δημοφιλείς *χρήστες* του δικτύου μας με βάση τον αριθμό των *Ακολούθων* τους (*Followers*).



Εικόνα 6.4: Οθόνη Εμφάνισης Δημοφιλών Χρηστών

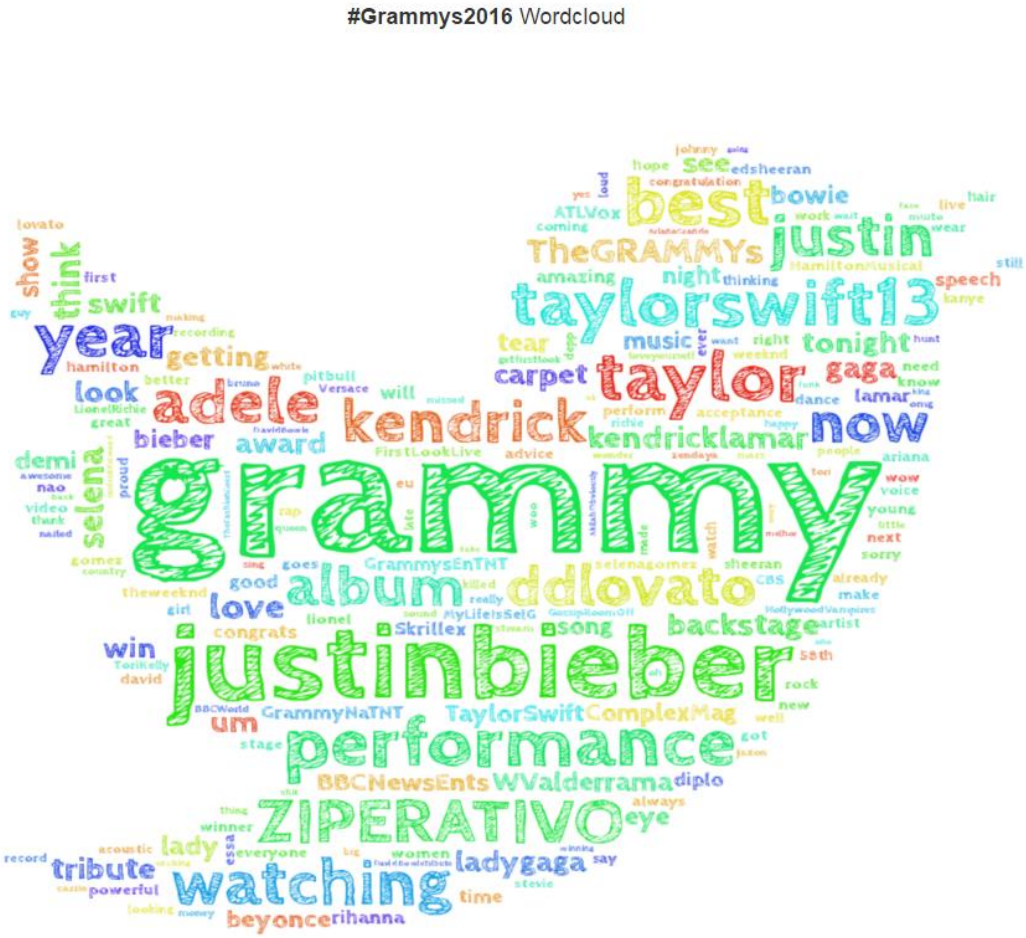
Στην περίπτωση της επιλογής ***HashtagInfo*** εμφανίζονται σε 2 στήλες, πληροφορίες σχετικές με τα *Hashtags* της βάσης. Στην αριστερή εμφανίζονται τα κορυφαία *Hashtags* ως προς τον αριθμό που έχουν συμπεριληφθεί σε *Tweet* εντός του συνόλου μας και στην δεξιά τα κορυφαία ζευγάρια *Hashtags*, δηλαδή των *Hashtags* που έχουν χρησιμοποιηθεί τις περισσότερες φορές μαζί σε *Tweet*.



Εικόνα 6.5: Οθόνη παρουσίασης κορυφαίων Hashtags και κορυφαίων συνδυασμών Hashtags

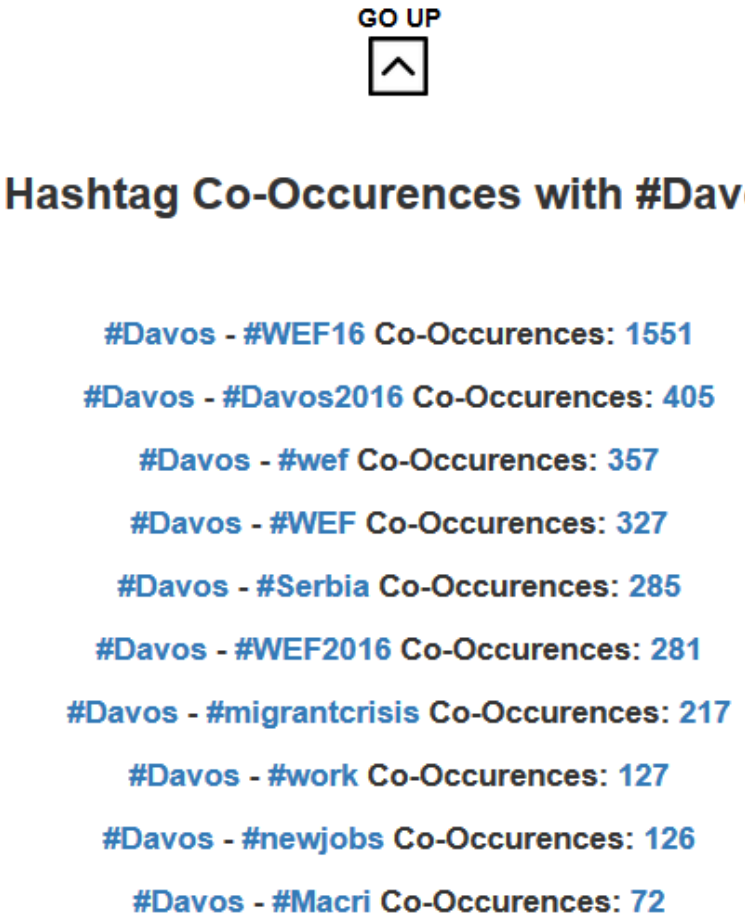
Τώρα σε περίπτωση που επιλέξει κάποιος την καρτέλα ***Wordclouds*** τότε γίνεται μετάβαση σε οθόνη όπου ο *χρήστης* της εφαρμογής καλείται να επιλέξει το *Hashtag* για το οποίο αυτός επιθυμεί να δεί το *Σύννεφο Λέξεων*. Με το *Σύννεφο Λέξεων* που προκύπτει μπορεί κανείς εύκολα να διακρίνει τις δημοφιλείς *λέξεις* εντός του συνόλου *Κειμένων* γύρω από το συγκεκριμένο *Hashtag*. Τόσο η οθόνη επιλογής όσο και το αποτέλεσμα για συγκεκριμένο *Hashtag* φαίνονται παρακάτω:

Εικόνα 6.6: Οθόνη επιλογής εμφάνισης Word Cloud της επιθυμίας μας



Εικόνα 6.7: Word Cloud σε περίπτωση που γίνει η επιλογή του Θέματος #Grammys2016

Πέρα των καρτελών υπάρχει η δυνατότητα εισαγωγής κειμένου που θα αντιπροσωπεύει το *Hashtag* για το οποίο θέλουμε να βρούμε Hashtags που το συνοδεύουν τις περισσότερες φορές *ανά Tweet* και επιστροφής των αποτελεσμάτων με επιλογή της καρτέλας ***HashtagCooccurences***.



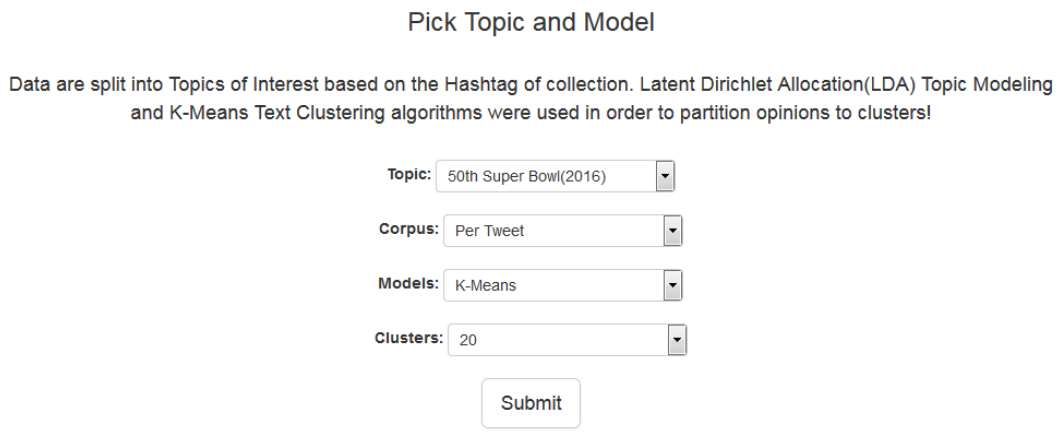
Εικόνα 6.8: Οθόνη εμφάνισης κορυφαίων συνυπάρξεων Hashtags μαζί με #Davos

Σε παρόμοια λογική με εισαγωγή του *χρήστη Twitter* της επιλογής μας βρίσκουμε τους *χρήστες* που *συν-ακολουθούνται* πιο πολλές φορές μαζί με αυτόν. Έτσι μπορούμε να πάρουμε πληροφορία συσχέτισης μεταξύ των *χρηστών* που ακολουθεί κάποιος και του *χρήστη* επιλογής. Η επιστροφή των αποτελεσμάτων της μορφής που φαίνεται παρακάτω, πυροδοτείται με την επιλογή ***UserCoffolowers***.

Εικόνα 6.9: Οθόνη εμφάνισης κορυφαίων συν-ακολουθούμενων μαζί με @kmitsotakis

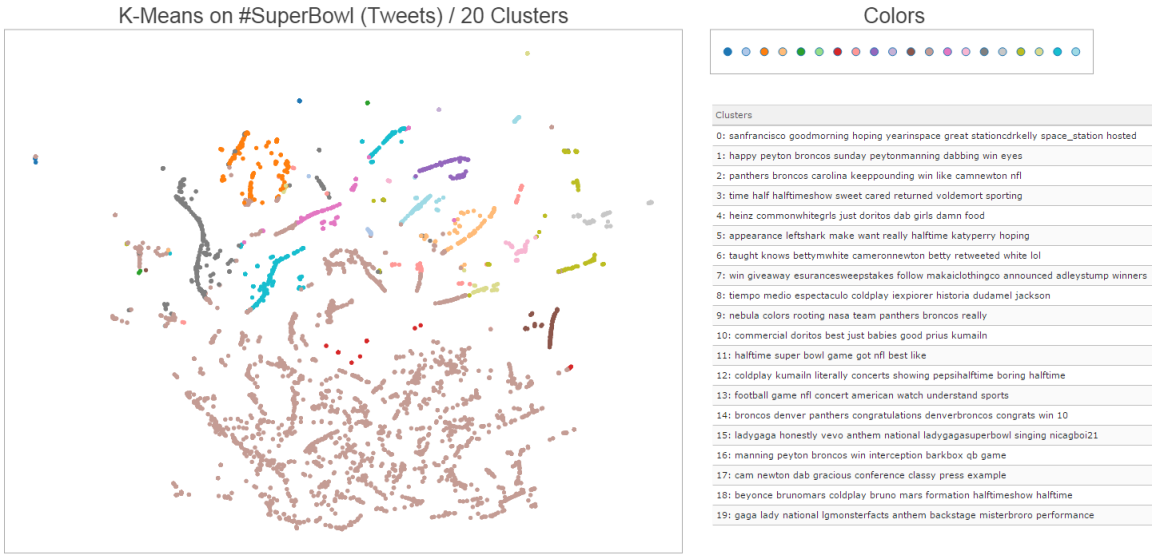
Τώρα σε περίπτωση που επιλέξουμε τη μετάβαση στη σελίδα ***Text Clustering*** οδηγούμαστε στο σύστημα που αφορά το κύριο κομμάτι της μελέτης μας. Αυτό αφορά τη *Μοντελοποίηση Θεμάτων*, το διαχωρισμό δηλαδή των *Κειμένων* μας σε Συστάδες. Σε αυτή την σελίδα δίνονται αρκετές επιλογές στο *χρήστη* σχετικά με το μοντέλο με βάση τις οποίες θα παραχθεί το διάγραμμα διασποράς των *Κειμένων*. Ο χρήστης καλείται να επιλέξει το ***μοντέλο*** που επιθυμεί να χρησιμοποιήσει ανάμεσα στα *LDA* και *K-Means*, το ***Hashtag*** από το οποίο θα προέρχονται τα κείμενα , τον ***τρόπο διαχωρισμού των Κειμένων*** *(ανά χρήστη, ανά Tweet)* και τον ***αριθμό των Συστάδων/Θεμάτων*** που επιθυμεί να δημιουργηθούν κατά την *μοντελοποίηση*.

Δύο σενάρια χρήσης αυτής της οθόνης με τα αντίστοιχα αποτελέσματα φαίνονται παρακάτω:

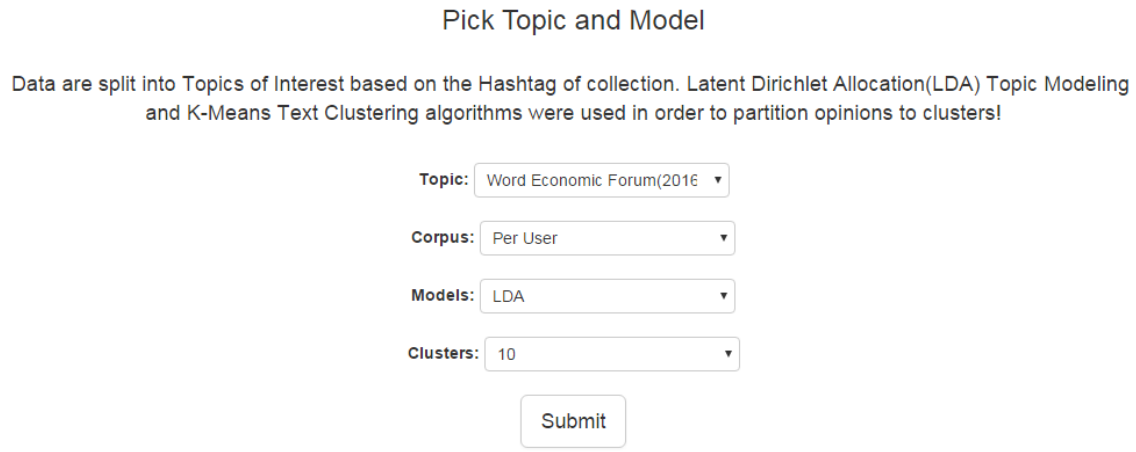


Εικόνα 6.10: 1ο Σενάριο Οθόνης επιλογής παραμέτρων μοντελοποίησης Θεμάτων

Τα αποτελέσματα επιλογής των παραπάνω παραμέτρων, δηλαδή : Θέματος Ενδιαφέροντος το *50th Super Bowl(2016) (#SuperBowl)* , διαχωρισμό Κειμένων *Per Tweet*, τον *K-Means* για μοντέλο και *20 Συστάδες για ομαδοποίηση* εμφανίζονται στην παρακάτω οθόνη:

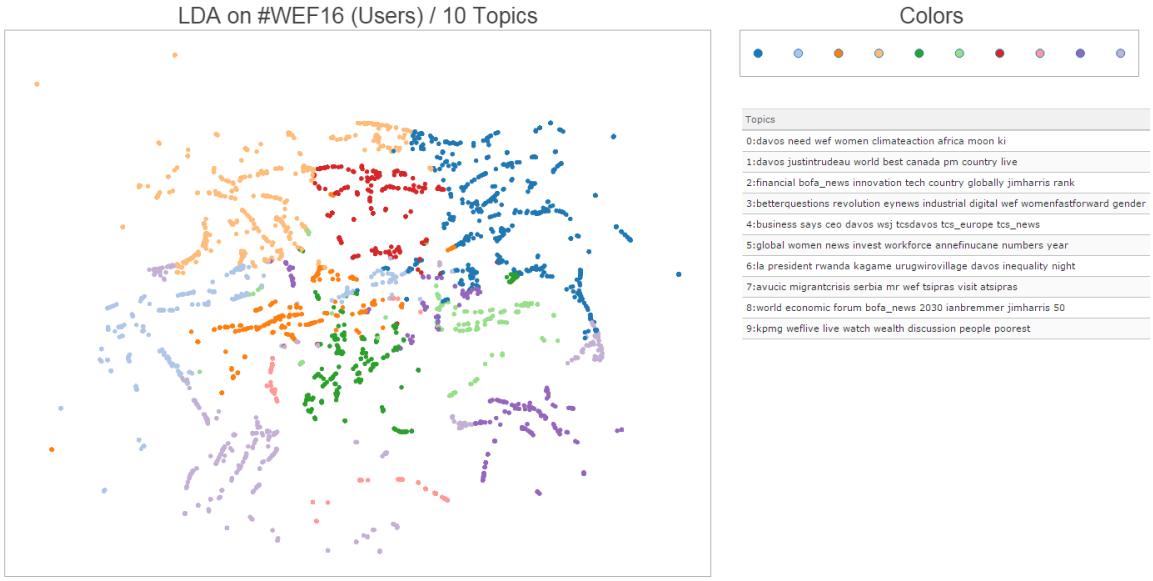


Εικόνα 6.11: Οθόνη Διαγράμματος Διασποράς για το 1ο σενάριο επιλογής παραμέτρων



Εικόνα 6.12: 2ο Σενάριο Οθόνης επιλογής παραμέτρων μοντελοποίησης Θεμάτων

Τα αποτελέσματα επιλογής των παραπάνω παραμέτρων, δηλαδή : Θέματος Ενδιαφέροντος το *World Economic Forum(2016) (#WEF2016)* , διαχωρισμό Κειμένων *Per User*, τον *LDA* για μοντέλο και *10 Συστάδες για ομαδοποίηση,* εμφανίζονται στην παρακάτω οθόνη:



Εικόνα 6.13: Οθόνη Διαγράμματος Διασποράς για το 2ο σενάριο επιλογής παραμέτρων

7

Επίλογος

7.1 Σύνοψη και Συμπεράσματα

Στα πλαίσια αυτής της διπλωματικής εργασίας αναπτύχθηκε ένα σύστημα το οποίο συλλέγει δεδομένα από το δίκτυο του *Twitter*, τα αποθηκεύει σε μια βάση δεδομένων, τα φιλτράρει κρατώντας την χρήσιμη πληροφορία από αυτά και επιτυγχάνει *μοντελοποίηση Θεμάτων* με βάση το λεκτικό περιεχόμενο αυτών, μέσω των μοντέλων *K-Means* και *Latent Dirichlet Allocation (LDA)*. Τέλος έχει στηθεί μια Web Εφαρμογή, στην οποία δίνονται πληθώρα επιλογών στον τελικό χρήστη για ανάλυση των δεδομένων.

Τα συμπεράσματα που προκύπτουν από την εργασία αυτή είναι ότι το σύστημα που δημιουργήθηκε και βασίζεται πάνω στους αλγόριθμους *Συσταδοποίησης K-Means* και *Μοντελοποίησης Θεμάτων LDA* μπορεί να χρησιμοποιηθεί για τον Εντοπισμό Θεμάτων Συζήτησης μέσα σε σύνολα δεδομέναν αλλά και για την κατάταξη των επιμέρους Κειμένων του συνόλου σε κάποιο από αυτά τα Θέματα.

7.2 Μελλοντικές Επεκτάσεις

Στον κόσμο του διαδικτύου σήμερα, η χρησιμοποίηση τεχνικών μοντελοποίησης των δεδομένων σε Θέματα έχει μεγάλη προοπτική, αφού το μεγαλύτερο μέρος των online πληροφοριών συγκεντρώνονται γύρω από Θέματα. Δεν θα ήταν υπερβολή αν πούμε ότι μία ιστοσελίδα μπορεί να περιλαμβάνει υλικό που χωρίζεται σε εκατοντάδες διαφορετικά Θέματα. Η δύναμη του K-Means και κυρίως του LDA όπως είδαμε είναι πέρα της απλής συσταδοποίησης των Κειμένων και η εξαγωγή κατατοπιστικών Θεμάτων γιά το περιεχόμενο αυτών.

Τα εξαγόμενα Θέματα θα μπορούσαν να χρησιμοποιηθούν σε πολλές εφαρμογές. Μιά πιθανή εφαρμογή θα ήταν η αυτόματη δημιουργία Θεμάτων από το περιεχόμενο μιας ιστοσελίδας ή ενός δικτύου, από τα οποία ο χρήστης θα επιλέγει το Θέμα για το οποίο επιθυμεί να ενημερωθεί, έτσι ώστε να τού εμφανιστεί το αντίστοιχο περιεχόμενο(κείμενα, Tweets κλπ.) πού έχει ήδη κατηγοριοποιηθεί σε αυτό.

Ένα άλλο παράδειγμα εφαρμογής είναι τα Θέματα να αποτελούν απάντηση σε ερώτηματα που γίνονται σε μια μηχανή αναζήτησης. Έτσι ανάλογα με την είσοδο του χρήστη, θα συμπεραίνεται το Θέμα πού ψάχνει και θα του επιστρέφονται τα επιθυμητά σχετικά αποτελέσματα.

8

Βιβλιογραφία

[Blei, et al.,2003]

D. M. Blei, A. Y. Ng, and M. I. Jordan. "*Latent dirichlet allocation.*" The Journal of Machine Learning Research, 3:993–1022, 2003.

[Steyvers & Griffiths, 2007]

Mark Steyvers and Tom Griffiths, 2007. "*Probabilistic Topic Models.*", pages 427–446. Psychology Press, February

[Griffiths & Steyvers, 2004]

Griffiths, T. L., and M. Steyvers. "*Finding Scientific Topics."* Proceedings of the National Academy of Sciences 101.Supplement 1 (2004): 5228-235.

[Boyd, Golder, & Lotan, 2010]

boyd, danah, Scott Golder, and Gilad Lotan. 2010. *"Tweet, Tweet, Retweet: Conversational Aspects of Retweeting on Twitter."* HICSS-43. IEEE: Kauai, HI, January 6.

[Java,et al.,2007]

Java, Akshay, Xiaodan Song, Tim Finin, and Belle Tseng. "*Why We Twitter*." Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 Workshop on Web Mining and Social Network Analysis - WebKDD/SNA-KDD '07 (2007).

[Krishnamurthy, Gill & Arlitt, 2008]

Krishnamurthy, Balachander, Phillipa Gill, and Martin Arlitt. "*A Few Chirps about Twitter*." Proceedings of the First Workshop on Online Social Networks - WOSP '08 (2008).

[Ramage, Dumais & Liebling, 2010]

Daniel Ramage, Susan Dumais, Dan Liebling . "*Characterizing Microblogs with Topic Models*". Proceedings of the Fourth International AAAI Conference on Weblogs and Social Media

[Yu Xiao,et al., 2010]

Yu Xiao "*A Survey of Document Clustering Techniques & Comparison of LDA and moVMF*", December 10, 2010

[Steinbach, Karypis & Kumar 2000]

M. Steinbach, G. Karypis, and V. Kumar. "*A comparison of document clustering techniques*." Technical Report 00-034, University of Minnesota, 2000.

[Honeycutt & Herring,2009]

"*Beyond Microblogging: Conversation and Collaboration via Twitter*." 2009 42nd Hawaii International Conference on System Sciences (2009).

[Naaman, Boase & Lai, 2010]

Naaman, Mor, Jeffrey Boase, and Chih-Hui Lai. "*Is It Really about Me?"*Proceedings of the 2010 ACM Conference on Computer Supported Cooperative Work - CSCW '10 (2010).

[Chang, Boyd-Graber & Blei, 2009]

J. Chang, J. Boyd-Graber, and D. M. Blei. *"Connections between the lines: augmenting social networks with text."* In KDD ’09: Proceedings of the 15th ACM SIGKDD

international conference on Knowledge discovery and data mining, pages 169–178, 2009.

[Phan, Nguyen & Horiguchi, 2008]

X.-H. Phan, L.-M. Nguyen, and S. Horiguchi. *"Learning to classify short and sparse text & web with hidden topics from large-scale data collections."* In WWW ’08: Proceedings of the 17th International Conference on World Wide Web, pages 91–100, 2008.

[Ramage, Hall, Nallapati & Manning, 2009]

D. Ramage, D. Hall, R. Nallapati, and C. D. Manning. *"Labeled lda: a supervised topic model for credit attribution in multi-labeled corpora."* In EMNLP ’09: Proceedings of the Conference on Empirical Methods in Natural Language Processing, pages 248–256. Association for Computational Linguistics, 2009.

[Rosen-Zvi, Griffiths, Steyvers & Smyth, 2004]

M. Rosen-Zvi, T. Griffiths, M. Steyvers, and P. Smyth. *"The author-topic model for authors and documents."* In UAI ’04: Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence, pages 487–494, 2004.

[Zhang, Giles, Foley & Yen, 2007]

H. Zhang, C. L. Giles, H. C. Foley, and J. Yen. *"Probabilistic community discovery using hierarchical latent gaussian mixture model."* In AAAI’07: Proceedings of the 22nd National Conference on Artificial Intelligence, pages 663–668, 2007.

[Weng,Lim,Jiang & He, 2010]

WENG, Jianshu; LIM, Ee Peng; JIANG, Jing; and He, Qi. *"Twitterrank: Finding Topic-Sensitive Influential Twitterers."* (2010). ACM International Conference on Web Search and Data Mining (WSDM 2010). , 261. Research Collection School Of Information Systems.

[Cataldi, Di Caro & Schifanella, 2010]

Mario Cataldi, Luigi Di Caro, Claudio Schifanella. "Emerging Topic Detection on Twitter based on Temporal and Social Terms Evaluation." (2010)

[WCA]

http://sites.stat.psu.edu/~ajw13/stat505/fa06/19_cluster/09_cluster_wards.html

[SKL] *"Scikit-learn: Machine Learning in Python"*, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011

[UL] *"Unsupervised Learning"* Zoubin Ghahramani Gatsby Computational Neuroscience Unit University College London, UK zoubin@gatsby.ucl.ac.uk

http://www.gatsby.ucl.ac.uk/~zoubin

[K] "*Kmeans vs Mini Batch Kmeans: A comparison*" Javier Béjar Departament de Llenguatges i Sistemes Informàtics Universitat Politècnica de Catalunya

[DMC] "*Data Mining Curriculum*". ACM SIGKDD. 20060430.Retrieved 20111028.

[TD] https://dev.twitter.com/overview/documentation

[ST] statista.com

[SB] statisticbrain.com

[GDB] *"O'Reilly Graph Databases"*, Ian Robinson,Jim Webber & Emil Eifrem

[N] *"Neo4j in Action"*, Aleksa Vukotic & Nicki Watt

[LN] "Learning Neo4j ", Rik Van Bruggen

[**NOSQLDM**] http://www.slideshare.net/emileifrem/nosql-east-a-nosql-overview-and-the-benefits-of-graph-databases

[DMML] *"Big Data, Data Mining & Machine Learning"*, Jared Dean

[KM]

https://sites.google.com/site/dataclusteringalgorithms/kmeansclusteringalgorithm

[DM] http://en.wikipedia.org/wiki/Data_mining

[WTSN] https://en.wikipedia.org/wiki/T-distributed_stochastic_neighbor_embedding

[van der Maaten, 2013]

Laurens van der Maaten. "*Barnes-Hut-SNE*" Pattern Recognition and Bioinformatics Group, Delft University of Technology Mekelweg 4, 2628 CD Delft, The Netherlands

[KM] https://sites.google.com/site/dataclusteringalgorithms/k-means-clustering-Algorithm

[UVSS] http://stanford2011.wikispaces.com/Class07_04.19

[WSIL] https://en.wikipedia.org/wiki/Silhouette_(clustering)

[TR] http://blogs.lse.ac.uk/impactofsocialsciences/2015/07/10/social-media-research-tools-overview/