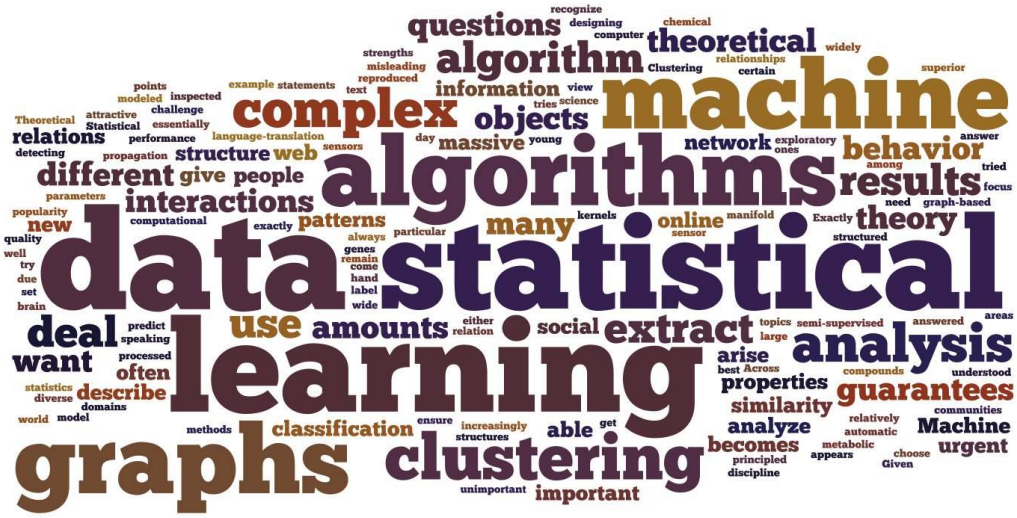
Πανεπιστήμιο Δυτικής Μακεδονίας

Τμήμα Μηχανικών Πληροφορικής και Τηλεπικοινωνιών

Διπλωματική Εργασία

Εκτίμηση Γνώμης Σε Κοινωνικά Δίκτυα Μέσω Τεχνικών Μηχανικής Μάθησης



Λύτος Αναστάσιος

Επιβλέπων: Σαρηγιαννίδης Παναγιώτης, Επίκουρος Καθηγητής Π.Δ.Μ.

Μέλη Εξεταστικής Επιτροπής:

Π. Αγγελίδης, Αναπληρωτής Καθηγητής

Κ. Στεργίου, Αναπληρωτής Καθηγητής

Μάρτιος 2017, Κοζάνη

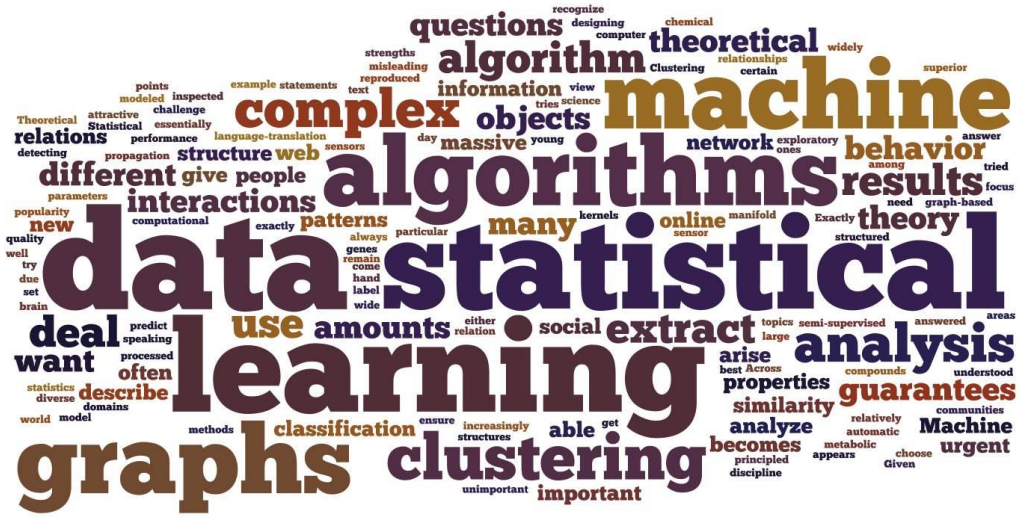
Εικόνα Εξωφύλλου: Koné Mamadou Tadiou, PhD
The Future Of Human Evolution, Artificial Intelligence, Machine Learning

University of Western Macedonia

Department of Informatics and Telecommunications

Master Thesis

Opinion Mining in Social Networks Using Machine Learning Techniques



Lytos Anastasios

Supervisor: Sarigiannidis Panagiotis, Assistant Professor U.O.W.M

Examination Committeeç:

P. Aggelidis, Assistant Professor, U.O.W.M

K. Stergiou, Assistant Professor, U.O.W.M.

March 2017, Kozani

Book Cover: Koné Mamadou Tadiou, PhD
The Future Of Human Evolution, Artificial Intelligence, Machine Learning

Περίληψη

Η παρούσα διπλωματική εργασία εξετάζει τις τεχνικές μηχανικής μάθησης που μπορούν να συμβάλλουν στην μελέτη, κατανόηση και επεξεργασία της φυσικής γλώσσας (natural language process). Επίσης συγκρίνει διαφορετικά λεξικά που δημιουργήθηκαν για ανάλυση συναισθήματος σε ανεξάρτητη βάση δεδομένων.

Τα δεδομένα που χρησιμοποίησα στην εκπόνηση αυτής της διπλωματικής δεν προέρχονται από κάποια γνωστή βάση δεδομένων, αλλά δημιούργησα εγώ τη βάση δεδομένων εξάγοντας τα απαραίτητα δεδομένα από δύο από τα πιο δημοφιλή κοινωνικά δίκτυα, Facebook και Twitter. Τα δεδομένα που εξήγαγα από τα δύο κοινωνικά δίκτυα, προέρχονται από δημοφιλείς και επίσημες σελίδες/λογαριασμούς τεχνολογικών κολοσσών. Πιο συγκεκριμένα, οι λογαριασμοί τα δεδομένα των οποίων χρησιμοποίησα ανήκουν στη Lenovo, Asus και Acer.

Εφόσον δημιούργησα τη βάση δεδομένων το επόμενο βήμα της εργασίας είναι η αξιολόγηση των δεδομένων που έχω συλλέξει από ήδη γνωστά λεξικά. Τα συγκεκριμένα λεξικά έχουν δημιουργηθεί στοχεύοντας στην ερμηνεία των λέξεων σε συναισθηματικό επίπεδο. Το κάθε λεξικό έχει δημιουργηθεί με διαφορετική τεχνική, βασιζόμενο σε διαφορετική προσέγγιση υλοποίησης, αλλά όλα τα λεξικά έχουν κοινό στόχο, την ακριβέστερη αποτύπωση της δύναμης (συναισθηματικού φόρτου) της κάθε λέξης. Για τη συγκεκριμένη διπλωματική εργασία χρησιμοποιώ εννέα (9) διαφορετικά λεξικά.

Έχοντας ολοκληρώσει την αξιολόγηση των προτάσεων το επόμενο βήμα της εργασίας είναι η μελέτη και η σύγκριση διαφορετικών τεχνικών μηχανικής μάθησης. Ο στόχος των τεχνικών μηχανικής μάθησης που χρησιμοποιώ είναι η σωστή πρόβλεψη της αποδοχής των δημοσιεύσεων από τους χρήστες του κοινωνικού δικτύου που πραγματοποιεί η κάθε εταιρία.

Τα σημεία προς έρευνα σε αυτή την εργασία κυμαίνονται πάνω σε τρεις βασικούς άξονες. Ο πρώτος είναι ποια είναι τα συγκεκριμένα χαρακτηριστικά της γλώσσας τα οποία προκαλούν στου χρήστες συναισθήματα. Στο twitter οι χρήστες μπορούν εκφράσουν τα συναισθήματα τους μέσω 3 αντιδράσεων, retweet, favorite και response. Στο Facebook υπάρχουν περισσότεροι τρόποι να εκφράσει ένας χρήστης συναισθήματα, αφού προσφέρεται η επιλογή συγκεκριμένης αντίδρασης από ένα σύνολο 6 αντιδράσεων σε μία κατάσταση(post), εκτός του σχολιασμού (comment) και της αναδημοσίευσης (share).

Το δεύτερο σημείο προς έρευνα είναι η μελέτη των διάφορων αλγορίθμων μηχανικής μάθησης και πόσο αποτελεσματικοί είναι στην μελέτη της φυσικής γλώσσας. Χρησιμοποίησα αλγορίθμους classification, εκτός των κλασσικών τεχνικών, χρησιμοποίησα ακόμη Support Vector Machines και νευρωνικά δίκτυα. Συνολικά χρησιμοποίησα 7 αλγορίθμους classification.

Το τελευταίο σημείο στο οποίο επικεντρώνομαι είναι η βαρύτητα του λεξικού για τη σωστή κατηγοριοποίηση και πρόβλεψη αποδοχής των δεδομένων προς μελέτη. Πόσο μεγάλο ρόλο στη σωστή πρόβλεψη αποδοχής διαδραματίζει το λεξικό; Ποια είναι τα χαρακτηριστικά που πρέπει να διαθέτει ένα λεξικό; Πώς διαφοροποιούνται τα λεξικά μεταξύ τους; Αυτά είναι μερικά από τα ερωτήματα που θα απαντήσω σε αυτή την διπλωματική εργασία.

Λέξεις κλειδιά: Facebook API, Twitter API, crawler, οπτικοποίηση δεδομένων, λεξικό, συναισθηματική ανάλυση, μηχανική μάθηση, εξόρυξη δεδομένων, classification, support vector machines, νευρωνικά δίκτυα,

Abstract

The present Master thesis deals with the techniques of machine learning that can contribute in the study and understanding in the field of natural language process. Also, compares different lexicon created for sentiment analysis on an independent database.

The data that I used in the development of this thesis don't derive from some known database, but I created a database extracting the necessary data from two of the most popular social networks, Facebook and Twitter. The data that I extracted from the two social networks come from popular and official pages/account of well-known technological companies. More specific, the accounts' data I used belong in Lenovo, Asus and Acer.

Provided the creation of the database, the next step of the thesis is the evaluation of the collected data from established lexicons. The specific lexicons have been created aiming in the interpretation of the words in sentiment level. Every lexicon has been created with different technique, created on different approach of implementation, but with identical target, a more accurate depiction of the emotional force of each word. For this thesis, I use nine (9) different dictionaries.

Having completed the evaluation of the data, the next step of the thesis is the study and comparison of different machine learning techniques. The target of the machine learning techniques I use is the correct prediction of the embracement of each company's posts from the users of the social network

The points of interest in this thesis range in three basis axes. The first axe is the specific characteristics of the language that cause emotions to the users. In twitter the users can express their feelings through 3 reactions, retweet, favorite and response. In Facebook, there are more ways for a user to express feelings, since there is the choice of a specific reaction from a pool of 6 reactions in a post, without considering the choices of comment and share.

Το δεύτερο σημείο προς έρευνα είναι η μελέτη των διάφορων αλγορίθμων μηχανικής μάθησης και πόσο αποτελεσματικοί είναι στην μελέτη της φυσικής γλώσσας. Χρησιμοποίησα αλγορίθμους classification, εκτός των κλασσικών τεχνικών, χρησιμοποίησα ακόμη Support Vector Machines και νευρωνικά δίκτυα. Συνολικά χρησιμοποίησα 7 αλγορίθμους classification.

The second point of interest is the study of various machine learning techniques and how effective are in the field of natural language process. I used classification algorithms, besides the traditional techniques, I also used Support Vector Machines and neural networks. In total I used 7 classification algorithms.

The last point I focus is the importance of the lexicon for the correct classification and prediction of embracement for the data I study. How important is the lexicon for the

correct classification? Which are the characteristics a lexicon should have? What makes the difference between lexicons? These are some the questions I will answer in this thesis.

Keywords: Facebook API, Twitter API, crawler, data visualization, sentiment analysis, machine learning, data mining, classification, support vector machines, neural networks

Πίνακας Περιεχομένων

1. Εισαγωγή

1.1 Πρόβλημα και κίνητρα για τη συγγραφή της εργασίας

Η ολοένα και μεγαλύτερη χρήση των κοινωνικών δικτύων έχει οδηγήσει και σε μεγαλύτερη συλλογή δεδομένων, μέσα από τις συμπεριφορές και αντιδράσεις των χρηστών των κοινωνικών δικτύων. Βιομηχανίες και εταιρίες marketing ωθούμενες από αυτή την πληθώρα δεδομένων αναζητούν τρόπους να εξάγουν πληροφορίες για προϊόντα και υπηρεσίες.

Η επιστημονική κοινότητα έχει εντοπίσει αυτή την ευκαιρία για εκμετάλλευση των νέων δεδομένων και κάνει προσπάθειες για την κατανόηση αυτού του τεράστιου όγκου δεδομένων. Έχουν υπάρξει μελέτες πάνω στην ανάλυση συναισθήματος (sentiment analysis) με τη κάθε μία να επικεντρώνεται σε διαφορετικούς τομείς και να διαθέτει διαφορετικούς στόχους.

Ως sentiment analysis ορίζεται η υπολογιστική και στατιστική μελέτη απόψεων, αισθημάτων και συναισθημάτων εκφραζόμενων σε γραπτό λόγο [1]. Οι περισσότερες έρευνες σε αυτό το πεδίο επικεντρώνονται στην κατηγοριοποίηση (classification) κειμένου ανάλογα με το συναίσθημα που εκφέρουν, το οποίο μπορεί να είναι θετικό, αρνητικό ή ουδέτερο [2] [3] [4], ενώ σε μερικές περιπτώσεις η κατηγοριοποίηση γίνεται δυαδική χωρίς την παρουσία ουδετερότητας [5].

Στην πράξη οι περισσότερες μέθοδοι προσέγγισης για το Sentiment Analysis υιοθετούν μία στρατηγική δύο βημάτων [2]. Το πρώτο βήμα είναι η κατηγοριοποίηση του κειμένου προς ανάλυση ως προς την υποκειμενικότητα του, το κείμενο κατηγοριοποιείται ως υποκειμενικό ή αντικειμενικό (ουδέτερο). Το δεύτερο βήμα στο Sentiment Analysis είναι η κατηγοριοποίηση του κειμένου όσον αφορά την χροιά ή πολικότητα (polarity) του, όπου οι υποκειμενικές προτάσεις κατηγοριοποιούνται ως θετικές ή αρνητικές.

Το κύριο ερώτημα στο οποίο απαντάει το sentiment analysis είναι ‘τι σκέφτονται οι άλλοι’, μέσα από ανάλυση κειμένου σε αξιολογήσεις προϊόντων και άρθρα ειδήσεων. Με αυτό τον τρόπο οι καταναλωτές αναζητούν πληροφορίες σχετικά με ένα προϊόν που τους

ενδιαφέρει και οι κατασκευαστές/έμποροι δίνουν όλο και μεγαλύτερη προσοχή στις διαδικτυακές κριτικές για τα προϊόντα και τις υπηρεσίες που προσφέρουν. Αυτή η δυνητική εμπορική εκμετάλλευση που μπορεί να πραγματοποιηθεί πάνω στο Sentiment Analysis έχει προσελκύσει το ενδιαφέρον διαφόρων επιστημονικών κοινοτήτων που επικεντρώνονται σε διαφορετικά ερευνητικά πεδία, όπως μηχανική μάθηση(machine learning), εξόρυξη δεδομένων (data mining) και επεξεργασία φυσική γλώσσας (natural language processing).

Πάνω στο αντικείμενο της κατηγοριοποίησης του Sentiment Analysis έχει υπάρξει αρκετά μεγάλη έρευνα, η οποία όμως επικεντρώνεται στην κατηγοριοποίηση μεγάλων κειμένων/εγγράφων όπως κριτικών [6]. Τα Tweets [7] [8], οι κοινοποιήσεις στο Facebook [9] και οποιαδήποτε μορφή microblogging διαφέρουν από τις κριτικές εξαιτίας τους σκοπού δημιουργίας τους: οι κριτικές εκφέρουν στρωτές και δομημένες σκέψεις του συγγραφέα, ενώ κάθε μορφή microblogging γράφεται σε πιο ανεπίσημη μορφή με περιορισμό λέξεων.

Σε γενικές γραμμές μπορεί να θεωρηθεί ότι το microblogging δεν είναι τόσο προσεκτικά δομημένο όσο μία κριτική. Παρόλο αυτά μπορούν να προσφέρουν σημαντικές πληροφορίες τόσο σε εμπορικό όσο και ακαδημαϊκό επίπεδο. Έχουν γίνει κάποιες έρευνες στον τομέα του Sentiment Analysis σε επίπεδο φράσης, συγκεκριμένα πάνω σε tweets [10] [11] και σε δημοσιεύσεις στο Facebook [12].

Όλες οι έρευνες διαλέγουν το κοινωνικό δίκτυο από το οποίο θα συλλέξουν τα δεδομένα, το λεξικό πάνω στο οποίο θα αξιολογήσουν τα δεδομένα που έχουν εξορύξει καθώς και τους classifiers (ταξινομητές) που θα χρησιμοποιήσουν. Με αυτό τον τρόπο δεν εντοπίζονται αλλαγές ανάμεσα στον τρόπο χρήσης διαφορετικών κοινωνικών δικτύων, δεν γίνεται εύρεση βέλτιστου classifier και δεν υπάρχει σύγκριση λεξικών για την απόδοση τους σε διαφορετικά είδη κειμένων.

1.2 Σχετική βιβλιογραφία

Τα τελευταία χρόνια παρατηρείται μία έκρηξη στον αριθμό των χρηστών των διαφόρων κοινωνικών δικτύων. Αυτή η μαζική χρήση των κοινωνικών δικτύων έχει ανοίξει νέους ορίζοντες σε πολλούς και διαφορετικούς ερευνητικούς τομείς. Η εκτίμηση της κοινής γνώμης (opinion mining) και η επεξεργασία της φυσικής γλώσσας (natural

language process) είναι δύο τομείς οι οποίοι παρουσιάζουν πρόσφορο έδαφος για τη χρησιμοποίηση των δεδομένων που προέρχονται από τα κοινωνικά δίκτυα.

Οι πρώιμες μελέτες πάνω στο sentiment analysis ξεκίνησαν την δεκαετία του 1990, με κύριο στόχο την κατηγοριοποίηση λέξεων ή φράσεων ανάλογα με τη σημασιολογική τους έννοια [13]. Σε αυτές τις πρώτες δουλειές χρησιμοποιήθηκαν γλωσσικοί ευρετικοί μηχανισμοί (heuristics) ή προεπιλεγμένα σύνολα λέξεων ως σπόροι (seeds). Κύριος στόχος αυτών των εργασιών ήταν η ταξινόμηση ολόκληρων εγγράφων, θεωρώντας ότι ο μέσος σημασιολογικός προσανατολισμός των λέξεων σε ένα κείμενο, είναι κατ' επέκταση ένας δείκτης για την συνολική ερμηνεία του κειμένου, κατά πόσο αυτή είναι θετική ή αρνητική.

Η συστηματική και μαζική χρήση πλατφορμών microblogging (μικρό-ιστολογίου) και κοινωνικών δικτύων από χρήστες του Διαδικτύου έχει οδηγήσει την επιστημονική κοινότητα που ασχολείται με το sentiment analysis να εκμεταλλευτεί αυτούς τους νέους πόρους δεδομένων. Εκατομμύρια μηνυμάτων εμφανίζονται καθημερινά σε κοινωνικά δίκτυα, όπως τα Facebook [9], Twitter [7], Tumblr [14], Google+ [15] κ.α. Στις συγκεκριμένες ιστοσελίδες οι χρήστες κοινοποιούν τις απόψεις τους για πολλά και διαφορετικά θέματα, δημιουργώντας μία απύθμενη βάση δεδομένων η οποία προσφέρεται για εφαρμογή μεθόδων και τεχνικών πάνω στο sentiment analysis. Ακόμη οι διαφορετικοί τρόποι έκφρασης σε κάθε πλατφόρμα και οι άτυποι κανόνες που διέπουν το κάθε κοινωνικό δίκτυο δημιουργούν νέες προκλήσεις και ιδιαιτερότητες ως προς την μελέτη των δεδομένων που προσφέρουν.

Οι διαφορές στις τεχνικές ανάλυσης της φυσικής γλώσσας, ανάλογα με το μέγεθος του κειμένου το οποίο μελετάται είναι το κυρίως θέμα ενός πολύ ενδιαφέροντος άρθρου ανασκόπησης [16]. Μέσα από το άρθρο κατανοούνται οι διαφορές στην επεξεργασία ολόκληρου κειμένου, πρότασης και ειδικού θέματος, καθώς επίσης δίνονται οι βασικές τεχνικές για την εξόρυξη γνώμης από φυσικό λόγο.

Η χρησιμότητα των εργαλείων γραμματικής ανάλυση (POS Tagger), οι διαφορές επίσημου/ανεπίσημου λόγου και σημαντικότητα των εικονιδίων emoticons και των hashtags είναι οι άξονες στους οποίους επικεντρώνεται η εργασία των Kouloumpis, Wilson και Moore [3].

Η κατηγοριοποίηση των δεδομένων που εξάγονται από το Twitter με κριτήριο τη χρήση θετικών και αρνητικών εικονιδίων emoticons είναι το κύριο θέμα έρευνας για την εργασία των Go, Bhayani και Huang [11]. Στην οποία εξετάζονται τρεις διαφορετικοί classifiers για την απόδοση τους σε πρόβλημα δυαδικής κατηγοριοποίησης. Η καινοτομία αυτής της εργασίας είναι η εισαγωγή του όρου distant supervised learning, για να τονίσουν τη διαφορά με τον όρο του full supervised learning.

Μία άλλη επιστημονική μελέτη [4] επικεντρώνεται στα λεξιλογικά φαινόμενα που παρατηρούνται στα microblogs και προσπαθεί να τα κατηγοριοποιήσει με βάση θετικά και αρνητικά emoticons (noisy labelled data). Επίσης εξετάζει την επιρροή των n-grams και προτείνει συγκεκριμένη μεθοδολογία για τη σωστή δημιουργία συλλογής δεδομένων.

Ο συνδυασμός των μεθόδων noisy labelled data και των χειροκίνητων μεθόδων αξιολόγησης είναι ένα ακόμη θέμα το οποίο έχει εξεταστεί [5]. Με τα αποτελέσματα της έρευνας να είναι θετικά προς αυτή τον τρόπο αξιολόγησης των δεδομένων.

Η προσθήκη πεδίων στις εγγραφές της βάσης δεδομένων, για χαρακτηριστικά που συναντώνται μόνο στο Twitter και την επίδραση τους στην σωστή πρόβλεψη μέσω classifiers είναι ένα ακόμη θέμα το οποίο έχει μελετηθεί [10]. Στην ίδια έρευνα έχει πραγματοποιηθεί και σύγκριση της απόδοσης των classifiers, όταν η ανάλυση του φυσικού λόγου γίνεται με διαφορετικό n-gram.

Η ανάλυση συναισθήματος και η εκτίμηση γνώμης έχει εφαρμογές και σε κοινωνικοπολιτικά πλαίσια. Έχει υπάρξει περιπτωσιολογική μελέτη (case study) για τις αμερικανικές εθνικές εκλογές του 2012 [17], η οποία επικεντρώνεται στις αντιδράσεις των χρηστών του Twitter σε πραγματικό χρόνο κατά τη διάρκεια σημαντικών πολιτικών εκδηλώσεων (ομιλίες, debates, κ.α.). Η συγκεκριμένη μελέτη εντυπωσιάζει εξαιτίας της επεκτάσιμης αρχιτεκτονικής της, βάση της οποίας μπορεί να διαχειριστεί ξαφνικά έντονες ριπές δεδομένων.

Τα περισσότερα επιστημονικά άρθρα περί εξόρυξη κοινής γνώμης και συναισθηματικής ανάλυσης σε κείμενο microblogging, εξάγουν τα δεδομένα τους από το Twitter, ενώ τα επιστημονικά άρθρα τα οποία εξάγουν δεδομένα από το Facebook είναι αισθητά λιγότερα. Ένα ιδιαίτερα καινοτόμο και πλήρης άρθρο πάνω στην συναισθηματική ανάλυση εκμεταλλεύτηκε τα δεδομένα που μπορούν να εξαχθούν από το Facebook μέσω plug-in εφαρμογής [12]. Οι συγγραφείς του άρθρου δημιουργήσαν την εφαρμογή για την εξαγωγή των δεδομένων, χρησιμοποίησαν υβριδικό μοντέλο για την κατηγοριοποίηση, συνδυάζοντας τεχνικές μηχανικής μάθησης και λεξικά συναισθήματος.

1.3 Συμβολή εργασίας

Η πλειοψηφία των εργασιών πάνω στο Sentiment Analysis γίνεται μόνο σε μία συγκεκριμένη βάση δεδομένων, με ένα συγκεκριμένο λεξικό, με ορισμένες από πριν δυνατές βαθμολογίες και με τη χρήση ενός (1) μέχρι τριών (3) classifiers. Εφόσον έχει πραγματοποιηθεί η εξαγωγή της βάσης δεδομένων στις περισσότερες των περιπτώσεων η κάθε πρόταση (ή tweet) αξιολογείται χειροκίνητα από κάποιον ερευνητή.

Στην παρούσα εργασία ελέγχω την επίδραση της πρότασης (tweet ή δημοσίευσης στο Facebook) μέσω της αντιδράσεις χιλιάδων χρηστών που έχουν επιλέξει να παρακολουθούν τα νέα και τις κοινοποιήσεις συγκεκριμένων εταιρειών. Με αυτό τον τρόπο μειώνεται η δυνατότητα λάθους εκτίμησης από τους ερευνητές που αξιολογούν χειροκίνητα τις προτάσεις καθώς και ο χρόνος που χρειάζεται για την χειροκίνητη αξιολόγηση.

Το δεύτερο σημείο στο οποίο καινοτομώ πάνω στην έρευνα του Sentiment Analysis είναι η δοκιμή και η σύγκριση διαφορετικών λεξικών για δεδομένα τα οποία εξάγονται από διαφορετικά κοινωνικά δίκτυα. Η μέχρι τώρα κυρίαρχη τακτική που ακολουθείται στο Sentiment Analysis είναι η αξιολόγηση των δεδομένων από ένα μόνο λεξικό με τρεις(θετική, αρνητική, ουδέτερη) ή δύο(θετική, αρνητική) πιθανές καταστάσεις.

Εν αντιθέσει αυτής της τακτικής, δοκίμασα εννιά (9) διαφορετικά λεξικά, χωρίς να ορίζω εκ των προτέρων δυνατές τιμές. Δηλαδή τα σκορ που λαμβάνει ένα tweet μπορεί να είναι ανάμεσα σε δύο ή τρεις τιμές, αν το λεξικό παρουσιάζει μικρή διασπορά στα σκορ που αποδίδει στις λέξεις ή να κυμαίνεται ανάμεσα σε είκοσι (20) πιθανές τιμές, αν το σύνολο των σκορ που αποδίδει το λεξικό παρουσιάζει μεγάλη διακύμανση.

Όσον αφορά τους classifiers που χρησιμοποιούνται για Sentiment Analysis, στις περισσότερες ερευνητικές εργασίες δεν δίνεται βάρος στη διαφορετική απόδοση των classifiers, για αυτό σε πολλές περιπτώσεις γίνεται χρήση μόνο ενός (1) ή δύο (2) διαφορετικών αλγορίθμων classification. Συνήθως οι classifiers που χρησιμοποιούνται είναι υλοποιήσεις Μηχανών Διανυσμάτων Υποστήριξης (Support Vector Machine - SVM), ταξινομητών Bayesian και λίγο πιο σπάνια υλοποιήσεις Δέντρων Απόφασης (Decision Tree). Στην εργασία μου συγκρίνω την απόδοση εφτά (7) διαφορετικών classifiers, πραγματοποιώ αναζήτηση βέλτιστων παραμέτρων για αυτούς τους classifiers ανάλογα με τα δεδομένα εισόδου.

Τέλος, παρουσιάζω ποια δομή και ποια χαρακτηριστικά πρέπει να έχει ένα λεξικό το οποίο καλείται να αξιολογήσει δημοσιεύσεις κοινωνικών δικτύων. Είναι σημαντικό να γίνει κατανοητό ότι λεξικά που παρουσιάζουν καλή απόδοση σε συντεταγμένα και δομημένα κείμενα δεν είναι αναγκαίο να παρουσιάζουν υψηλή απόδοση σε δημοσιεύσεις που προέρχονται από κοινωνικά δίκτυα. Η χαμηλή απόδοση λεξικών σε συγκεκριμένου είδους δεδομένα με ιδιαίτερα χαρακτηριστικά (εικονίδια emoticon, hashtags, χρήση αργκό κ.α.) δεν σημαίνει ότι τα συγκεκριμένα λεξικά δεν μπορούν να χρησιμοποιηθούν σε Sentiment Analysis, απλά δεν συστήνονται για τη χρήση τους σε συγκεκριμένο είδους δεδομένα.

1.4 Συνοπτική παρουσίαση και δομή εργασίας

Η εργασία χωρίζεται σε κεφάλαια, όπου το κάθε ένα καλύπτει συγκεκριμένες πτυχές του συνόλου και μπορεί να υπάρξει αυτόνομα. Προς όφελος της αυτονομίας των

κεφαλαίων δεν έχω χωρίσει την εργασία σε θεωρητικό και πειραματικό μέρος, αντίθετα προσφέρω το θεωρητικό υπόβαθρο για κάθε ενέργεια που πραγματοποιώ στο ίδιο κεφάλαιο με το πειραματικό μέρος.

Η συγκεκριμένη δομή της εργασίας βοηθά τον αναγνώστη να κατανοήσει τα προβλήματα που αντιμετωπίζονται, τις σχεδιαστικές αποφάσεις που έχω λάβει, το σκεπτικό πίσω από την χρησιμοποίηση συγκεκριμένων αλγορίθμων και τέλος να επέλθουν τα συμπεράσματα της εργασίας.

Συνοπτικά η εργασία ακολουθεί την παρακάτω δομή:

- Συλλογή των δεδομένων που χρησιμοποιήθηκαν για τη δημιουργία της βάσης δεδομένων που χρησιμοποιήθηκε. Πώς λειτουργεί το λογισμικό εξαγωγής δεδομένων, ποια δεδομένα είναι αυτά που χρησιμοποιούνται και πώς.
- Δημιουργία βάσης δεδομένων. Τι θα οριστεί ως πεδίο, ποια θα είναι τα χαρακτηριστικά του, τι τύπου μεταβλητές θα είναι τα χαρακτηριστικά.
- Εμπλουτισμός της υπάρχουσας βάσης δεδομένων με τη χρήση εργαλείου για τον εντοπισμό των μερών του λόγου που διαθέτει το προς εξέταση κομμάτι φυσικής γλώσσας.
- Οπτικοποίηση των δεδομένων, έτσι ο χρήστης να έχει μία οπτική απεικόνιση του προβλήματος που μελετάται.
- Θεωρητική εισαγωγή στις τεχνικές μηχανικής μάθησης. Επιβάλλεται ο αναγνώστης να αποκτήσει τις θεμελιώδεις αρχές κάθε αλγορίθμου που χρησιμοποιείται, έτσι ώστε να καταλάβει γιατί χρησιμοποιούνται, τι αποτέλεσμα περιμένουμε από τον αλγόριθμο και πώς αξιολογούμε αυτό το αποτέλεσμα.
- Μετά το θεωρητικό υπόβαθρο θα ακολουθήσει η εκτέλεση του κάθε αλγορίθμου μαζί με σχολιασμό των αποτελεσμάτων, έτσι ώστε να υπάρχει μία φυσιολογική ροή στην κατανόηση της εργασίας. Με αυτό τον τρόπο δεν δημιουργούνται κενά στην κατανόηση της εργασίας κι ο αναγνώστης ακολουθεί βήμα προς βήμα την εξέλιξη της.
- Επισκόπηση, σύγκριση και ερμηνεία των αποτελεσμάτων που έχω εξάγει
- Στο τέλος της εργασίας εμφανίζονται τα συμπεράσματα που εξήχθησαν από όλα τα παραπάνω βήματα της εργασίας και προτείνονται μελλοντικές επεκτάσεις

2. Συλλογή και Προεπεξεργασία Δεδομένων

Το πρώτο βήμα της εργασίας είναι η συλλογή των δεδομένων μου. Όπως έχει επισημανθεί δεν χρησιμοποίησα μία έτοιμη βάση δεδομένων, αλλά δημιούργησα τη δική μου μαζεύοντας δεδομένα από κοινωνικά δίκτυα.

Τα δεδομένα που εξήγαγα χαρακτηρίζονται ως δημόσια (public). Αυτός ο χαρακτηρισμός δίνεται στα δεδομένα τα οποία είναι προσβάσιμα από τον καθένα ανά πασά στιγμή χωρίς να χρειάζεται να διαθέτει λογαριασμό στο συγκεκριμένο κοινωνικό δίκτυο.

Η συλλογή των δεδομένων από το Διαδίκτυο συνήθως πραγματοποιείται από προγράμματα ανίχνευσης του Διαδικτύου (web crawlers). Οι web crawlers, οι οποίοι είναι γνωστοί κι ως ants, automatic indexers, bots, web spiders, web robots και web scutters, είναι αυτοματοποιημένα προγράμματα που σαρώνουν με ένα συγκεκριμένο τρόπο ιστοσελίδες προκειμένου να δημιουργήσουν ένα κατάλογο δεδομένων. Υπάρχουν πολλοί web crawlers διαθέσιμοι στο Διαδίκτυο, οι περισσότεροι από τους οποίους είναι Open Source. Οι πιο γνωστοί web crawlers είναι οι scrapy [18], Apache Nutch [19] και Heritrix [20].

Οι web crawlers σαρώνουν html σελίδες και αυτό έχει ως αποτέλεσμα τα αποτελέσματα που επιστρέφουν να μην είναι πάντα εύκολα ευανάγνωστα, να περιέχουν αχρείαστες πληροφορίες και απαιτείται μεγάλη επεξεργασία στα δεδομένα, προκειμένου να εξαχθούν πληροφορίες από αυτά. Ένα παράδειγμα κώδικα html κάνοντας σάρωση δεδομένων στο λογαριασμό της Asus στο twitter φαίνεται παρακάτω στιγμιότυπο (snippet):

```
<div class="js-tweet-text-container">
 <p class="TweetTextSize TweetTextSize--16px js-tweet-text tweet-text" lang="en" data-aria-label-part="0">Can you calculate the widest angle of the
 <a href="/hashtag/ASUS?src=hash" data-query-source="hashtag_click" class="twitter-hashtag pretty-link js-nav" dir="ltr" ><s>#</s><b>ASUS</b></a>
 <a href="/hashtag/TransformerMini?src=hash" data-query-source="hashtag_click" class="twitter-hashtag pretty-link js-nav" dir="ltr" ><s>#</s>
 <b>TransformerMini</b></a>? <a href="/hashtag/SmartHinge?src=hash" data-query-source="hashtag_click" class="twitter-hashtag pretty-link js-nav" dir="ltr" >
 <s>#</s><b>SmartHinge</b></a><a href="https://t.co/cljMaJOHcV" class="twitter-timeline-link u-hidden" data-pre-embedded="true" dir="ltr" >pic.twitter.com/cljMaJOHcV</a>
 </p>
</div>
```

Κώδικας 1: Κομμάτι HTML κώδικα από τη σάρωση της σελίδας της ASUS στο Facebook

Ο πηγαίος κώδικας σε html από το λογαριασμό της Asus αποτελείται από 7883 γραμμές. Τα tweets του χρήστη δεν έχουν κάποιο ξεχωριστό id και είναι δύσκολο να εντοπιστούν μέσα στις χιλιάδες γραμμές κώδικα. Από το παραπάνω snippet δεν μπορούν να εξαχθούν πολλές πληροφορίες. Συγκεκριμένα, οι μόνες δύο πληροφορίες που μας ενδιαφέρουν είναι η πρόταση που χρησιμοποιήθηκε στο tweet ("Can you calculate the widest angle of the") και η χρήση φωτογραφίας, η οποία βέβαια δεν μπορεί να αξιολογηθεί. Οι αντιδράσεις των χρηστών στο συγκεκριμένο tweet φαίνονται σε άλλο snippet κώδικα το οποίο βρίσκεται μετά από 40 γραμμές.

Με τη χρήση του παραπάνω παραδείγματος, γίνεται εύκολα κατανοητό για τον αναγνώστη ότι ένας απλός web crawler δεν είναι κατάλληλος για να εξάγει χρήσιμα δεδομένα από κοινωνικά δίκτυα. Να σημειωθεί ακόμα, ότι ο πηγαίος κώδικας του Facebook είναι ακόμα πιο χαώδης και απαιτεί περισσότερη επεξεργασία για να εξαχθούν χρήσιμα δεδομένα.

Η πολυπλοκότητα της εύρεσης χρήσιμων δεδομένων μέσα από τον κώδικα της html δημιουργεί την ανάγκη για εύρεση άλλου τρόπου συλλογής δεδομένων. Το Facebook [21] και το Twitter [22] προσφέρουν διεπαφές προγραμματισμού εφαρμογών (Application Programming Interface, εν συντομία API). Οι συγκεκριμένες διεπαφές υπάρχουν προκειμένου να είναι εύκολο για τους προγραμματιστές να δημιουργήσουν προγράμματα και εφαρμογές χρησιμοποιώντας ή ανταλλάσσοντας κάποιες συγκεκριμένες πληροφορίες από το κοινωνικό δίκτυο.

Χρησιμοποιώντας το διαθέσιμο API έγραψα ένα πρόγραμμα (script) το οποίο συλλέγει δεδομένα από το Facebook και ένα πρόγραμμα το οποίο συλλέγει δεδομένα από το Twitter. Δεν είναι στα πλαίσια της εργασίας να εξηγηθεί η λειτουργία του API των συγκεκριμένων κοινωνικών δικτύων, για αυτό το λόγο δεν γίνεται εκτενής αναφορά στο τρόπος λειτουργίας του ούτε στο τρόπο διασύνδεσης του script με το αυτό. Επίσης δεν παραθέτω ολόκληρο το πρόγραμμα, αλλά παρουσιάζω σταδιακά τις συναρτήσεις που το αποτελούν για τη καλύτερη κατανόηση του. Ολόκληρο το πρόγραμμα υπάρχει σε διαδικτυακό αποθετήριο (online depository), βρίσκεται στον προσωπικό μου χώρο στο Github (https://github.com/tasoslytos/thesis).

2.1 Λογισμικό Συλλογής Δεδομένων Προερχομένων Από το Facebook

Για το πρόγραμμα συλλογής δεδομένων από το Facebook χρησιμοποίησα 12 διαφορετικές συναρτήσεις. Η δημιουργία τόσων πολλών συναρτήσεων για μία κυρίως ενέργεια, αυτή της συλλογής δεδομένων, έγινε για να δοθεί στον κώδικα δυνατότητα επεκτασιμότητας σε μελλοντικές τροποποιήσεις. Ο κατακερματισμός του προγράμματος σε συναρτήσεις εξυπηρετεί και την εκτέλεση των επιμέρους διεργασιών ανεξάρτητα από την εκτέλεση των υπολοίπων συναρτήσεων.

Η πρώτη συνάρτηση που περιγράφω είναι η *getFacebookPageUrl(page_name, access_token, num_statuses),* η οποία κατασκευάζει το url(Uniform Resource Locator/Ενιαίος Εντοπιστής Πόρων) από τη σελίδα του Facebook που αναζητώ δεδομένα. Η συνάρτηση λαμβάνει τρεις (3) παραμέτρους. Το όνομα της σελίδας από την οποία θέλω να αναζητήσω δεδομένα, τον κωδικό πρόσβασης που δίνεται στους προγραμματιστές που χρησιμοποιούν το Facebook API και έναν ακέραιο αριθμό, ο οποίος αντιπροσωπεύει τον αριθμό των δημοσιεύσεων που επιθυμώ να επεξεργαστώ τα δεδομένα τους. Για αποφυγή λαθών χώρισα το url σε 4 κομμάτια:

- την κοινή βάση του url, η οποία δείχνει το graph API του Facebook
- το όνομα της σελίδας την οποία επεξεργάζομαι,
- τα πεδία τα οποία αναζητώ και
- τις παραμέτρους που θέτω εγώ για τον αριθμό των δημοσιεύσεων που επεξεργάζομαι και το κωδικό πρόσβασης από το API του Facebook

Έπειτα υπάρχει μία προαιρετική τύπωση για έλεγχο, το http ερώτημα μέσω δικής μου συνάρτησης στο url που έθεσα και η επιστροφή των δεδομένων που έχω στην κατοχή μου μετά το επιτυχές http ερώτημα.

```
# returns the desired url, after the necessary construction
def getFacebookPageUrl(page_name, access_token, num_statuses):

    url_base = "https://graph.facebook.com/v2.6"
    node = "/%s/posts" % page_name
    url_fields = "/?fields=message,link,created_time,type,name,id," + \
            "comments.limit(0).summary(true),shares,reactions" + \
            ".limit(0).summary(true)"
    parameters = "&limit=%s&access_token=%s" % (num_statuses, access_token)
    url = url_base + node + url_fields + parameters

    # print the url in order to check it manually if I want
    if debug_variable == True:
        print url
```

```
    # retrieve data
    data = json.loads(requestUntilSucceed(url))

    return data
```

Κώδικας 2: Η συνάρτηση getFacebookPageUrl(page_name, access_token, num_statuses), γυρίζει το προς εξέταση url

Η συνάρτηση που εκτελεί το http ερώτημα είναι η *requestUntilSucceed(url)* η οποία καλείται μέσα από τη συνάρτηση *getFacebookPageUrl(page_name, access_token, num_statuses)*, καθώς και μέσα από άλλες συναρτήσεις όπως θα επισημανθεί παρακάτω.

Για τη δημιουργία της συγκεκριμένης συνάρτησης χρειάστηκε η βιβλιοθήκη της urllib2 [23]. Η συνάρτηση λαμβάνει ως όρισμα το url, μέσω της βιβλιοθήκης γίνεται το request στη συγκεκριμένη σελίδα και έπειτα ο κώδικας μπαίνει σε μία επανάληψη μέχρι το ερώτημα να επιστρέψει κωδικό επιτυχίας (200). Στην συνάρτηση υπάρχει και το απαραίτητο exception για το χειρισμό λαθών.

```
# the function that makes persistent http request
def requestUntilSucceed(url):
    req = urllib2.Request(url)
    success = False
    while success is False:
        try:
            response = urllib2.urlopen(req)
            if response.getcode() == 200:
                success = True
        except Exception, e:
            print e
            time.sleep(5)

            print "Error for URL %s: %s" % (url, datetime.datetime.now())
            print "Retrying."

    return response.read()
```

Κώδικας 3: Η συνάρτηση requestUntilSucceed(url), πραγματοποιεί τη σύνδεση http

Η επόμενη συνάρτηση που θα μελετηθεί είναι η *processFacebookPageData(status, access_token),* είναι η συνάρτηση που επεξεργάζεται τα δεδομένα που επιστρέφει το ερώτημα του http. Λαμβάνει δύο (2) παραμέτρους εισόδου, η πρώτη είναι η δημοσίευση που έχει πραγματοποιηθεί στο Facebook από την υπό εξέταση σελίδα και η δεύτερη είναι ο κωδικός που έχει δοθεί από το API του Facebook. Η λειτουργία της είναι να ελέγχει τα δεδομένα που επέστρεψε η συνάρτηση *requestUntilSucceed(url)* και πραγματοποιεί το πρώτο στάδιο επεξεργασίας των δεδομένων.

Μέσα στη συνάρτηση δημιουργώ τοπικές μεταβλητές έτσι ώστε να φορτώσω σε αυτές τα επιμέρους δεδομένα που επιστρέφει η μη επεξεργασμένη δημοσίευση του Facebook. Η

συνάρτηση ελέγχει την ύπαρξη συγκεκριμένων πεδίων καθώς και τη μορφή τους. Να σημειωθεί ότι δεν είναι απαραίτητη η ύπαρξη όλων των πεδίων σε κάθε δημοσίευση και επίσης το Facebook πρόσθεσε τις αντιδράσεις στις 24 Φεβρουαρίου του 2016, πριν από αυτή την ημερομηνία η μόνη δυνατή αντίδραση ήταν το 'Μου αρέσει' (like). Κατά τη διάρκεια επεξεργασίας των δεδομένων χρησιμοποιούντα δύο ακόμα συναρτήσεις που έχω δημιουργήσει εγώ, η *getNumberTotalReactions(reaction_type, reactions)* και η *getReactions(status_id, access_token)*, τις οποίες αναλύω στη συνέχεια, καθώς και έτοιμες συναρτήσεις που προσφέρονται από βιβλιοθήκες, όπως η *unicodedata, datetime* [24].

```
    # checks if everything goes right
    # some items does not exist necessarily
    def processFacebookPageData(status, access_token):

        status_id = status['id']
        status_message = '' if 'message' not in status.keys() else \
            unicodeNormalize(status['message'])
                        .
                        .
                        .
        status_published = datetime.datetime.strptime(
            status['created_time'],'%Y-%m-%dT%H:%M:%S+0000')
                        .
                        .
                        .
        num_reactions = 0 if 'reactions' not in status else \
            status['reactions']['summary']['total_count']
                        .
                        .
                        .
        num_likes = num_reactions if status_published < '2016-02-24 00:00:00' \
            else num_likes
                        .
                        .
                        .
    num_loves = getNumberTotalReactions('love', reactions)
                        .
                        .
                        .
```

Κώδικας 4: Μέρος της συνάρτησης processFacebookPageData(status, access_token), πραγματοποιεί την πρώτη επεξεργασία δεδομένων. Οι κάθετες τελείες ανάμεσα στις εντολές υποδηλώνουν κώδικα που έχει αφαιρεθεί για λόγους οικονομίας χώρου

Η συνάρτηση *unicodeNormalize(text)*, η οποία χρησιμοποιείται μέσα στη συνάρτηση *processFacebookPageData(status, access_token)*, είναι μία απλή συνάρτηση η οποία λαμβάνει ως παράμετρο ένα αλφαριθμητικό και το κωδικοποιεί σε μορφή UTF-8. Σε κάποιες περιπτώσεις αυτή η αλλαγή δεν είναι απαραίτητη, αλλά πραγματοποιείται σε κάθε δημοσίευση έτσι ώστε να καλύπτονται όλες οι περιπτώσεις.

```
# needed to write tricky unicode correctly to csv
def unicodeNormalize(text):
    return text.translate({ 0x2018:0x27, 0x2019:0x27, 0x201C:0x22, 0x201D:0x22, 0xa0:0x20
}).encode('utf-8')
```

Κώδικας 5: Η συνάρτηση unicodeNormalize(text), μετατρέπει το κείμενο που λαμβάνει στην παράμετρο εισόδου σε κωδικοποίηση UTF-8

Η συνάρτηση *getNumberTotalReactions(reaction_type, reactions)* είναι μία συνάρτηση η οποία ορίζεται μέσα στη *processFacebookPageData(status, access_token)* η οποία λαμβάνει δύο ορίσματα, το πρώτο είναι η συγκεκριμένη αντίδραση που αναζητώ και δεύτερο είναι οι συνολικές αντιδράσεις από τη δημοσίευση που μελετώ.

```
# function that returns the number of each reaction
def getNumberTotalReactions(reaction_type, reactions):
    if reaction_type not in reactions:
        return 0
    else:
        return reactions[reaction_type]['summary']['total_count']
```

Κώδικας 6: Η συνάρτηση getNumberTotalReactions(reaction_type, reactions), επιστρέφει τις συνολικές αντιδράσεις των χρηστών σε μία δημοσίευση

Η τρίτη συνάρτηση που χρησιμοποιώ μέσα στη *processFacebookPageData(status, access_token)* είναι η *getReactions(status_id, access_token)*. Η *getReactions(status_id, access_token)* είναι μία συνάρτηση η οποία λαμβάνει δύο ορίσματα, το αναγνωριστικό της δημοσίευσης και τον κωδικό πρόσβασης από το API του Facebook και επιστρέφει τις αντιδράσεις της δημοσίευσης που μελετάω. Σύμφωνα με το API, για να επιστραφούν οι αντιδράσεις της κάθε κατηγορίας (‘love’, ‘wow’, ‘haha’, ‘sad’, ‘angry’) από μία δημοσίευση πρέπει να ζητηθούν ρητώς από τον προγραμματιστή.

```
# returns the reactions
# must ask for each reaction explicitly
def getReactions(status_id, access_token):

    url_base = "https://graph.facebook.com/v2.6"
    node = "/%s" % status_id
    reactions = "/?fields=" \
          "reactions.type(LIKE).limit(0).summary(total_count).as(like)" \
          ",reactions.type(LOVE).limit(0).summary(total_count).as(love)" \
                        .
                        .
                        .
    parameters = "&access_token=%s" % access_token
    url = url_base + node + reactions + parameters

    # retrieve data
```

```
    data = json.loads(requestUntilSucceed(url))

    return data
```

Κώδικας 7: Μέρος της συνάρτηση getReactions(status_id, access_token), πραγματοποιεί http ερώτημα ρητά για κάθε πιθανή αντίδραση. . Οι κάθετες τελείες ανάμεσα στις εντολές υποδηλώνουν κώδικα που έχει αφαιρεθεί για λόγους οικονομίας χώρου

Η επόμενη συνάρτηση είναι αυτή η οποία χρησιμοποιεί τις συναρτήσεις που έχω παραθέσει νωρίτερα για πραγματοποίηση των http ερωτημάτων και των απαραίτητων επεξεργασιών στα δεδομένα που επιστρέφουν τα ερωτήματα. Η συνάρτηση *storeFacebookInformationDataBase(page_name, access_token)* καλεί τις προαναφερθείσες συναρτήσεις και επίσης αποθηκεύει τα δεδομένα σε βάση δεδομένων. Την παρουσίαση της δημιουργίας της βάσης δεδομένων που χρησιμοποιώ την παραθέτω στο κεφάλαιο που ακολουθεί.

Η συνάρτηση αρχικά καλεί βοηθητική ενσωματωμένη συνάρτηση για μέτρηση χρόνου, εμφανίζει βοηθητικό μήνυμα για τον προγραμματιστή, μηδενίζει τον αριθμητή που θα χρησιμοποιήσουμε και καλεί τη συνάρτηση *getFacebookPageUrl(page_name, access_token, num_statuses)* για x δημοσιεύσεις της σελίδας που ορίσαμε.

Έπειτα η συνάρτηση μπαίνει σε μία λούπα 100 επαναλήψεων, στην οποία καλείται η *processFacebookPageData(status, access_token)*, πραγματοποιείται η σύνδεση με τη βάση δεδομένων καθώς και η ενημέρωση της με τα νέα δεδομένα. Κατά την εκτέλεσης της διαδικασίας που μόλις παρουσίασα υπάρχει μήνυμα προόδου για τον προγραμματιστή και έλεγχος για την πετυχημένη σύνδεση με τη βάση δεδομένων.

Η σύνδεση με τη βάση δεδομένων επιτυγχάνεται μέσω της συνάρτησης *connectDb()*. Η συγκεκριμένη συνάρτηση είναι αυτή που πραγματοποιεί τη σύνδεση με τη βάση δεδομένων που έχω δημιουργήσει, χρησιμοποιώντας ως σύνδεσμο με τη βάση δεδομένων (database connector) τη pymysql [25].

```
# connect to DataBase
def connectDb():
    connection = pymysql.connect(host='localhost',
                                 user='root',
                                 password='',
                                 db='my_database')
    return connection
```

Κώδικας 8: Η συνάρτηση connectDb(), πραγματοποιεί στη σύνδεση με τη βάση δεδομένων

Ακόμη δίνεται η επιλογή στο χρήστη να εκτελεί τη συγκεκριμένη συνάρτηση μέχρι το σημείο όπου η σελίδα προς εξέταση δεν έχει άλλες διαθέσιμες δημοσιεύσεις. Το μόνο που χρειάζεται από το χρήστη είναι να θέσει σε σχόλια την εντολή της while με το συγκεκριμένο αριθμό επαναλήψεων και να αφαιρέσει τα σχόλια από τη while η οποία αναζητάει συνεχώς νέα σελίδα προς εξέταση.

```
# store the information from the Facebook page to my database
def storeFacebookInformationDataBase(page_name, access_token):

    scrape_starttime = datetime.datetime.now()
                        .
                        .
                        .
    statuses = getFacebookPageUrl(page_name, access_token, 100)

    #while has_next_page:
    while num_processed<100:
        for status in statuses['data']:
                        .
                        .
                        .
            status_id_db, status_message_db, link_name_db, status_type_db, status_link_db, status_published_db, num_reactions_db, num_comments_db, num_shares_db, num_likes_db, num_loves_db, num_wows_db, num_hahas_db, num_sads_db, num_angrys_db = processFacebookPageData(status,access_token)
                        .
                        .
                        .
            # SQL statement for adding Facebook data
            insert_info = ("INSERT INTO statuses_pos " "(status_id, status_message, link_name, status_type, status_link, status_published, num_reactions, num_comments, num_shares, num_likes, num_loves, num_wows, num_hahas, num_sads, num_angrys)" "VALUES (%s, %s, %s,%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s)")
                        .
                        .
                        .

            connection = connectDb()
            cursor = connection.cursor()
                        .
                        .
                        .

        # if there is no next page, we're done.
        if 'paging' in statuses.keys():
            statuses = json.loads(requestUntilSucceed(
                                statuses['paging']['next']))
                        .
                        .
                        .
```

Κώδικας 9: Μέρος της συνάρτησης storeFacebookInformationDataBase(page_name, access_token), η βασική συνάρτηση καλεί όλες τις προηγούμενες και αποθηκεύει τα δεδομένα. Οι κάθετες τελείες ανάμεσα στις εντολές υποδηλώνουν κώδικα που έχει αφαιρεθεί για λόγους οικονομίας χώρου

Εφόσον ολοκλήρωσα την επεξήγηση της *storeFacebookInformationDataBase(page_name, access_token)*, το επόμενο βήμα είναι να μελετήσω τη συνάρτηση *storeFacebookInformationCSV(page_name, access_token)*. Είναι η συνάρτηση η οποία έχει ως λειτουργία τη κλήση των προηγούμενων συναρτήσεων για τη δημιουργία http ερωτημάτων και βασικής επεξεργασίας των δεδομένων, αλλά η διαφορά της με την *storeFacebookInformationDataBase(page_name, access_token)* είναι ότι αυτή αποθηκεύει τα δεδομένα σε ένα αρχείο csv και όχι στη βάση δεδομένων που έχω δημιουργήσει.

```
# store the information from the Facebook page to my database
def storeFacebookInformationCSV(page_name, access_token):
   with open('directory\%s_facebook_statuses.csv' % page_name, 'wb') as file:
      w = csv.writer(file)
      w.writerow(["status_id", "status_message", "link_name", "status_type", "status_link",
"status_published", "num_reactions", "num_comments", "num_shares", "num_likes",
"num_loves", "num_wows", "num_hahas", "num_sads", "num_angrys"])
                        .
                        .
                        .
      # call the function getFacebookPageUrl
      statuses = getFacebookPageUrl(page_name, access_token, 100)
      #while has_next_page:
      while num_processed<100:
         for status in statuses['data']:
            # Ensure it is a status with the expected metadata
            if 'reactions' in status:
               w.writerow(processFacebookPageData(status,
                  access_token))
                        .
                        .
                        .
         # if there is no next page, we're done.
         if 'paging' in statuses.keys():
            statuses                    =                    json.loads(requestUntilSucceed(
statuses['paging']['next']))
         else:
            has_next_page = False
      print "\nDone!\n%s Statuses Processed in %s" % \
            (num_processed, datetime.datetime.now() - scrape_starttime)
```

Κώδικας 10: Η συνάρτηση storeFacebookInformationCSV(page_name, access_token), η βασική συνάρτηση καλεί όλες τις προηγούμενες και αποθηκεύει σε αρχείο .csv. Οι κάθετες τελείες ανάμεσα στις εντολές υποδηλώνουν κώδικα που έχει αφαιρεθεί για λόγους οικονομίας χώρου

2.2 Δημιουργία Βάσης Δεδομένων

Εφόσον έχω ολοκληρώσει την επεξήγηση των συναρτήσεων που χρησιμοποιούνται για τη συλλογή δεδομένων από τις σελίδες του Facebook που έχω επιλέξει, σε αυτό το υποκεφάλαιο αναπτύσσω το σκεπτικό για τη δημιουργία της βάσης δεδομένων που χρησιμοποιώ. Σε αυτό το σημείο πρέπει να αναφέρω ότι προτιμώ τα δεδομένα να αποθηκεύονται σε βάσεις δεδομένων MySQL, αντί σε αρχεία .csv για λόγους συμβατότητας.

Για τη βάση δεδομένων που έχω αναπτύξει θα σταθώ σε δύο σημεία: τη δημιουργία της βάσης δεδομένων και τη δημιουργία του πίνακα που χρησιμοποιώ για την αποθήκευση των δεδομένων που έχω συλλέξει. Για τη δημιουργία της βάσης δεδομένων και του πίνακα στον οποίο αποθηκεύω δεδομένων χρησιμοποίησα εντολή της MySQL.

```
mysql> CREATE DATABASE facebook_database;
```

Κώδικας 11: Εντολή δημιουργίας βάσης δεδομένων σε γλώσσα MySQL

Η ίδια λειτουργία μπορεί να γίνει και σε γραφικό περιβάλλον με τη χρήση εργαλείων που προσφέρουν ολοκληρωμένο περιβάλλον για ανάπτυξη βάσης δεδομένων, όπως το XAMPP [26].

Αντίστοιχα η δημιουργία του πίνακα μέσα στη βάση δεδομένων πραγματοποιήθηκε κι αυτή από εντολή MySQL. Ο πίνακας περιέχει 15 πεδία, πέντε (5) είναι αλφαριθμητικά (varchar), ένα πεδίο είναι τύπου ημερομηνίας (date) και τα υπόλοιπα είναι ακέραιης (int) μορφής. Ο χώρος που δεσμεύει κάθε μεταβλητή είναι αρκετά μεγάλος ώστε να μην υπάρξει πρόβλημα έλλειψης χώρου. Ίσως να δεσμεύω περισσότερο χώρο από ότι πραγματικά χρειάζομαι, αλλά το σκεπτικό πίσω από αυτή την επιλογή είναι ότι ο αποθηκευτικός χώρος έχει χαμηλό κόστος.

```
mysql> CREATE TABLE `statuses_pos` (
  `status_id` varchar(111) CHARACTER SET latin1 NOT NULL,
  `status_message` varchar(500) CHARACTER SET latin1 NOT NULL,
  `link_name` varchar(255) COLLATE utf8_unicode_ci NOT NULL,
  `status_type` varchar(255) COLLATE utf8_unicode_ci NOT NULL,
  `status_link` varchar(255) COLLATE utf8_unicode_ci NOT NULL,
  `status_published` date NOT NULL,
  `num_reactions` int(11) NOT NULL,
  `num_comments` int(11) NOT NULL,
  `num_shares` int(11) NOT NULL,
  `num_likes` int(11) NOT NULL,
  `num_loves` int(11) NOT NULL,
  `num_wows` int(11) NOT NULL,
```

```
  `num_hahas` int(11) NOT NULL,
  `num_sads` int(11) NOT NULL,
  `num_angrys` int(11) NOT NULL,
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

Κώδικας 12: Εντολή δημιουργίας πίνακα σε βάση δεδομένων σε γλώσσα MySQL

2.3 Γραμματική Ανάλυση των Δεδομένων

Εφόσον έχω ολοκληρώσει τη λήψη δεδομένων μέσω http ερωτημάτων και την αποθήκευση τους σε βάση δεδομένων που έχω δημιουργήσει, το επόμενο βήμα είναι η εύρεση των μερών του λόγου (ρήμα, ουσιαστικό, επίθετο, επίρρημα κτλ.) από τα οποία αποτελείται η κάθε δημοσίευση. Για τη διεκπεραίωση αυτής της διεργασίας χρησιμοποίησα το εργαλείο Stanford Log-linear Part-Of-Speech Tagger [27].

Το εργαλείο προσθήκης ετικετών ανάλογα με το μέρος του λόγου(Part-Of-Speech Tagger), εν συντομία POS Tagger, είναι ένα λογισμικό το οποίο δέχεται ως είσοδο κείμενο και αναθέτει σε κάθε λέξη του κειμένου το μέρος του λόγου στο οποίο ανήκει. Το συγκεκριμένο λογισμικό θέτει πιο λεπτομερή περιγραφή για το μέρος του λόγου στο οποίο ανήκει κάθε λέξη, όπως ουσιαστικό-πληθυντικό.

Καθώς οι δημοσιεύσεις που μελετάμε είναι στην αγγλική γλώσσα, έτσι και τα μέρη του λόγου ανταποκρίνονται στην αγγλική γραμματική. Παρακάτω φαίνεται η λίστα με τα πιθανά μέρη του λόγου που αναθέτει το λογισμικό:

1. CC Coordinating conjunction
2. CD Cardinal number
3. DT Determiner
4. EX Existential there
5. FW Foreign word
6. IN Preposition or subordinating conjunction
7. JJ Adjective
8. JJR Adjective, comparative
9. JJS Adjective, superlative
10. LS List item marker
11. MD Modal

12. NN Noun, singular or mass
13. NNS Noun, plural
14. NNP Proper noun, singular
15. NNPS Proper noun, plural
16. PDT Predeterminer
17. POS Possessive ending
18. PRP Personal pronoun
19. PRP$ Possessive pronoun
20. RB Adverb
21. RBR Adverb, comparative
22. RBS Adverb, superlative
23. RP Particle
24. SYM Symbol
25. TO
26. UH Interjection
27. VB Verb, base form
28. VBD Verb, past tense
29. VBG Verb, gerund or present participle
30. VBN Verb, past participle
31. VBP Verb, non3rd person singular present
32. VBZ Verb, 3rd person singular present
33. WDT Whdeterminer
34. WP Whpronoun
35. WP$ Possessive whpronoun
36. WRB Whadverb

Τα κεφαλαία γράμματα είναι η συντομογραφία για το κάθε μέρος του λόγου καθώς επίσης τα χρησιμοποιώ ως μέρος ονομασίας της μεταβλητής που συσχετίζεται με αυτό.

Η επόμενη συνάρτηση που θα αναλύσω είναι η *analyzeTextStatus()*. Η *analyzeTextStatus(),* είναι η συνάρτηση που εκτελεί το ερώτημα αναζήτησης στη βάση δεδομένων για την κάθε δημοσίευση που υπάρχει σε αυτή, καλεί το εργαλείο του POS Tagger, με τη βοήθεια άλλων συναρτήσεων αναγνωρίζει το μέρος του λόγου στο οποίο ανήκει και τέλος ανανεώνει τη βάση δεδομένων.

Στις πρώτες γραμμές της συνάρτησης πραγματοποιείται η σύνδεση με τη βάση δεδομένων καθώς και το ερώτημα σε αυτή για το αναγνωριστικό της δημοσίευσης και την ίδια τη δημοσίευση. Μετά υπάρχει εγγραφή της δημοσίευσης σε ένα αρχείο μορφής .txt για την παρακολούθηση της εκτέλεσης του προγράμματος και η κλήση του εργαλείου POS Tagger.

Το POS Tagger είναι ένα εργαλείο γραμμένο σε java και εκτελείται όπως κάθε πρόγραμμα σε java. Μέσα στη συνάρτηση καλώ το εκτελέσιμο της java μέσω υποδιεργασίας (subprocess) και την έξοδο του POS Tagger την οδηγώ σε ένα αρχείο .txt. Έπειτα καλώ τη συνάρτηση *firstProcess()*, η οποία επιστρέφει το πλήθος των μερών του λόγου που υπάρχουν σε κάθε δημοσίευση προς εξέταση.

Τέλος πραγματοποιείται μία ακόμα σύνδεση με τη βάση δεδομένων και ενημέρωση της κάθε δημοσίευσης που βρίσκεται σε αυτή με τα νέα μεταδεδομένα που την αφορούν.

```
# insert data into the stanford tool to find which parts of a speech a status has
def analyzeTextStatus():

    cnx = connectDb()
    cnx2 = connectDb()
    cursor1 = cnx.cursor()
    query = ("SELECT status_id, status_message FROM statuses_pos ")

    cursor1.execute(query)

    for (status_id, status_message) in cursor1:
        with open('POSTagger\processed_output.txt', 'w') as fout:
            fout.write(status_message)

        subprocess.call('stanford-postagger models\wsj-0-18-left3words-distsim.tagger
processed_output_twitter.txt > processed_output_POS_twitter.txt', cwd='POSTagger',
shell=True)
        cc_counter, cd_counter, dt_counter, ex_counter, fw_counter, in_counter, jj_counter,
jjr_counter, jjs_counter, ls_counter, md_counter, nn_counter, nns_counter, nnp_counter,
nnps_counter, pdt_counter, pos_counter, prp_counter, prp6_counter, rb_counter, rbr_counter,
rbs_counter, rp_counter, sym_counter, to_counter, uh_counter, vb_counter, vbd_counter,
vbg_counter, vbn_counter, vbp_counter, vbz_counter, wdt_counter, wp_counter, wp6_counter,
wrb_counter = firstProcess()
                        .
                        .
                        .
        try:
            with cnx2.cursor() as cursor2:
                # create a new record
                # insert the data we pulled into db
                cursor2.execute ("""
                        UPDATE statuses_pos
                        SET    num_cc=%s, num_cd=%s, num_dt=%s, num_ex=%s, num_fw=%s,
num_in=%s, num_jj=%s, num_jjr=%s, num_jjs=%s, num_ls=%s, num_md=%s, num_nn=%s,
num_nns=%s, num_nnp=%s, num_nnps=%s, num_pdt=%s, num_pos=%s, num_prp=%s,
num_prp6=%s, num_rb=%s, num_rbr=%s, num_rbs=%s, num_rp=%s, num_sym=%s,
num_to=%s, num_uh=%s, num_vb=%s, num_vbd=%s, num_vbg=%s, num_vbn=%s,
num_vbp=%s, num_vbz=%s, num_wdt=%s, num_wp=%s, num_wp6=%s, num_wrb=%s
                        WHERE status_id=%s
                    """, (cc_counter, cd_counter, dt_counter, ex_counter, fw_counter, in_counter,
jj_counter, jjr_counter, jjs_counter, ls_counter, md_counter, nn_counter, nns_counter,
nnp_counter, nnps_counter, pdt_counter, pos_counter, prp_counter, prp6_counter, rb_counter,
rbr_counter, rbs_counter, rp_counter, sym_counter, to_counter, uh_counter, vb_counter,
vbd_counter, vbg_counter, vbn_counter, vbp_counter, vbz_counter, wdt_counter, wp_counter,
wp6_counter, wrb_counter, status_id) )
                        .
                        .
                        .
    cursor1.close()
    cursor2.close()
```

```
    cnx.close()
    cnx2.close()
```

Κώδικας 13: Μέρος της συνάρτησης analyzeTextStatus(), πραγματοποιεί γραμματικό έλεγχο στα δεδομένα που έχω συλλέξει. Οι κάθετες τελείες ανάμεσα στις εντολές υποδηλώνουν κώδικα που έχει αφαιρεθεί για λόγους οικονομίας χώρου

Η συνάρτηση *firstProcess()*, που καλείται μέσα από την *analyzeTextStatus()* μέσα στην επανάληψη για την κάθε δημοσίευση, διαβάζει την έξοδο από το POS Tagger και με τη βοήθεια της συνάρτησης *wordInText(word, text)* βρίσκει αν το ζητούμενο μέρος του λόγου υπάρχει στο προς εξέταση κείμενο που βρίσκεται στο αρχείο μορφής .txt. Η λειτουργία της *wordInText(word, text)* αναλύεται στη συνέχεια.

```
# finds the number of each POS the status has
def firstProcess():

    cc_counter = 0
    cd_counter = 0
    dt_counter = 0
                        .
                        .
                        .
    file=open("processed_output_POS.txt","r+")

    for word in file.read().split():
        if wordInText("_cc", word):
            cc_counter = cc_counter + 1
        if wordInText("_cd",word):
            cd_counter = cd_counter + 1
        if wordInText("_dt",word):
            dt_counter = dt_counter + 1
                        .
                        .
                        .
    return cc_counter, cd_counter, dt_counter
                        .
                        .
                        .
    file.close();
```

Κώδικας 14: Μέρος της συνάρτησης firstProcess(), υπολογίζει πλήθος των μερών του λόγου από τα οποία αποτελείται το προς εξέταση κείμενο

Για να γίνει καλύτερα κατανοητό η μορφή του κειμένου την οποία επεξεργάζομαι, παραθέτω μία τυπική έξοδο του POS Tagger παρακάτω.

```
A_DT passenger_NN plane_NN has_VBZ crashed_VBN shortly_RB after_IN take-off_NN from_IN Kyrgyzstan_NNP 's_POS capital_NN ,_, Bishkek_NNP ,_, killing_VBG a_DT large_JJ number_NN of_IN those_DT on_IN board_NN ._.
The_DT head_NN of_IN Kyrgyzstan_NNP 's_POS civil_JJ aviation_NN authority_NN said_VBD that_IN out_IN of_IN about_IN 90_CD passengers_NNS and_CC crew_NN ,_, only_RB about_IN 20_CD people_NNS have_VBP survived_VBN ._.
The_DT Itek_NNP Air_NNP Boeing_NNP 737_CD took_VBD off_RP bound_VBN for_IN Mashhad_NNP ,_, in_IN north-eastern_JJ Iran_NNP ,_, but_CC turned_VBD round_NN some_DT 10_CD minutes_NNS later_RB ._.
```

Έξοδος εκτέλεσης 1: Τυπική έξοδος του POS Tagger σε έγγραφο κειμένου

Η συνάρτηση *wordInText(word, text)* λαμβάνει 2 παραμέτρους, τη λέξη που αναζητείται και το κείμενο στο οποίο αναζητείται. Έπειτα μέσω της ενσωματωμένης συνάρτησης *lower()* μετατρέπω λέξη και κείμενο σε πεζά, έτσι ώστε να αποφύγω αστοχίες. Η τρίτη εντολή πραγματοποιείται μέσω της βιβλιοθήκης re [28] και είναι αυτή που εκτελεί την αναζήτηση της λέξης στο κείμενο.

```
# finds a specific word in a given text
def wordInText(word, text):
    word = word.lower()
    text = text.lower()
    match = re.search(word, text)
    if match:
        return True
    return False
```

Κώδικας 15: Η συνάρτηση wordInText(word, text), ελέγχει αν μία συγκεκριμένη λέξη υπάρχει στο κείμενο εισόδου

Σε αυτό το σημείο αξίζει να γίνει μια μικρή ανακεφαλαίωση των διεργασιών που έχουν αναλυθεί και της λειτουργίας που λογισμικού. Το λογισμικό κατασκευάζει το url που αντιστοιχεί στην προς εξέταση σελίδα, γίνεται το http ερώτημα, ακολουθεί ο έλεγχος των δεδομένων που επιστρέφει το ερώτημα, εξάγει τα ζητούμενα δεδομένα, γίνεται η σύνδεση και η ανανέωση με τη βάση δεδομένων, καλείται το POS Tagger και ανανεώνεται ξανά η βάση δεδομένων. Επίσης υπάρχει και μια εναλλακτική συνάρτηση για εγγραφή των δεδομένων σε .csv αρχείο αντί για βάση δεδομένων MySQL.

2.4 Λογισμικό Συλλογής Δεδομένων Προερχομένων Από το Twitter

Το πρόγραμμα συλλογής δεδομένων από το twitter είναι παρόμοιο με το πρόγραμμα συλλογής δεδομένων από το Facebook. Υπάρχουν βέβαια διαφορετικές προσεγγίσεις στην επικοινωνία με το κοινωνικό δίκτυο ανάλογα με το API που χρησιμοποιείται, αλλά η μεθοδολογία είναι ίδια.

Το πρόγραμμα συλλογής δεδομένων για το twitter είναι πιο απλό και χρησιμοποιεί λιγότερες συναρτήσεις, 5 έναντι 12 που χρησιμοποιήθηκαν για τη συλλογή δεδομένων από το Facebook. Αυτό συμβαίνει εξαιτίας του πιο απλού API που διαθέτει το twitter.

Για αυτό το λόγο και για λόγους οικονομίας χώρου δεν πραγματοποιώ εξίσου λεπτομερή ανάλυση για τον τρόπο συλλογής των δεδομένων που προέρχονται από τους λογαριασμού του Twitter.

Η συνάρτηση η οποία συλλέγει δεδομένα από το twitter και τα εισάγει στη βάση δεδομένων που έχω δημιουργήσει για το twitter είναι η *fetchAndStore()*. Η συγκεκριμένη συνάρτηση αρχικά απαιτεί τα στοιχεία του προγραμματιστή για την διεκπεραίωση της σύνδεσης με το API του Twitter. Ο τρόπος σύνδεσης με τα στοιχεία μου ως προγραμματιστής με άδεια από το Twitter αναλύεται στη συνέχεια.

Στη συνέχεια δίνεται το όνομα του λογαριασμού του οποίου θέλω να συλλέξω τις δημοσιεύσεις του και εκτελείται το http ερώτημα για συλλογή δεδομένων μέσω της ενσωματωμένης συνάρτησης *user_timeline(screen_name = user, count = x)*. Λόγω της ύπαρξης αυτό του συγκεκριμένου ερωτήματος δεν υφίσταται λόγος ύπαρξης της συνάρτησης που πραγματοποιεί επίμονη σύνδεση, όπως είχα στην περίπτωση της συλλογής των δεδομένων του Facebook.

Έπειτα υπάρχει μία επανάληψη για κάθε στοιχείο που έχουμε ανακτήσει μέσω του twitter API και τα εισάγουμε στη βάση δεδομένων μου μέσω MySQL ερωτημάτων. Η σύνδεση με τη βάση δεδομένων γίνεται μέσω της συνάρτησης *connectDb()*, η οποία είναι ίδια με τη συνάρτηση *connectDb()*, που χρησιμοποιούμε για τα δεδομένα του Facebook.

Η βάση δεδομένων στην οποία αποθηκεύω τα δεδομένα που έχω συλλέξει από το Twitter είναι διαφορετική από αυτή που αποθηκεύω τα δεδομένα που έχω συλλέξει από το Facebook, αλλά έχει δημιουργηθεί με πανομοιότυπο τρόπο.

Όσον αφορά τα πιστοποιητικά (credentials) που απαιτεί η διασύνδεση του λογισμικού με το API του Twitter, τα παραθέτω σε διαφορετικό αρχείο. Η ενσωμάτωση τους γίνεται μέσω της εντολής *execfile("configuration_file.py", config)*, η οποία εκτελεί το αρχείο που λαμβάνει ως παράμετρο. Στο αρχείο που έχω όλες τις συναρτήσεις που χρειάζονται για τη συλλογή, την επεξεργασία και την αποθήκευση των δεδομένων έχω και τις παρακάτω γραμμές κώδικα:

```
# load API credentials, using external file
# using external file in favor of abstractness
```

```
config = {}
execfile("config.py", config)
```

Κώδικας 16: Οι απαραίτητες εντολές για ενσωμάτωση του αρχείου που διαθέτει τα credentials για τη διασύνδεση με το API του Twitter

Αντίστοιχα στο αρχείο στο αρχείο *config.py* παραθέτω τα credentials που μου έχουν δοθεί για να πραγματοποιηθεί η διασύνδεση μου με το API του Twitter. Για τη διασύνδεση απαιτούνται δύο κλειδιά και δύο κωδικοί, ένας για το κάθε κλειδί. Για λόγους προσωπικών δεδομένων έχω καλύψει τα credentials που μου έχουν δοθεί.

```
# essential data for the credentials of the programmer
# given from the twitter API
consumer_key = "**********"
consumer_secret = "**********"
access_key = "**********"
access_secret = "**********"
```

Κώδικας 17: Τα απαραίτητα credentials για τη διασύνδεση με το API του Twitter

```
# collect data from twitter and insert them into database
def fetchAndStore():

    # create twitter API object
    # using the external file
    twitter = Twitter(
            auth = OAuth(config["access_key"], config["access_secret"], config["consumer_key"],
config["consumer_secret"]))

    # this is the page we're going to query.
    user = "asus"

    # query the user timeline.
    results = twitter.statuses.user_timeline(screen_name = user, count = 10)

    # loop through each status item
    # insert to database
    for status in results:
                                .
                                .
                                .
        # SQL statement for adding Twitter data
        insert_info = ("INSERT INTO tweet_data " "(tweet_id, created_at, tweet_text,
retweet_count, favorite_count)" "VALUES (%s, %s, %s, %s,%s)")
                                .
                                .
                                .
        try:
            with connection.cursor() as cursor:
                # Create a new record
                # insert the data we pulled into db
                cursor.execute(insert_info, page_data)
                                .
                                .
                                .
    cursor.close()
    connection.commit()
```

Κώδικας 18: Μέρος της συνάρτησης fetchAndStore(), συλλέγει δεδομένα από το Twitter και τα αποθηκεύει σε βάση δεδομένων. Οι κάθετες τελείες ανάμεσα στις εντολές υποδηλώνουν κώδικα που έχει αφαιρεθεί για λόγους οικονομίας χώρου

Η δεύτερη συνάρτηση που αναλύω είναι η *fetchAndUpdate()*. Η συγκεκριμένη συνάρτηση ανακτά δεδομένα από τη βάση δεδομένων, πραγματοποιεί γραμματικό έλεγχο σε αυτά και ανανεώνει ξανά τη βάση δεδομένων.

Στη συνάρτηση πραγματοποιείται η σύνδεση με τη βάση μέσω της *connectDb()*, η οποία είναι ίδια με αυτή που έχω παραθέσει στο κεφάλαιο που περιγράφω τη συλλογή δεδομένων από το Facebook.

Στη συνέχει για κάθε εγγραφή της βάσης πραγματοποιώ γραμματική ανάλυση μέσω του POS Tagger και καταγράφω το πλήθος των μερών του λόγου που αποτελούν το κάθε tweet, μέσω της συνάρτηση *firstProcess()*. Η συνάρτηση *firstProcess()* είναι ίδια με αυτή που έχω ήδη παραθέσει στο προηγούμενο κεφάλαιο.

Εφόσον έχει ολοκληρωθεί η γραμματική ανάλυση γίνεται η ανανέωση στη βάση δεδομένων. Η βάση δεδομένων διαθέτει πλέον και το πλήθος των μερών του λόγου που αποτελούν το κάθε tweet.

```
# fetch data from database
# use POSTagger tool
def fetchAndUpdate():
                        .
                        .
                        .
   for (tweet_id, tweet_text) in cursor1:
      with open('POSTagger\processed_output_twitter.txt', 'w') as fout:
         fout.write(tweet_text)

      subprocess.call('stanford-postagger models\wsj-0-18-left3words-distsim.tagger
processed_output_twitter.txt > processed_output_POS_twitter.txt', cwd='POSTagger',
shell=True)
      cc_counter, cd_counter, dt_counter, ex_counter, fw_counter, in_counter, jj_counter,
jjr_counter, jjs_counter, ls_counter, md_counter, nn_counter, nns_counter, nnp_counter,
nnps_counter, pdt_counter, pos_counter, prp_counter, prp6_counter, rb_counter, rbr_counter,
rbs_counter, rp_counter, sym_counter, to_counter, uh_counter, vb_counter, vbd_counter,
vbg_counter, vbn_counter, vbp_counter, vbz_counter, wdt_counter, wp_counter, wp6_counter,
wrb_counter = firstProcess()
                        .
                        .
                        .
      try:
         with cnx2.cursor() as cursor2:
            # Create a new record
            #insert the data we pulled into db
            cursor2.execute ("""
                  UPDATE tweet_data
                  SET  num_cc=%s, num_cd=%s, num_dt=%s,
                     .
                     .
                     .
                  WHERE tweet_id=%s
               """, (cc_counter, cd_counter, dt_counter
                     .
                     .
                     .
```

Κώδικας 19 : Μέρος της συνάρτησης fetchAndUpdate(), επεξεργάζεται τα δεδομένα από το Twitter καιανανεώνει τη βάση δεδομένων. Οι κάθετες τελείες ανάμεσα στις εντολές υποδηλώνουν κώδικα που έχει αφαιρεθεί για λόγους οικονομίας χώρου

3. Οπτικοποίηση Δεδομένων

Σε αυτό το κεφάλαιο θα πραγματοποιηθεί η πρώτη απόπειρα ερμηνείας και κατανόησης των δεδομένων που έχω συλλέξει από τα κοινωνικά δίκτυα. Με τον όρο οπτικοποίηση (visualization) εννοούμε την ανάπτυξη και τη χρήση οπτικών μέσων ώστε να καταστήσουμε πιο κατανοητό ένα θέμα. Με αυτό τον τρόπο μας δίνεται η δυνατότητα να επεξεργαστούμε αριθμητικά δεδομένα και να τα μετατρέψουμε σε εικόνες δύο ή τριών διαστάσεων.

Εκτός τη βοήθειας που προσφέρει η οπτικοποίηση των δεδομένων στην ερμηνεία και στην κατανόηση, προβάλλονται ακόμα νέα μη αναμενόμενα δεδομένα τα οποία προσφέρουν νέα οπτική στον αναγνώστη. Ακόμη, σε πολύπλοκα προβλήματα, όπως αυτά που αντιμετωπίζω σε αυτή την εργασία, όπου τα δεδομένα περιπλέκονται μεταξύ τους, δίνεται η δυνατότητα διερεύνησης των σχέσεων και αλληλεξαρτήσεων που υπάρχουν ανάμεσα στα δεδομένα.

Για την οπτικοποίηση των δεδομένων και την περαιτέρω εξεργασίας τους μέσω τεχνικών μηχανικής μάθησης χρησιμοποίησα την πλατφόρμα Anaconda [29], μία πλατφόρμα ανοικτού λογισμικού γραμμένη σε Python [30] [31]. Η πλατφόρμα έχει δημιουργηθεί για επεξεργασία δεδομένων(data science) και περιλαμβάνει πολλές σχετικές βιβλιοθήκες(pandas [32], matplotlib [33] κ.α.).

3.1 Οπτικοποίηση Δεδομένων Προερχομένων Από το Facebook

Σε αυτή το υποκεφάλαιο εξετάζονται οι σχέσεις μεταξύ των πεδίων που υπάρχουν σε κάθε εγγραφή στη βάση δεδομένων που έχω δημιουργήσει. Για τη μελέτη της σχέσης

μεταξύ των πεδίων της βάσης δημιουργώ διαγράμματα, από τα οποία εξάγονται σημαντικές πληροφορίες για τα δεδομένα που έχω αποθηκεύσει.

Για την καλύτερη κατανόηση των δεδομένων που μελετάω χωρίζω την οπτικοποίηση αυτών σε τρία (3) σκέλη. Στο πρώτο μελετάω τις αντιδράσεις των χρηστών ανά δημοσίευσης, για να κατανοήσω πως οι χρήστες του Facebook χρησιμοποιούν τις διαθέσιμες αντιδράσεις. Στο δεύτερο μέρος της ανάλυσης ασχολούμαι με το γραμματικό μέρος που διαθέτουν οι εγγραφές της βάσης δεδομένων, έτσι γίνεται κατανοητή η επίδραση των διαφορετικών μερών του λόγου στους χρήστες του κοινωνικού δικτύου. Στο τελευταίο κομμάτι για την ολοκλήρωση της διαδικασίας της οπτικοποίησης των δεδομένων μελετάω την επιρροή που έχουν στους χρήστες τα διαφορετικά είδη δημοσίευσης (κείμενο, εικόνα, βίντεο).

3.1.1 Αντιδράσεις Χρηστών ανά Δημοσίευση

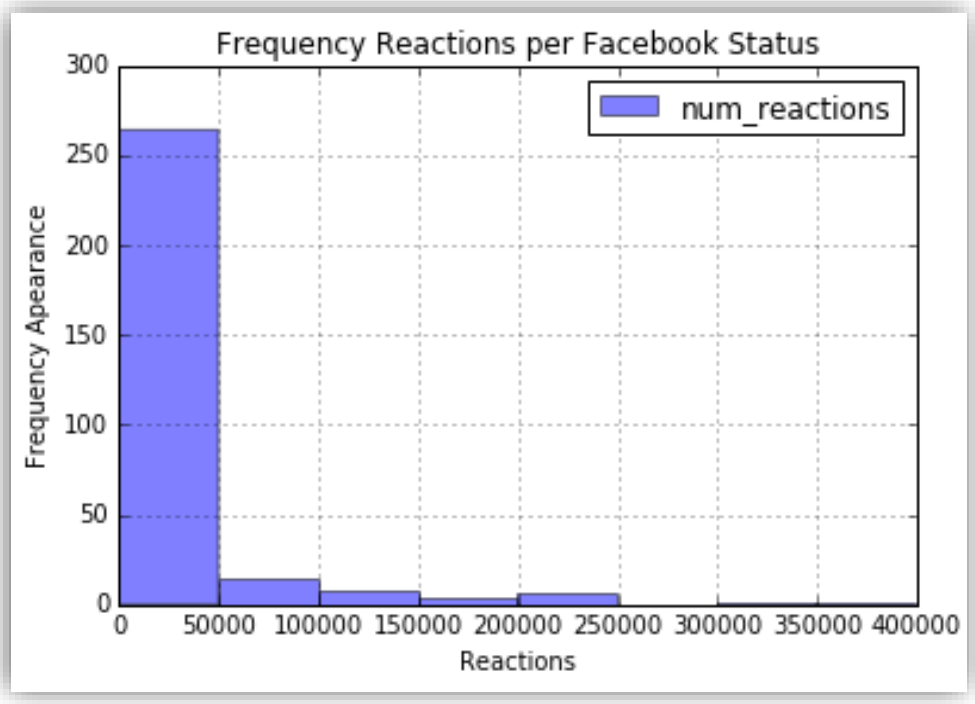
Το επόμενο βήμα στην οπτικοποίηση των δεδομένων είναι να μελετήσω τις αντιδράσεις των χρηστών ανά δημοσίευση. Να υπενθυμίσω ότι στο Facebook δίνεται η δυνατότητα στο χρήση να σχολιάσει μία δημοσίευση, να την κοινοποιήσει και να αντιδράσει σε αυτή με 6 διαφορετικούς τρόπους. Παρακάτω φαίνονται τιμές ανά δημοσίευση για κάθε αντίδραση.

Είδος αντίδρασης	**Συχνότητα/δημοσίευση**
reactions	18399.855705
comment	65.815436
share	118.090604
like	18238.278523
love	72.932886
wow	75.298658
haha	10.392617
sad	1.496644
angry	1.463087

Πίνακας 1: Είδη αντίδρασης και συχνότητα αντίδρασης ανά δημοσίευση

Ο αριθμός των reactions είναι το άθροισμα των ‘like’, ‘love’, ‘wow’, ‘haha’, ‘sad’, ‘angry’. Τα ‘like’ είναι η ισχυρή πλειοψηφία στις αντιδράσεις των χρηστών, συγκεκριμένα

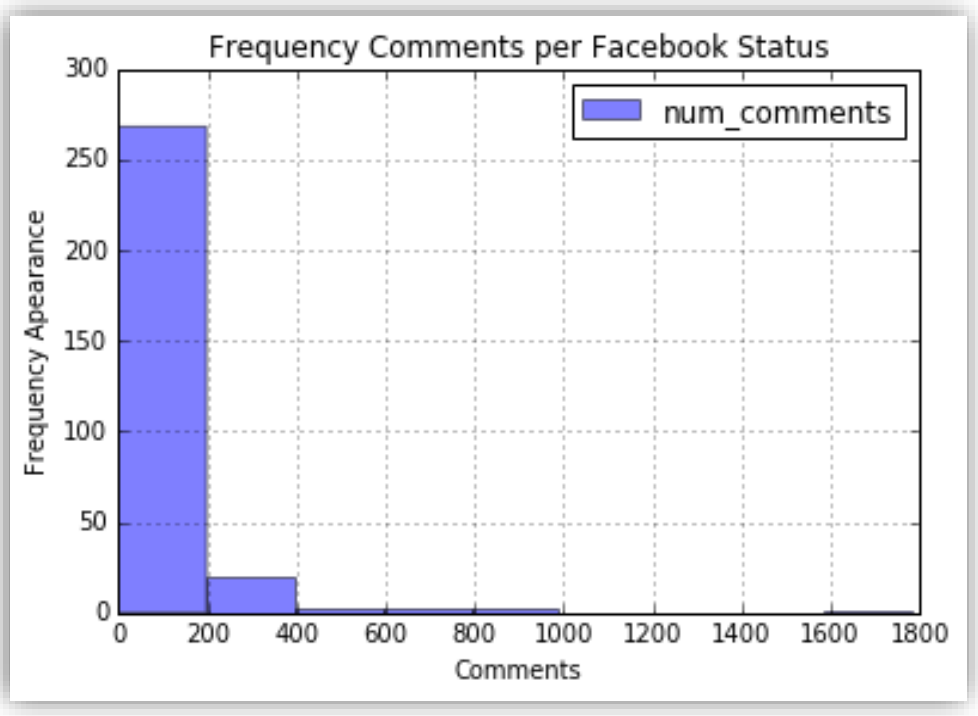
αποτελούν το 99% των αντιδράσεων. Για αυτό το λόγο σε κάποιες περιπτώσεις θα χρησιμοποιώ τη συνολική μεταβλητή και σε κάποιες άλλες, όταν μελετάω τη σχέση συγκεκριμένης αντιδράσεις, τις μεμονωμένες μεταβλητές. Τα διάγραμμα συχνότητας των αντιδράσεων, φαίνεται παρακάτω.



Διάγραμμα 1: Συχνότητα εμφάνισης αντιδράσεων

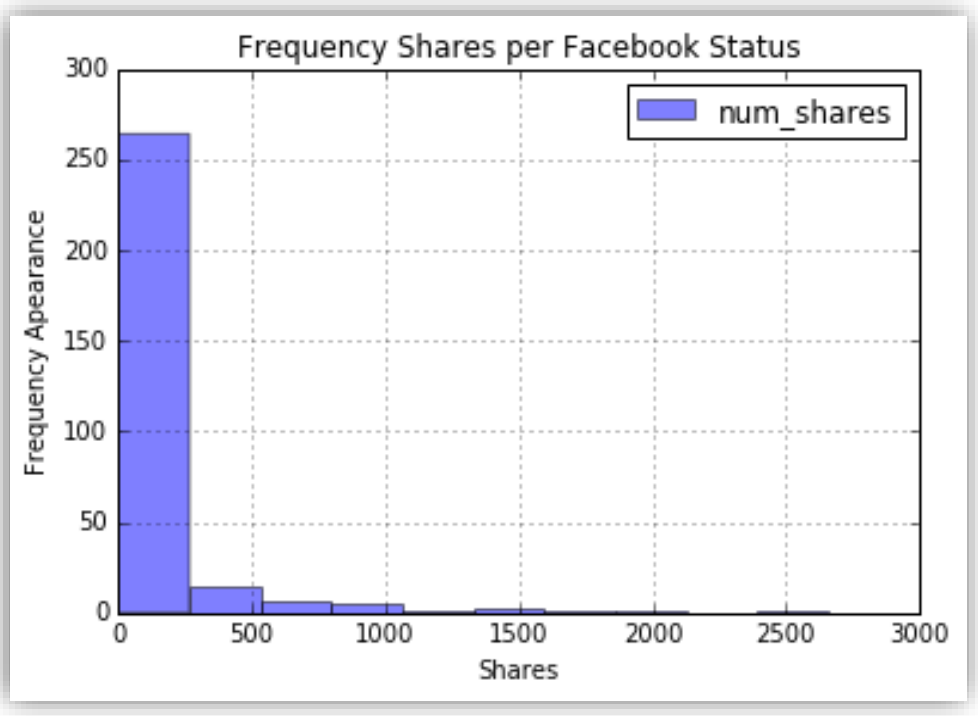
Από το παραπάνω διάγραμμα διακρίνεται ότι οι συνολικές αντιδράσεις είναι λιγότερες από 5000 στο 85% των περιπτώσεων. Στην παραπάνω περίπτωση όπου έχουμε πολύ μεγάλη συγκέντρωση σε μία τιμή, δεν είναι δυνατή η εύρεση κάποιο μοτίβου στα δεδομένα μας.

Ακολουθούν τα διαγράμματα συχνότητας σχολίων ανά δημοσίευση και συχνότητας κοινοποίησης ανά δημοσίευση. Η κατανομή στις δύο περιπτώσεις είναι παρόμοια με την κατανομή στο διάγραμμα που προηγήθηκε. Συγκέντρωση μεγαλύτερη του 80% στις μικρές τιμές και κατανομή που σταδιακά φθίνει καθώς πληθαίνουν τα σχόλια/κοινοποιήσεις.



Διάγραμμα 2: Συχνότητα εμφάνισης σχολίων

Στο διάγραμμα των σχολίων πρέπει να παρατηρήσετε τις μονάδες στον άξονα x΄x. Τα σχόλια είναι εμφανώς λιγότερα από τις αντιδράσεις ανά δημοσίευση. Πιο συγκεκριμένα, στο 85% των περιπτώσεων παρατηρείται να έχουμε λιγότερα από 200 σχόλια και λιγότερες από 50000 αντιδράσεις.



Διάγραμμα 3: Συχνότητα εμφάνισης αναδημοσιεύσεων

Στο διάγραμμα των κοινοποιήσεων η κατανομή είναι παρόμοια με τις δύο περιπτώσεις που μόλις ανέφερα, με το 85% των δημοσιεύσεων να έχουν λιγότερες από 200 κοινοποιήσεις.

Από τα τρία διαγράμματα που έχω παρουσιάσει, καθώς επίσης και από τον πίνακα με τα είδη των αντιδράσεων και τη συχνότητα εμφάνισης τους, συμπεραίνεται ότι αντιδράσεις, σχόλια και κοινοποίησες εμφανίζουν το ίδιο μοντέλο στη συχνότητα εμφάνισης τους. Υπάρχει πληθώρα δημοσιεύσεων από τις σελίδες που μελετάω οι οποίες φαίνονται αδιάφορες στους χρήστες του κοινωνικού δικτύου που ακολουθούν τη συγκεκριμένη σελίδα.

3.1.2 Κατανομή των Διαφορετικών Μερών του Λόγου

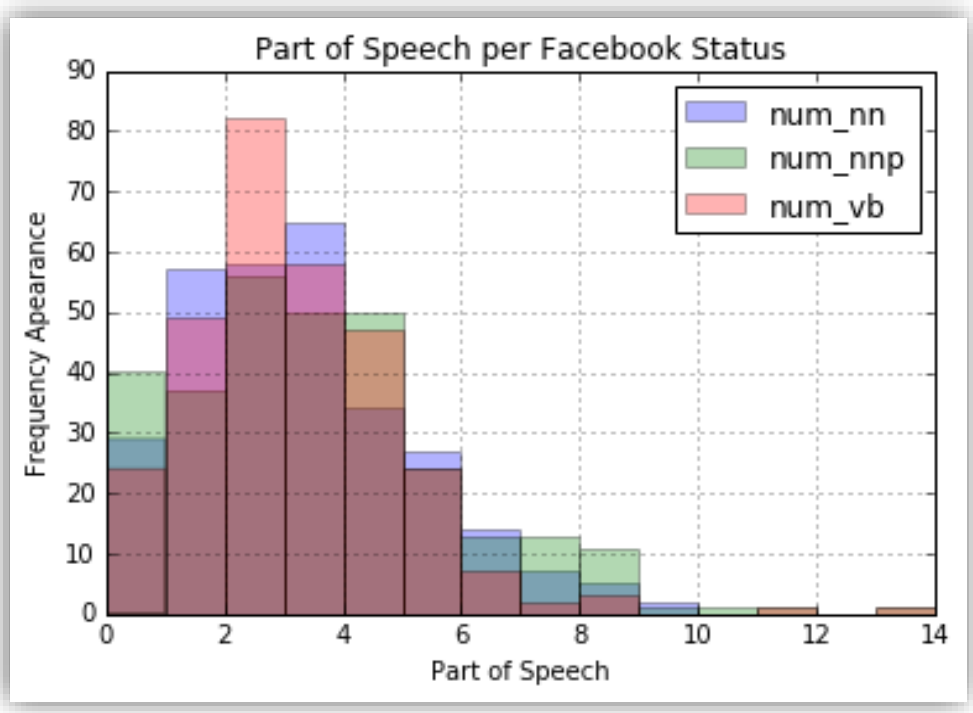
Εφόσον ολοκλήρωσα την οπτικοποίηση των αντιδράσεων θα προχωρήσω στην κατανομή των διαφορετικών μερών του λόγου. Όπως έχει ήδη αναφερθεί το εργαλείο του POSTagger προσφέρει 36 διαφορετικά μέρη του λόγου. Η συχνότητα εμφάνισης των διαφορετικών μερών του λόγου ανά δημοσίευση φαίνεται στον πίνακα που ακολουθεί.

Μέρος του Λόγου	**Συχνότητα/δημοσίευση**
num_nnp	3.087248
num_nn	2.785235
num_vb	2.681208
num_dt	1.379195
num_in	1.738255
num_jj	1.278523
num_prp	1.422819
num_nns	0.828859
num_cd	0.644295
num_rb	0.620805
num_vbz	0.604027
num_cc	0.476510
num_to	0.422819
num_vbp	0.395973
num_vbg	0.278523
num_md	0.208054
num_vbd	0.238255
num_vbn	0.234899
num_wp	0.151007
num_wrb	0.147651
num_rp	0.137584
num_jjr	0.124161
num_pos	0.104027
num_jjs	0.093960
num_wdt	0.063758
num_rbr	0.046980
num_nnps	0.040268
num_fw	0.030201
num_rbs	0.030201
num_uh	0.020134
num_sym	0.016779
num_ex	0.013423
num_pdt	0.013423
num_ls	0.003356
num_wp6	0.000000

num_prp6	0.000000

Πίνακας 2: Συχνότητα εμφάνισης μερών του λόγου ανά δημοσίευση

Όπως φαίνεται από τον παραπάνω πίνακα, μόλις 7 μέρη του λόγου εμφανίζονται, κατά μέσο όρο, σε κάθε δημοσίευση και δύο δεν εμφανίστηκαν ποτέ στο δείγμα που έχω στην κατοχή μου. Παρακάτω παραθέτω το ιστόγραμμα που εμφανίζει τη κατανομή των τριών μερών του λόγου με τη μεγαλύτερη συχνότητα(nnp – κύρια ονόματα, vb - ρήμα σε βασική μορφή και nn – ουσιαστικό σε ενικό αριθμό).



Διάγραμμα 4: Συχνότητα εμφάνισης των τριών πιο συχνά χρησιμοποιημένων μερών του λόγου

Οι κατανομές των τριών πιο συχνών μερών του λόγου παρουσιάζουν κάποιες ομοιότητες και κάποιες διαφορές. Αρχικά η μεγάλη συγκέντρωση των τιμών είναι στο εύρος από 1 έως 5 και η έντονη μείωση μετά τη τιμή 6 διακρίνονται σε όλα τα μέρη του λόγου.

Τα ρήματα παρουσιάζουν μία κατανομή η οποία μπορεί να χαρακτηριστεί κανονική με μέση τιμή ίση με 3, ενώ δεν υπάρχουν δημοσιεύσεις με 9 και 10 ρήματα στο δείγμα μας. Τα ουσιαστικά σε ενικό αριθμό παρουσιάζουν συγκέντρωση στο εύρος 2-4, συγκεκριμένα αυτό το υποσύνολο αποτελεί το 60% των εμφανίσεων ουσιαστικών στο δείγμα μας. Τα κύρια ονόματα, παρουσιάζουν την μεγαλύτερη συχνότητα εμφάνισης και την πιο ακανόνιστη κατανομή. Όπως και στα απλά ουσιαστικά, έτσι και στα κύρια ονόματα, δεν υπάρχει μία τιμή με εντυπωσιακά υψηλή συγκέντρωση, αλλά τρεις τιμές (3-5) όπου παρουσιάζουν μεγάλη συγκέντρωση. Το δεύτερο σημείο που αξίζει να σημειωθεί είναι ότι στις τιμές συχνότητας μικρότερες του 1 και μεγαλύτερες του 7 τα κύρια ονόματα έχουν τη μεγαλύτερη συγκέντρωση σε σχέση με τα άλλα δύο μέρη του λόγου.

Η εξήγηση που δίνω για την ιδιαίτερη κατανομή των κυρίων ονομάτων είναι ότι παρακολουθώ δημοσιεύσεις τεχνολογικών εταιρειών, οι οποίες τείνουν να αναφέρουν το όνομα της εταιρείας τους και το προϊόν το οποίο προβάλλουν σε κάθε δημοσίευση. Ακόμη πιθανά *#hashtags* που χρησιμοποιούνται κατηγοριοποιούνται ως κύρια ονόματα.

Η δημιουργία ιστογράμματος με συμμετοχή περισσοτέρων μερών του λόγου δεν μπορεί να δώσει κάποια πληροφορία, αφού οι διαφορετικές κατανομές μπερδεύονται μεταξύ τους, πραγματοποιώντας αδύνατη την εξαγωγή κάποιου μοτίβου.

3.1.3 Επίδραση Διαφορετικών Ειδών Δημοσίευσης

Εφόσον ολοκλήρωσα την οπτική απεικόνιση των διαφόρων μερών του λόγου που υπάρχει στο δείγμα μου, θα προχωρήσω στην εύρεση συσχετίσεων μεταξύ των διαφορετικών πεδίων.

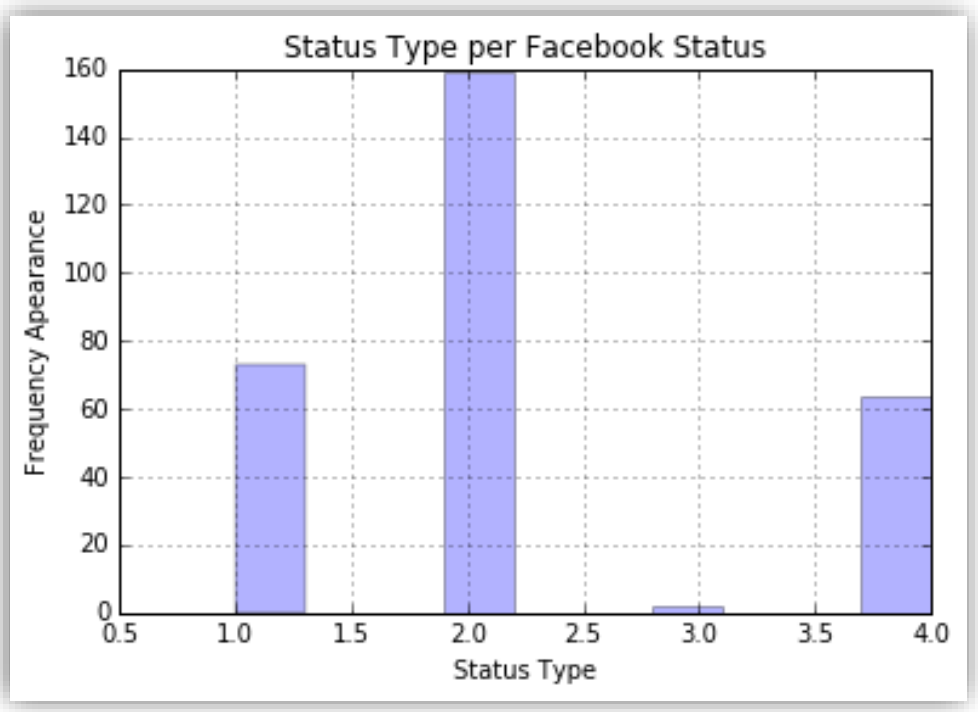
Η πρώτη μελέτη σχέσεων που μελετάω είναι η συχνότητα εμφάνισης των διαφορετικών τύπων δημοσίευσης που προσφέρονται από το Facebook, απλό κείμενο, φωτογραφία, βίντεο και υπερσύνδεσμος (hyperlink).

Για την αλλαγή του τύπου της δημοσίευσης από αλφαριθμητικό σε ακέραιο, επειδή η δημιουργία ιστογράμματος απαιτεί χρήση δύο ακεραίων μεταβλητών, χρησιμοποίησα τη συνάρτηση *map,* η κλήση της συνάρτησης και η αντιστοίχιση με ακέραιο φαίνεται στο παρακάτω στιγμιότυπο.

```
X.status_type = X.status_type.map({'link':1, 'photo':2, 'status':3, 'video':4 })
```

Κώδικας 20: Η συνάρτηση μετατροπής αλφαριθμητικών σε ακέραιους αριθμούς

Στο διάγραμμα που ακολουθεί παρουσιάζω το συνολικό πλήθος εμφανίσεων των δημοσιεύσεων που έχω συλλέξει στη βάση δεδομένων μου.

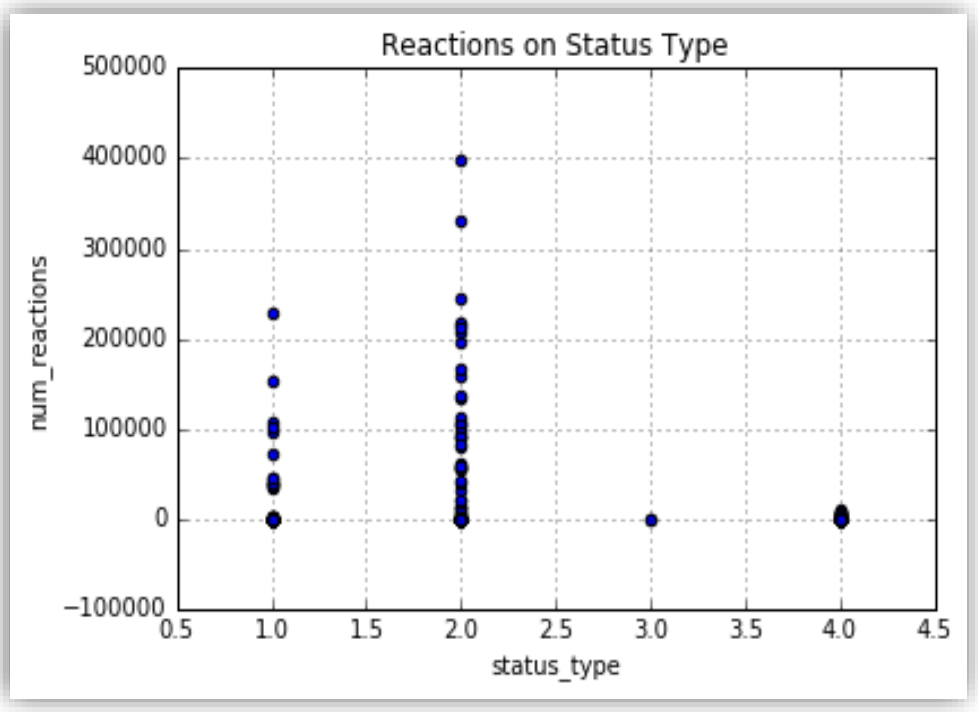


Διάγραμμα 5: Συχνότητα χρήσης διαφορετικών ειδών δημοσίευσης

Οι δύο σημαντικότερες παρατηρήσεις είναι ότι οι δημοσιεύσεις απλού κειμένου σπανίζουν και ότι οι δημοσιεύσεις που περιέχουν φωτογραφία αποτελούν το 50% των συνολικών δημοσιεύσεων. Το υπόλοιπο 50% μοιράζεται σε δημοσιεύσεις που περιέχουν υπέρ-συνδέσμους και βίντεο, με μια ελαφριά υπεροχή των δημοσιεύσεων με υπέρ-συνδέσμους. Πρέπει να σημειώσω ότι μία δημοσίευση η οποία διαθέτει φωτογραφία ή βίντεο ή υπερσύνδεσμο διαθέτει και περιγραφή απλού κειμένου.

Συνεχίζω τη μελέτη μου στη συμπεριφορά των χρηστών ανάλογα με το είδος της δημοσίευσης. Ο λόγος της περαιτέρω εξέτασης μου ανάλογα με το είδος της δημοσίευσης προκύπτει από τη φύση του πεδίου, που έχει μόλις 4 διακριτές τιμές. Δεν υπάρχει άλλο πεδίο στα δεδομένα μου με τόσο περιορισμένο και διακριτό σύνολο τιμών. Εκτός αυτού από την εμπειρία μου ως χρήστης κοινωνικών δικτύων, μπορώ να καταλάβω την αμεσότητα και επιρροή μίας φωτογραφίας έναντι μίας δημοσίευσης χωρίς φωτογραφία. Αυτή τη δύναμη της φωτογραφίας αξιοποιούν και οι διαφορετικές σελίδες προκειμένου να προσελκύσουν ακόλουθους και πιθανούς αγοραστές.

Η επόμενη σχέση που μελετάω είναι η συσχέτιση των αντιδράσεων των χρηστών ανάλογα με το είδος της δημοσίευσης. Σε αυτή την περίπτωση θεωρώ όλες τις αντιδράσεις ως μία, επειδή οι αντιδράσεις πλην του ‘like’ αποτελούν μόλις το 1% των συνολικών αντιδράσεων, αυτό έχει ως αποτέλεσμα ο απόλυτος αριθμός των αντιδράσεων να είναι πολύ μικρός και να μην γίνεται να εξαχθεί κάποιο συμπέρασμα από την κατανομή τους.

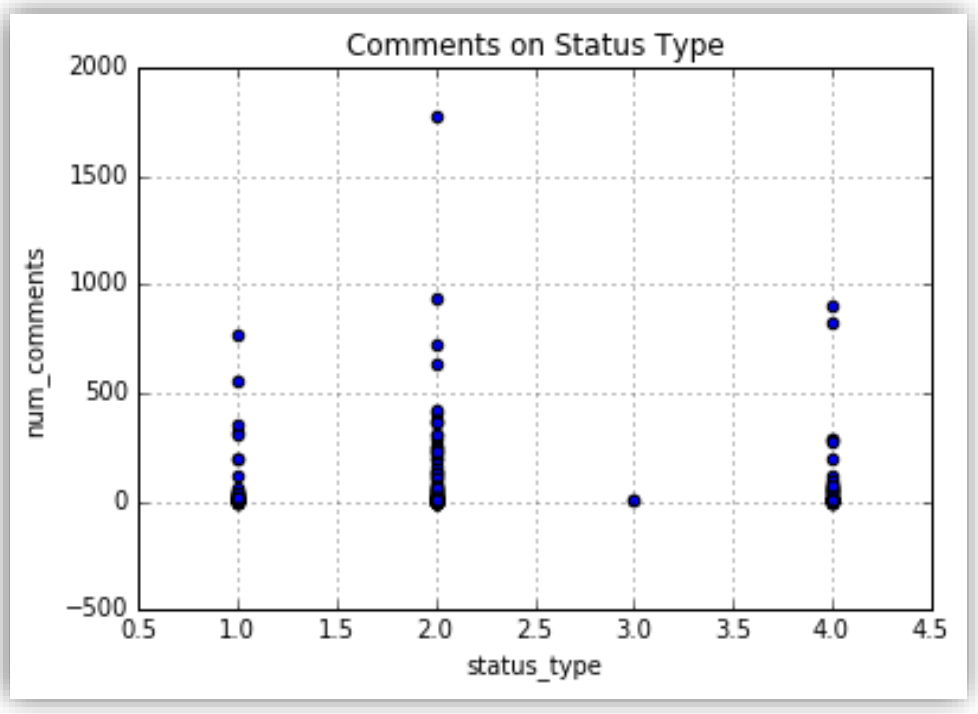


Διάγραμμα 6: Αντιδράσεις χρηστών ανάλογα με το είδος της δημοσίευσης

Από το διάγραμμα αντιδράσεις χρηστών – είδος δημοσίευσης προκαλεί εντύπωση οι αισθητά λιγότερες αντιδράσεις των χρηστών σε δημοσιεύσεις βίντεο. Να υπενθυμίσω ότι οι δημοσιεύσεις που περιλαμβάνουν βίντεο έχουν το ίδιο πλήθος εμφανίσεων με τις δημοσιεύσεις που περιλαμβάνουν υπερσύνδεσμο.

Οι δημοσιεύσεις που περιλαμβάνουν βίντεο εκτός του ότι εμφανίζουν λιγότερη αποδοχή στο κόσμο δεν εμφανίζουν κάποιο outlier, το οποίο να υποδηλώνει ότι ένα συγκεκριμένο βίντεο είχε μεγαλύτερη αποδοχή. Η μόνη δυνατή ερμηνεία σε αυτή την παρατήρηση είναι ότι οι χρήστες δεν είχαν την υπομονή ή/και διάθεση να βλέπουν βίντεο στο timeline τους.

Τα επόμενα δύο διαγράμματα μελετούν τη σχέση μεταξύ των διαφορετικών ειδών δημοσιεύσεων και των άλλων δύο βασικών λειτουργιών που προσφέρουν τα κοινωνικά δίκτυα, σχολιασμού και κοινοποίησης.

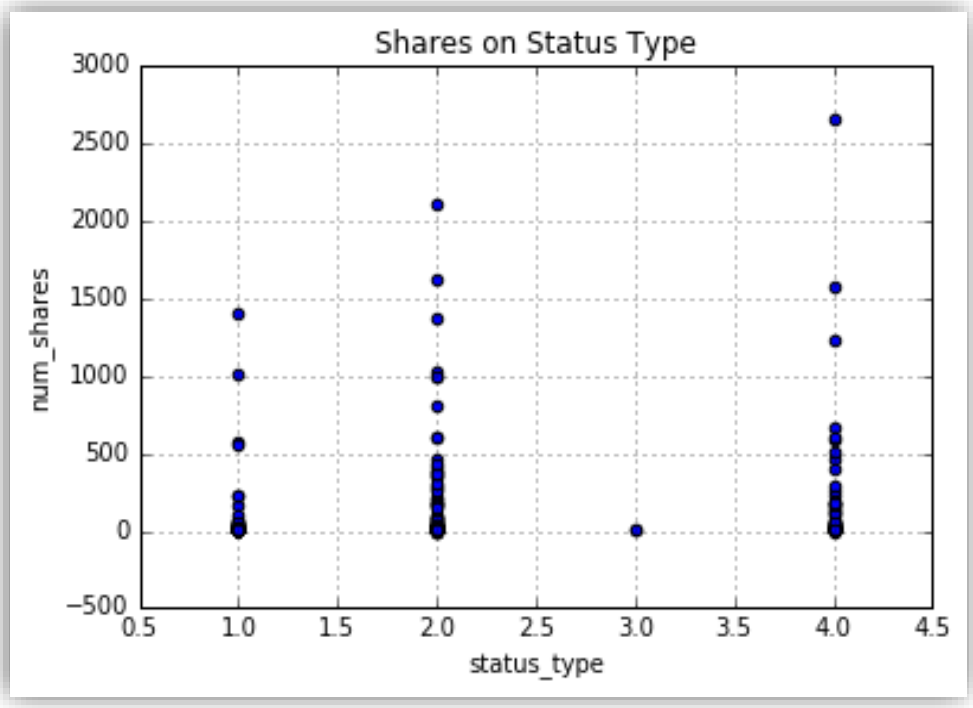


Διάγραμμα 7: Πλήθος σχολίων ανάλογα με το είδος της δημοσίευσης

Η σχέση σχολίων και είδος δημοσίευσης είναι διαφορετική από αυτή που υπήρχε μεταξύ δημοσιεύσεων και αντιδράσεων. Ο λόγος είναι ότι οι αντιδράσεις είναι 100 φορές περισσότερες από τα σχόλια οδηγώντας σε ιδιαίτερα έντονη κλιμάκωση των μοτίβων συμπεριφοράς των χρηστών.

Ακόμη παρατηρείται ότι οι χρήστες τείνουν να σχολιάζουν και στις δημοσιεύσεις με βίντεο, αλλά τα περισσότερα σχόλια πραγματοποιούνται, όπως περιμέναμε άλλωστε στις φωτογραφίες. Τα σχόλια στις δημοσιεύσεις που περιέχουν βίντεο είναι πολλά αν συγκριθούν με τις απλές αντιδράσεις των χρηστών στο ίδιο είδος δημοσίευσης. Η ερμηνεία που δίνω εγώ σε αυτό το φαινόμενο είναι ότι τις δημοσιεύσεις τύπου βίντεο τις παρακολουθούν λιγότεροι, αλλά πιο 'πιστοί' ακόλουθοι της εταιρίας.

Το επόμενο διάγραμμα παρουσιάζει τη σχέση που υπάρχει ανάμεσα στις αναδημοσιεύσεις των χρηστών, ανάλογα με το είδος της δημοσίευσης που πραγματοποιείται από τη σελίδα.



Διάγραμμα 8: Πλήθος αναδημοσιεύσεων ανάλογα με το είδος της δημοσίευσης

Στη σχέση μεταξύ των κοινοποιήσεων και τις διαφορετικές δημοσιεύσεις το μοτίβο που διαγράφεται είναι παρόμοιο με το μοτίβο σχολίων-δημοσιεύσεων. Η μόνη διαφορά είναι η παρουσία ενός outlier με πολλές κονιοποιήσεις σε δημοσίευση που εμπεριέχει βίντεο.

3.2 Οπτικοποίηση Δεδομένων Προερχομένων Από το Twitter

Εφόσον ολοκλήρωσα την οπτικοποίηση των δεδομένων που έχω συλλέξει από το Facebook, προχωράω στην οπτικοποίηση των δεδομένων που έχω συγκεντρώσει από το Twitter. Όπως στην ανάλυση του κώδικα για τη συλλογή δεδομένων, έτσι και στην οπτικοποίηση των δεδομένων για το Twitter, θα είμαι πιο φειδωλός σε περιγραφές και αναλύσεις για να μην επαναλαμβάνομαι.

3.2.1 Αντιδράσεις Χρηστών ανά Δημοσίευση

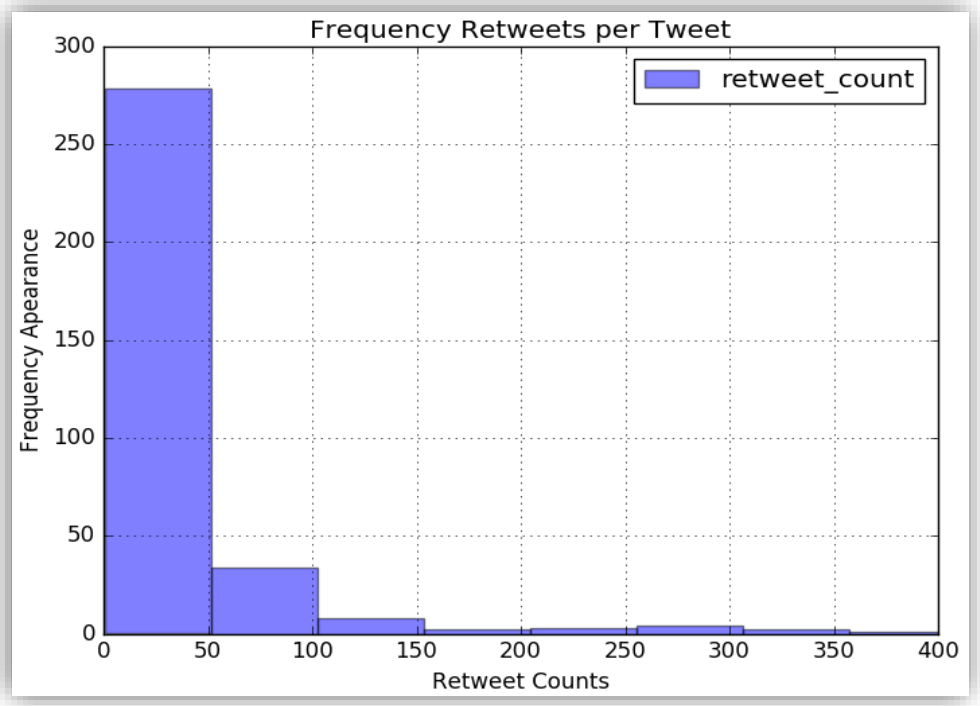
Το twitter προσφέρει μόνο δύο ειδών αντιδράσεις στους χρήστες του το retweet(αναδημοσίευση) και το favorite(αγαπημένο). Το API του twitter δεν συγκαταλέγει τις απαντήσεις(response) ως αντίδραση, αντίθετα τοποθετεί τις απαντήσεις και τις αναφορές(mentions) στην ίδια κατηγορία. Έτσι είναι λάθος να θεωρηθούν όλες οι αναφορές προς ένα λογαριασμό απαντήσεις σε tweets. Ακόμη, στην πρώιμη εξέταση και δοκιμή του API πραγματοποίησα ερώτημα για να εξετάσω τη φύση των ερωτημάτων και η πλειοψηφία αυτών είναι ερωτήσεις, κι όχι απαντήσεις ή αντιδράσεις σε tweets.

Οι αντιδράσεις ανά tweet φαίνονται στον πίνακα που ακολουθεί

Είδος αντίδρασης	**Συχνότητα/δημοσίευση**
retweet	45.100890
favorite	132.765579

Πίνακας 3: Είδη αντίδρασης και συχνότητα αντίδρασης ανά δημοσίευση

Οι χρήστες ξεκάθαρα χρησιμοποιούν το favorite τρεις φορές περισσότερο από το retweet και στα παρακάτω ιστογράμματα φαίνονται οι συχνότητες χρήσης των δύο αντιδράσεων. Στα δεδομένα που έχω συλλέξει από το twitter υπάρχουν 337 εγγραφές.

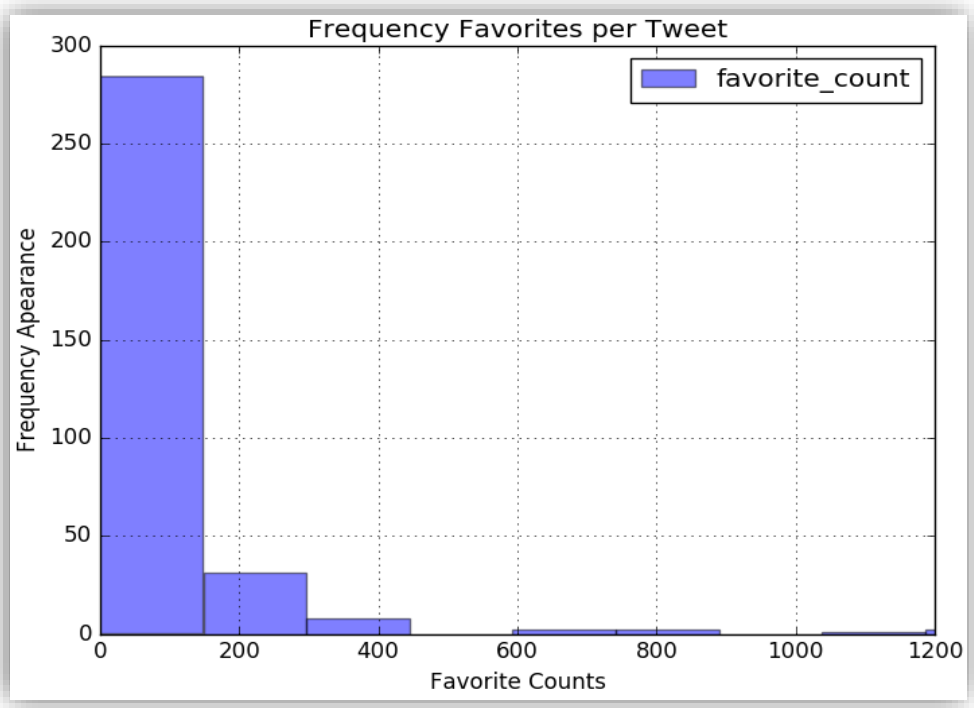


Διάγραμμα 9: Συχνότητα πραγματοποίησης retweet

Στο διάγραμμα στο οποίο μελετάω τη συχνότητα πραγματοποίησης retweet οι αντιδράσεις συσσωρεύονται στο διάστημα [0-50], δείχνοντας ότι σπάνια υπάρχει tweet,

έτσι ώστε να προκαλέσει περισσότερο τους χρήστες, από ότι στην πλειοψηφία των περιπτώσεων.

Προχωράω στην μελέτη της συχνότητας εμφάνισης favorite στα tweets που πραγματοποιούνται από τους λογαριασμούς που εξετάζω.



Διάγραμμα 10: Συχνότητα πραγματοποίησης favorite

Στην περίπτωση των favorites, η μείωση των αντιδράσεων είναι κι αυτή εκθετική και παρουσιάζει πανομοιότυπο μοτίβο μείωσης. Επειδή η το 'favorite' ως αντίδραση εμφανίζεται τρεις φορές συχνότερα, πραγματοποίησα και την απαραίτητη αλλαγή κλίμακας, έτσι οι δύο αντιδράσεις φαίνεται να έχουν ακριβώς το ίδιο μοτίβο εμφάνισης.

3.2.2 Κατανομή των Διαφορετικών Μερών του Λόγου

Ακολουθώντας τη δομή που είχα ακολουθήσει για τα δεδομένα του Facebook, έτσι και σε αυτό το κεφάλαιο θα προχωρήσω στην οπτικοποίηση για την κατανομή των διαφορετικών μερών του λόγου μέσα στα tweets που έχω στο δείγμα μου. Η συχνότητα

εμφάνισης των διαφόρων μερών του λόγου ανά tweet φαίνεται στον πίνακα που ακολουθεί.

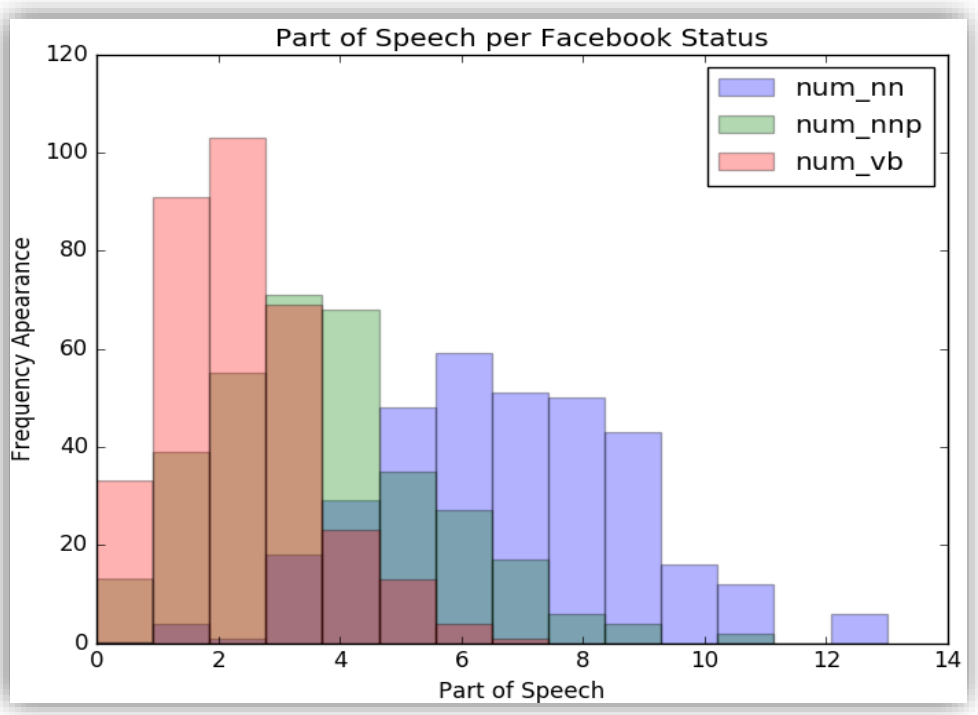
Μέρος του Λόγου	**Συχνότητα/δημοσίευση**
num_nn	6.777448
num_nnp	3.548961
num_vb	2.053412
num_in	1.278932
num_dt	1.142433
num_jj	1.136499
num_prp	1.002967
num_nns	0.753709
num_cd	0.670623
num_rb	0.554896
num_vbz	0.525223
num_cc	0.323442
num_to	0.320475
num_vbp	0.305638
num_vbg	0.234421
num_vbn	0.166172
num_vbd	0.163205
num_md	0.136499
num_rp	0.112760
num_wrb	0.106825
num_wp	0.094955
num_jjr	0.083086
num_fw	0.068249
num_jjs	0.059347
num_sym	0.053412
num_rbr	0.038576
num_wdt	0.035608
num_pos	0.029674
num_nnps	0.020772
num_ex	0.011869
num_pdt	0.008902
num_ls	0.002967
num_wp6	0.000000
num_prp6	0.000000
num_uh	0.000000

Πίνακας 4: Συχνότητα εμφάνισης μερών του λόγου ανά tweet

Αν συγκρίνουμε τη χρησιμοποίηση διαφόρων μερών του λόγου στο Facebook, με τη χρησιμοποίηση μερών του λόγου στο Twitter, θα διαπιστωθεί έντονη ομοιότητα. Η μόνη έντονη διαφορά έγκειται στην έντονη χρησιμοποίηση κυρίων ονομάτων στα δεδομένα του Facebook.

Υπάρχουν 7 μέρη του λόγου εμφανίζονται σε κάθε δημοσίευση, τα ίδια 7 που παρατηρήθηκαν και στα δεδομένα του Facebook, και άλλα τρία μέρη του λόγου που δεν εμφανίστηκαν ποτέ στο δείγμα που έχω στην κατοχή μου.

Παρακάτω παραθέτω το ιστόγραμμα που εμφανίζει τη κατανομή των τριών μερών του λόγου με τη μεγαλύτερη συχνότητα(nn – ουσιαστικό σε ενικό αριθμό, nnp – κύρια ονόματα και vb - ρήμα σε βασική μορφή).



Διάγραμμα 11: Συχνότητα εμφάνισης των τριών πιο συχνά χρησιμοποιημένων μερών του λόγου

Οι κατανομές των τριών πιο συχνών μερών του λόγου μπορούν να χαρακτηριστούν και οι 3 ως κανονικές κατανομές, με διαφορετικές μέσες τιμές η κάθε μία. Οι τρεις μέσες τιμές είναι διατεταγμένες σύμφωνα με τη συχνότητα εμφάνισης τους. Συγκρίνοντας αυτό το διάγραμμα με το αντίστοιχο που εξήγαγα από τα δεδομένα του Facebook θα μπορούσα να χαρακτηρίσω τα tweets πιο φυσικά ως προς τη δομή του λόγου.

Το API του Twitter δεν προσφέρει τη δυνατότητα διαχωρισμού των δημοσιεύσεων σε κατηγορίες όπως το Facebook, όπου τις χωρίζει σε απλό κείμενο, φωτογραφία, βίντεο ή υπερσύνδεσμο. Για αυτό το λόγο δεν υπάρχει υποκεφάλαιο για την επίδραση των διαφορετικών ειδών δημοσίευσης στα δεδομένα που έχουν εξαχθεί από το Twitter.

4. Χρήση Λεξικών

Εφόσον έχω ολοκληρώσει τη συλλογή δεδομένων τόσο από το Facebook όσο και από το Twitter, το επόμενο βήμα είναι η αξιολόγηση των δεδομένων από λεξικά. Υπάρχουν λεξικά τα οποία έχουν δημιουργηθεί έτσι ώστε να αποτυπώνουν την υποκειμενικότητα ή αντικειμενικότητα μίας λέξης, καθώς επίσης και την χροιά της (θετική, αρνητική) όταν η λέξη εκφράζει υποκειμενικότητα.

Για την αξιολόγηση του Sentiment Analysis μέσα σε κείμενο υπάρχουν δύο κυρίαρχες προσεγγίσεις . Η πρώτη προσέγγιση ξεκινάει με κείμενα (π.χ. tweets) τα οποία έχουν αξιολογηθεί χειροκίνητα και μέσω τεχνικών μηχανικής μάθησης με επίβλεψη (supervised machine learning) ταξινομούν νέα κείμενα ανάλογα με τη χροιά που αυτά έχουν [2]. Η δεύτερη προσέγγιση είναι μέσω της δημιουργίας ενός λεξικού, το οποίο εντοπίζει δείγματα Sentiment Analysis, και αξιολογεί το κείμενο προς ανάλυση μέσω συγκεκριμένων συναρτήσεων, ανάλογα με το βαθμό ταύτισης των λέξεων και φράσεων που υπάρχουν στο κείμενο και στο λεξικό.

4.1 Χρησιμοποίηση Λεξικού AFINN

Στη συγκεκριμένη εργασία χρησιμοποιώ ένα συνδυασμό των δύο παραπάνω μεθόδων, αξιολογώ μέσω λεξικού τα κείμενα που έχω συλλέξει και έπειτα κατηγοριοποιώ τα δεδομένα μέσω τεχνικών μηχανικής μάθησης. Το πρώτο λεξικό που χρησιμοποιώ είναι αυτό του AFINN [34]. Το AFFIN είναι ένα λεξικό αγγλικών λέξεων το οποίο έχει σχεδιαστεί για να αποδίδει χροιά (valence) στις λέξεις που υπάρχουν μέσα στο λεξικό. Η χροιά στις λέξεις αποδίδεται με έναν ακέραιο ανάμεσα στο μείον πέντε (-5) και στο συν πέντε(5), από την αρνητικότερη στην θετικότερη χροιά. Οι λέξεις έχουν βαθμολογηθεί χειροκίνητα από τον Finn Arup Nielsen ανάμεσα στις χρονιές 2009-2011. Χαρακτηριστικές εγγραφές του λεξικού φαίνονται στον πίνακα που ακολουθεί.

Λέξη	Βαθμολογία
accepting	1
accepts	1
ACCESS	0
accident	-2
accidental	-2

Πίνακας 5: Παραδείγματα εγγραφών του λεξικού AFINN

Η κατασκευή του συγκεκριμένου λεξικού έχει ως σκεπτικό δημιουργίας την εύρεση συναισθήματος σε κείμενου τύπου microblog, μάλιστα η χειροκίνητη αξιολόγηση των λέξεων που υπάρχουν στο λεξικό έχουν εξαχθεί από το Twitter. Στο λεξικό του AFINN περιλαμβάνονται λέξεις που ανήκουν στην αργκό καθώς επίσης και ύβρις. Το λεξικό έχει χρησιμοποιηθεί για Sentiment Analysis σε δημοσιεύσεις από το Twitter [35], αλλά δεν είναι πλήρως δοκιμασμένο.

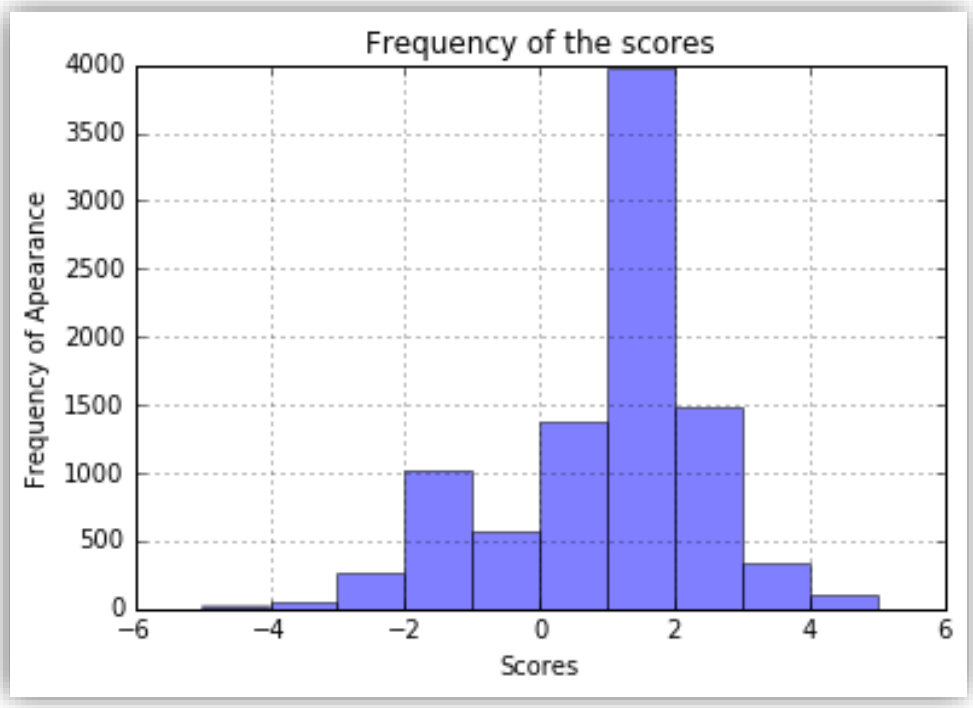
Η διαφορά του συγκεκριμένου λεξικού με τα υπόλοιπα έγκειται στην συμπερίληψη συγκεκριμένων λέξεων που χρησιμοποιούνται στο Διαδίκτυο, όπως κάποιες υβριστικές λέξεις καθώς και ακρωνύμια, όπως τα ‘WTF’ και ‘lol’. Η συμπερίληψη τέτοιων λέξεων μπορεί να αυξήσει την επιτυχία στην κατηγοριοποίηση των δεδομένων, όταν τα δεδομένα προέρχονται από κείμενα μικρού μήκους και με ανεπίσημο χαρακτήρα, όπως κείμενα που υπάρχουν σε φόρουμ και κοινωνικά δίκτυα.

Το λεξικό διαθέτει συνολικά 9,164 εγγραφές, των οποίων η μέση τιμή ισούται με 0.513967. Το λεξικό βαθμολογεί τις λέξεις στο πεδίο τιμών [-5,5] με το πλήθος των βαθμολογιών για την κάθε τιμή να φαίνεται στο παρακάτω πίνακα.

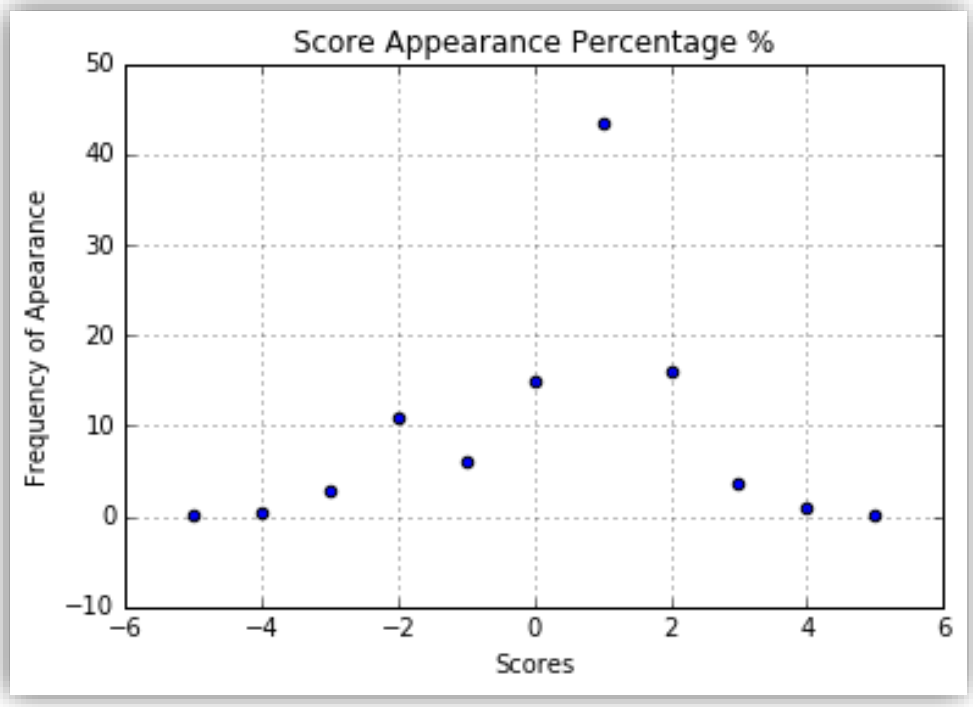
Σκορ Εγγραφής	Πλήθος Εμφάνισης	Συχνότητα/Εγγραφή
-5	16	0.17459624618
-4	44	0.48013967699
-3	264	2.88083806198
-2	1010	11.0213880402
-1	567	6.18725447403
0	1380	15.0589262331
1	3971	43.332605849
2	1475	16.0955914448
3	337	3.67743343518
4	87	0.9493670886
5	13	0.14185945002

Πίνακας 6: Πλήθος εμφάνισης τιμών και συχνότητας ανά εγγραφή για τα πιθανά σκορ του AFINN

Η οπτική απεικόνιση του πίνακα φαίνεται στα δύο διαγράμματα που ακολουθούν. Στο πρώτο διάγραμμα διακρίνεται η σχέση των σκορ με το πλήθος εμφανίσεων σε απόλυτη τιμή και στο δεύτερο, διάγραμμα διασποράς, διακρίνεται η συχνότητα εμφάνισης σε ποσοστό επί τοις εκατό των συγκεκριμένων σκορ ανά εγγραφή.



Διάγραμμα 12: Πλήθος εμφάνισης των διαθέσιμων σκορ στο λεξικό του AFINN



Διάγραμμα 13: Πιθανότητα εμφάνισης % διαθέσιμων σκορ στο λεξικό του AFINN

Για την αξιολόγηση των δεδομένων που έχω συλλέξει δημιούργησα μία συνάρτηση η οποία πραγματοποιεί τη σύνδεση με τη βάση δεδομένων και ερώτημα σε αυτή για να ανακτήσει των δημοσιεύσεων. Έπειτα έχω δημιουργήσει μία επανάληψη για όλες τις δημοσιεύσεις, τοποθετώ τις δημοσιεύσεις σε μία λίστα και διαχωρίζω τη δημοσίευση στις επιμέρους λέξεις. Η συνάρτηση διαθέτει ακόμα μία επανάληψη για τις λέξεις που αποτελούν τη δημοσίευση, μέσα στην οποία διαβάζεται το λεξικό και συγκρίνονται οι εγγραφές του με τις λέξεις της κάθε δημοσίευσης και ανάλογα με τη βαθμολογία της κάθε λέξης αθροίζεται το συνολικό σκορ της δημοσίευσης. Τέλος, ανανεώνεται η βάση δεδομένων με το νέο πεδίο machine score.

```
# call the dictionary that creates the machine score
def machineScore():
                        .
                        .
                        .
    query = ("SELECT status_id, status_message FROM statuses_pos ")
    cursor.execute(query)
    for (status_id, status_message) in cursor:
        my_list.append(status_message)
                        .
                        .
                        .

        words = status_message.split()
                        .
                        .
                        .
        machine_result = 0
        for counter in range (0,len(words)):
                        .
                        .
                        .
            with open('AFINN.csv', 'rb') as f:
                reader = csv.reader(f)
                for row in reader:
                        .
                        .
                        .
                    if row[0] == words[counter]:
                        machine_result = machine_result + int(row[1])
                        .
                        .
                        .
        try:
            with cnx2.cursor() as cursor2:
                # Create a new record
                # insert the data we pulled into db
                cursor2.execute ("""
                        UPDATE statuses_pos
                        SET  computer_score=%s
                        WHERE status_id=%s
                    """, (machine_result, status_id))
                        .
                        .
                        .
```

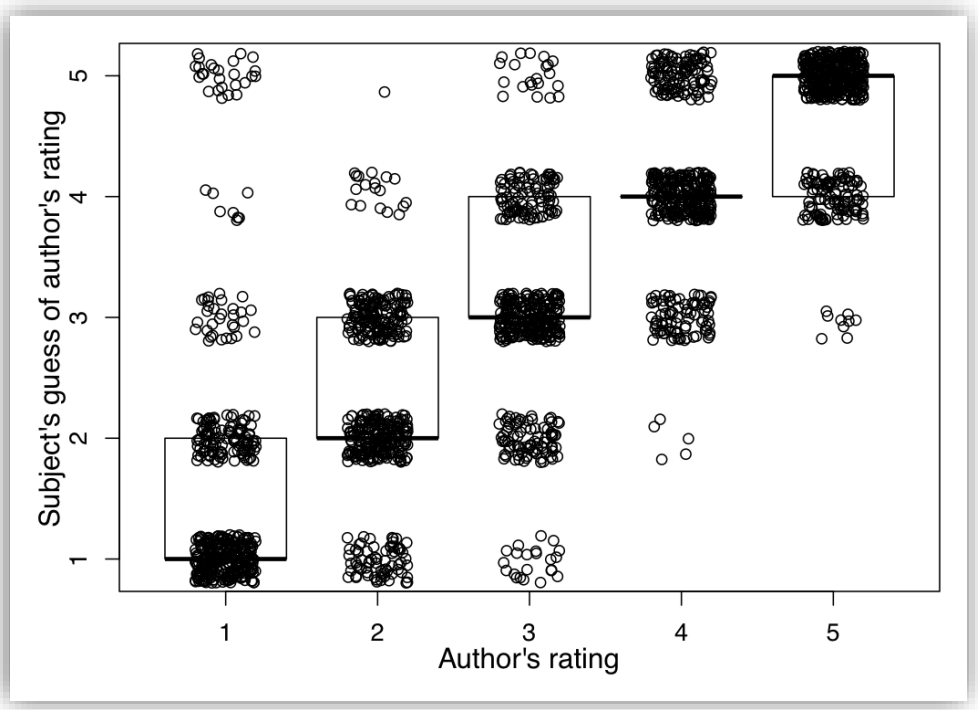
Κώδικας 21:Μέρος της συνάρτησης machineScore(), κάνει κλήση στη βάση δεδομένων, πραγματοποιεί την αξιολόγηση μέσω λεξικού και ανανεώνει τη βάση δεδομένων. Οι κάθετες τελείες ανάμεσα στις εντολές υποδηλώνουν κώδικα που έχει αφαιρεθεί για λόγους οικονομίας χώρου

4.2 Χρησιμοποίηση Λεξικών Προερχομένων Από Ιστοσελίδες

Έχοντας ολοκληρώσει την ανάλυση του λεξικού AFINN, προχωράω στην μελέτη μιας οικογένειας λεξικών. Το AFINN δημιουργήθηκε από την χειροκίνητη αξιολόγηση tweets, πιο συγκεκριμένα ξεκίνησε με tweets που συσχετίζονταν με το Συνέδριο των Ηνωμένων Εθνών για την Κλιματική Αλλαγή (United Nations Climate Change Conference Copenhagen 2009, COP 15).

Τα επόμενα τέσσερα λεξικά που θα εξετάσω έχουν εμπνευστεί από δημοφιλείς ιστοσελίδες που περιέχουν κριτικές είτε για ταινίες είτε για ταξιδιωτικούς προορισμούς είτε για βιβλία είτε για εστιατόρια. Η θεμελιώδης ιδέα πίσω από τη δημιουργία αυτών των λεξικών είναι ότι οι επισκέπτες των συγκεκριμένων ιστοσελίδων μπορούν με μια μικρή απόκλιση να μαντέψουν την τελική βαθμολογία του χρήστη που πραγματοποιεί την κριτική μέσα από τα γραφόμενα του.

Στην παρακάτω εικόνα διακρίνεται το μοτίβο εικασίας των χρηστών για τη βαθμολογία μίας κριτικής. Επίσης διακρίνεται και το ποσοστό επιτυχίας αυτών των εικασιών.



Εικόνα 1: Πειραματικά αποτελέσματα που δείχνουν την αξιόπιστη εικασία των χρηστών για πιθανές βαθμολογίες σε κριτικές σε σύστημα πέντε (5) αστέρων [36]. Τα δεδομένα πάρθηκα από κριτικές στην ιστοσελίδα opentable.com

4.2.1 Χρήση λεξικού imdb

Σε αυτό το υποκεφάλαιο μελετάω το λεξικό που έχει δημιουργηθεί από τον Alexander Potts [37], η δημιουργία του έχει βασιστεί στις κριτικές χρηστών σε ταινίες που είναι καταχωρημένες στην ιστοσελίδα του imdb [38]. Για αυτή την εργασία η προσωρινή ονομασία που θα αναθέσω στο συγκεκριμένο λεξικό είναι ‘λεξικό του imdb’.

Το λεξικό του imdb διαθέτει συνολικά 631,040 εγγραφές και 5 στήλες. Στη πρώτη στήλη τοποθετείται η λέξη, στη δεύτερη το μέρος του λόγου στο οποίο ανήκει η λέξη, στη τρίτη υπάρχει μία από τις δέκα δυνατές βαθμολογίες που δίνονται, στη τέταρτη το σύνολο των εμφανίσεων της λέξης σε κριτική με τη συγκεκριμένη βαθμολογία και στη τελευταία στήλη υπάρχει το σύνολο των λέξεων που εμφανίζονται σε κριτικές με τη συγκεκριμένη

βαθμολογία. Δηλαδή η κάθε λέξη υπάρχει 10 φορές στο λεξικό, μία για κάθε κατηγορία βαθμολογίας. Για την κατανόηση της δομής του λεξικού του imdb παραθέτω τον παρακάτω πίνακα, ο οποίος διαθέτει τις βαθμολογίες για τη λέξη 'bad'.

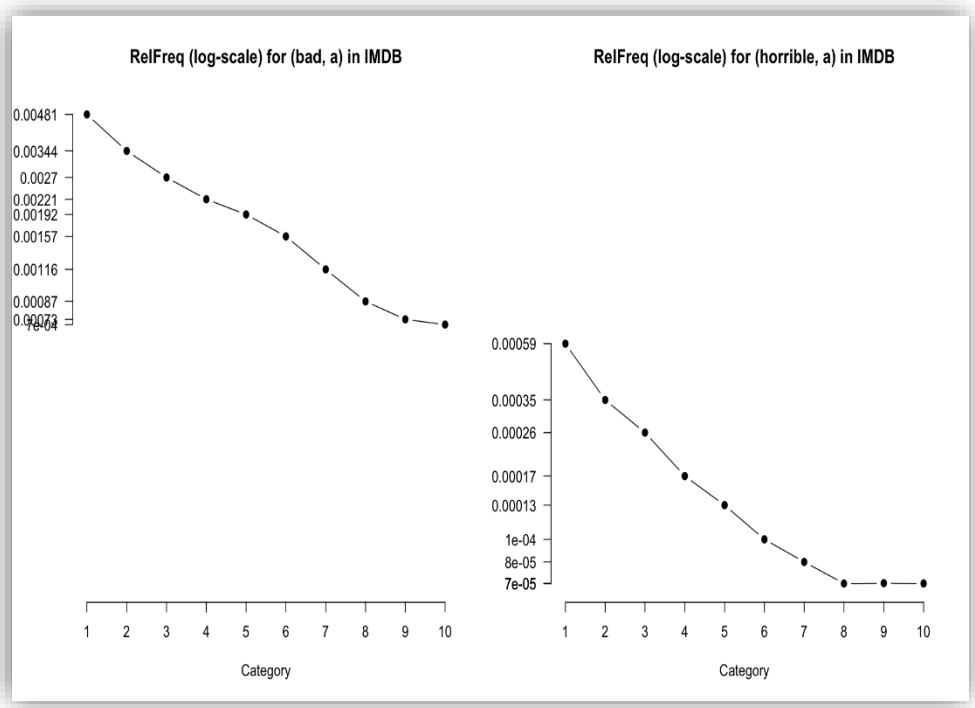
Word	**Tag**	**Category**	**Count**	**Total**
bad	a	1	122232	25395214
bad	a	2	40491	11755132
bad	a	3	37787	13995838
bad	a	4	33070	14963866
bad	a	5	39205	20390515
bad	a	6	43101	27420036
bad	a	7	46696	40192077
bad	a	8	42228	48723444
bad	a	9	29588	40277743
bad	a	10	51778	73948447

Πίνακας 7: Εγγραφές για τη λέξη 'bad' στο λεξικό του imdb

Στο λεξικό του imdb η τρίτη στήλη, η οποία δηλώνει μία από τις δέκα πιθανές εγγραφές, επαναλαμβάνεται κάθε δέκα (10) εγγραφές, όπως και η πέμπτη στήλη η οποία αναφέρει τις συνολικές λέξεις στη συγκεκριμένη κατηγορία.

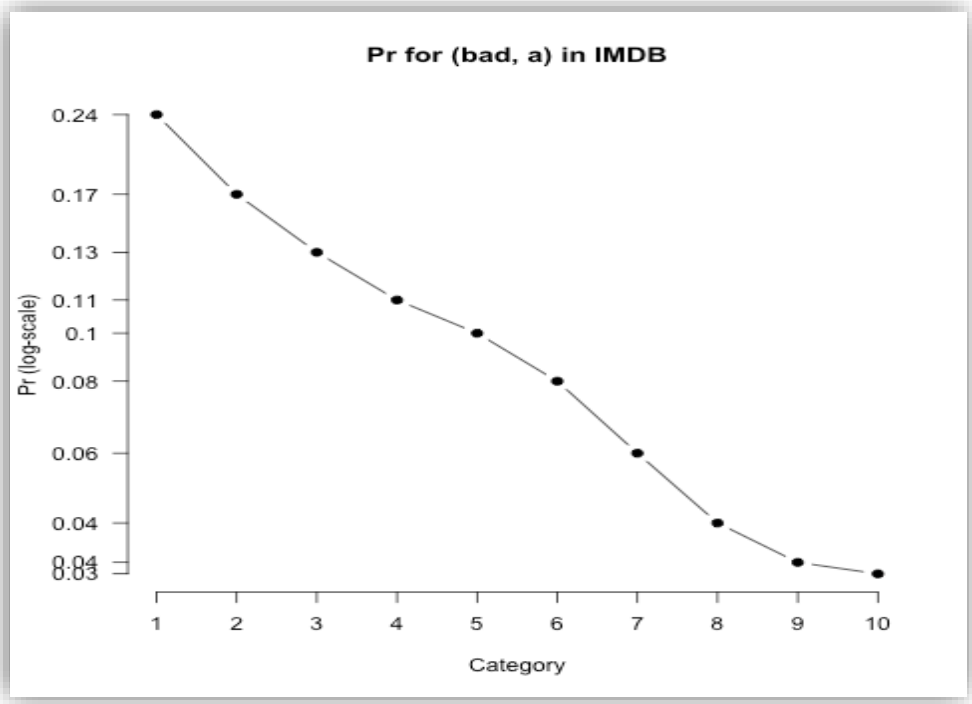
Το λεξικό δεν διαθέτει συγκεκριμένο σκορ για την κάθε εγγραφή, έτσι πρέπει να το δημιουργήσω εγώ. Η πρώτη μετρική που δημιούργησα είναι το Relative Frequency (Σχετική Συχνότητα Εμφάνισης), η οποία είναι το πηλίκο του αριθμού εμφανίσεων της κάθε λέξης σε συγκεκριμένη κατηγορία, προς το συνολικό αριθμό εγγραφών της συγκεκριμένης κατηγορίας. Δηλαδή για κάθε γραμμή είναι το πηλίκο της τρίτης στήλης προς την τέταρτη.

Το μειονέκτημα της συγκεκριμένης μετρικής, είναι ότι είναι ιδιαίτερα ευαίσθητη στη συνολική συχνότητα της εγγραφής. Αυτή η ευαισθησία φανερώνεται στο παρακάτω διάγραμμα, όπου η λέξη bad εμφανίζεται πιο συχνά ακόμα και σε κριτικές ταινιών που βαθμολογούνται με 10 σε σχέση με τη λέξη horrible σε κριτικές που βαθμολογούν την ταινία με 1. Γίνεται κατανοητό ότι το Relative Frequency δεν αποτυπώνει τη βαρύτητα των λέξεων.

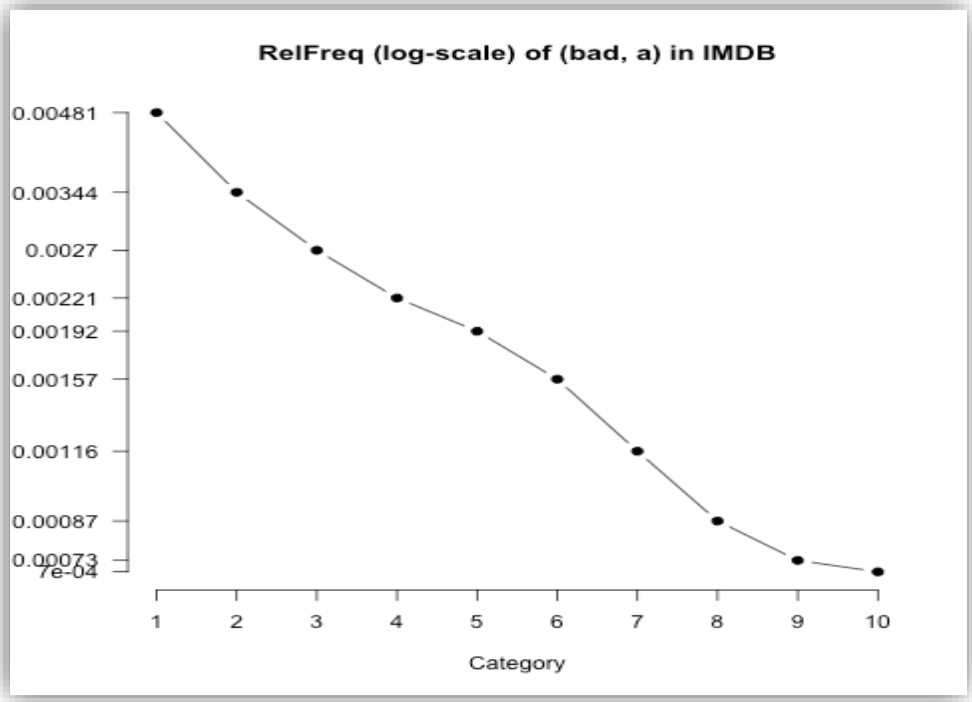


Διάγραμμα 14: Σχετική συχνότητα των λέξεων 'bad' και 'horrible' σε λογαριθμική κλίμακα [36]

Επειδή το Relative Frequency αποδεικνύεται αδύναμο να σταθεί ως αντικειμενική μετρική σκορ, δημιουργώ τη μετρική του Probabilities (Πιθανότητα). Η συγκεκριμένη μετρική είναι ουσιαστικά η μεταβλητή του Relative Frequency σε διαφορετική κλίμακα. Είναι το πηλίκο του Relative Frequency της κάθε εγγραφής, προς το σύνολο εμφανίσεων της λέξης. Αυτή η μετρική έχει την ίδια διακύμανση με τη Relative Frequency με τη διαφορά ότι βρίσκεται στην κλίμακα [0,1]. Οι δύο μετρικές φαίνονται στα παρακάτω διαγράμματα. Οι δύο μετρικές φαίνονται στα διαγράμματα που ακολουθούν.



Διάγραμμα 15: Η πιθανότητα εμφάνισης της λέξης 'bad' σε κάθε πιθανή κατηγορία [36]



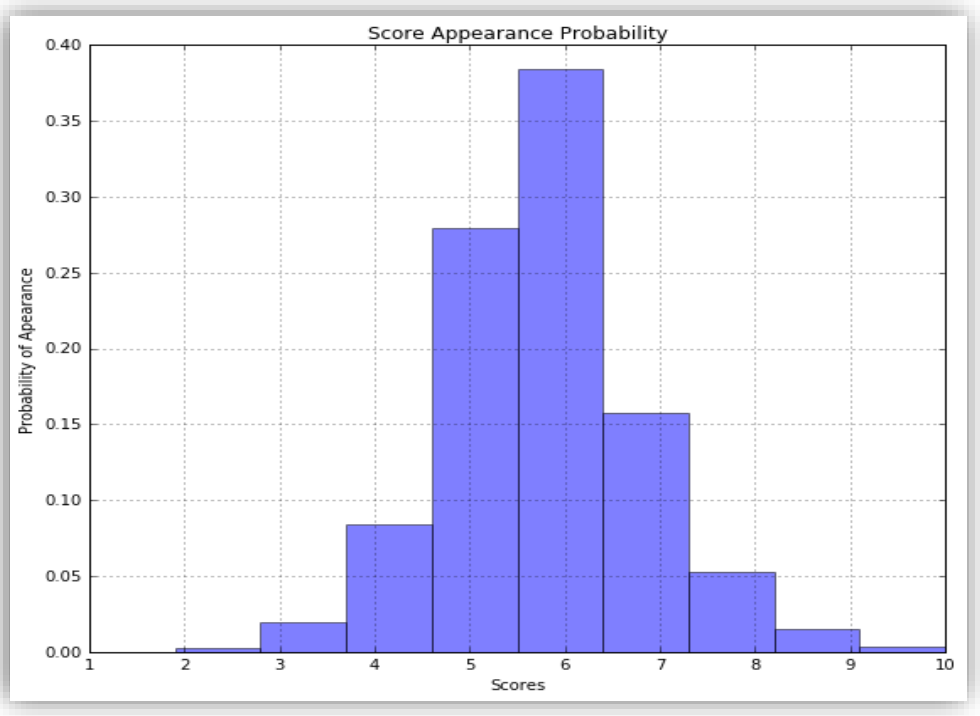
Διάγραμμα 16: Η σχετική συχνότητα εμφάνισης της λέξης 'bad' σε κάθε πιθανή κατηγορία [36]

Εφόσον έχω δημιουργήσει και τη μετρική Probabilities, το τελευταίο βήμα είναι η δημιουργία της τελικής μετρικής για τη βαρύτητα της κάθε λέξης. Δημιούργησα μία λίστα βαρύτητας με 10 θέσεις [1,2,3,4,5,6,7,8,9,10], την οποία πολλαπλασιάζω με την πιθανότητα εμφάνισης(Probabilities) κάθε λέξης και έπειτα αθροίζω τα επιμέρους γινόμενα. Παραδείγματος χάριν, έστω οι πιθανότητες εμφάνισης της λέξης x σε κάθε κατηγορία είναι : [0, 0, 0.2, 0.1, 0, 0, 0, 0, 0.7, 0], θα αθροίσω τα επιμέρους γινόμενα 3*0.2 + 4*0.1 + 10*0.7 τα οποία καταλήγουν σε βαθμολογία ίση με 8. Έτσι δημιουργείται η τελική βαθμολογία για κάθε λέξη. Στον πίνακα που παραθέτω παρακάτω δίνω δέκα (10) εγγραφές, στις οποίες διακρίνεται η λέξη, το μέρος του λόγου της λέξης και η βαθμολογία που έχω θέσει στη λέξη.

Word	**Tag**	**Expected Rate**
backwards	r	5.195962
backwater	n	5.68107
backwoods	n	5.134421
backyard	n	4.765561
baclanova	n	7.264575
bacon	n	5.627349
bacri	n	7.360036
bacteria	n	4.40163
bacterial	a	4.592892
bad	a	3.794504

Πίνακας 8: Δέκα εγγραφές στην τελική έκδοση του λεξικού

Το λεξικό του imdb μετά το τέλος της επεξεργασίας διαθέτει 63,104 εγγραφές οι οποίες λαμβάνουν σκορ στο εύρος [1,10]. Οι τιμές που λαμβάνουν οι λέξεις δεν είναι ακέραιες για αυτό το λόγο δεν μπορώ να παραθέσω έναν πίνακα με τη πλήθος εμφάνισης και συχνότητα ανά εγγραφή για κάθε τιμή, για αυτό το λόγο παραθέτω μόνο διάγραμμα με το τη πιθανότητα εμφάνισης των διαφόρων σκορ σε ποσοστό επί τοις εκατό. Η μέση τιμή που λαμβάνουν οι λέξεις είναι ίση με 5.773240 με διακύμανση 1.037391, συνολικά υπάρχουν 63,058 μοναδικά σκορ.



Διάγραμμα 17: Ποσοστό εμφάνισης διαθέσιμων σκορ στο λεξικό του imdb*, ομαδοποιημένα σε 10 υποσύνολα*

4.2.2 Χρήση λεξικού Amazon/TripAdvisor

Το συγκεκριμένο λεξικό, όπως και τα δύο λεξικά που ακολουθούν, μπορούν να θεωρηθούν ως φυσική συνέχεια του λεξικού του imdb. Δημιουργήθηκαν από το ίδιο άτομο, με την ίδια λογική και με την ίδια τεχνική [39].

Στο συγκεκριμένο λεξικό οι εγγραφές έχουν εξαχθεί από τις ιστοσελίδες των Amazon [40] και TripAdvisor [41] και έχουν ομαδοποιηθεί σε ένα σύνολο, έτσι θεωρώ ότι το λεξικό είναι μίξη αυτών των δύο ιστοσελίδων και το ονομάζω, στα πλαίσιο αυτής της εργασίας, λεξικό Amazon/TripAdvisor.

Στις συγκεκριμένες ιστοσελίδες η βαθμολογία των χρηστών είναι στην κλίμακα [1,5] είτε για προϊόν που διατίθεται μέσω του Amazon είτε για κριτική μέρους με τουριστικό ενδιαφέρον στο TripAdvisor. Το συγκεκριμένο αρχείο έχει 6 στήλες, στην πρώτη στήλη υπάρχει η λέξη μαζί με την ετικέτα της, στην μορφή WORD/tag, στη δεύτερη στήλη η

βαθμολογία στην οποία συναντάται η συγκεκριμένη λέξη, για το υποσύνολο του Amazon/TripAdvisor είναι [1,5], στην τρίτη στήλη ο δημιουργός του λεξικού θέτει την κατηγορία της στην κλίμακα του [-0.5,0.5] αυξάνοντας κατά 0.5 σε κάθε εγγραφή, στην τέταρτη υπάρχει ένας αριθμός που υποδηλώνει τον αριθμό εμφανίσεων της λέξης στη συγκεκριμένη βαθμολογία, στην πέμπτη υπάρχει ο συνολικός αριθμός λέξεων που εμφανίζονται στην ανάλογη κατηγορία.

Τέλος, για την έκδοση του λεξικού που έχω εγώ στην κατοχή μου [42], όλα τα λεξικά αυτού του υποκεφαλαίου βρίσκονται στο ίδιο αρχείο, στην τελευταία στήλη υπάρχει αναφορά στη σελίδα από όπου προήλθαν τα δεδομένα (Amazon/TripAdvisor, Goodreads, imdb, Opentable).

Για την κατανόηση της δομής του λεξικού του Amazon/TripAdvisor παραθέτω τον παρακάτω πίνακα, ο οποίος διαθέτει τις βαθμολογίες για τη λέξη ‘bad’.

Word	**Tag**	**Category**	**Count**	**Total**	**Corpus**
bad/a	1	-0.5	1241	3419923	Amazon/TripAdvisor
bad/a	2	-0.25	791	3912625	Amazon/TripAdvisor
bad/a	3	0	870	6011388	Amazon/TripAdvisor
bad/a	4	0.25	1301	10187257	Amazon/TripAdvisor
bad/a	5	0.5	2025	16202230	Amazon/TripAdvisor

Πίνακας 9: Εγγραφές για τη λέξη 'bad' στο λεξικό του Amazon/TripAdvisor

Όπως στο λεξικό του imdb, έτσι και στο λεξικό του Amazon/TripAdvisor πρέπει να δημιουργήσω το δικό μου σκορ για την κάθε εγγραφή του λεξικού. Η διαδικασία για την εξαγωγή σκορ για τις εγγραφές του λεξικού είναι ίδια με αυτή που ακολούθησα στο λεξικό του imdb. Δημιούργησα την μετρική του Relative Frequency για την κάθε εγγραφή, στη συνέχεια όρισα την πιθανότητα εμφάνισης της λέξης για την κάθε κατηγορία και τέλος για την μετρική του σκορ πολλαπλασίασα τις πιθανότητες εμφάνισης με το σύνολο των πιθανών βαθμολογιών, [1,2,3,4,5].

Το συγκεκριμένο λεξικό διαθέτει κάποιες διπλοεγγραφές. Ο λόγος ύπαρξης αυτών των διπλοεγγραφών είναι η διπλή ετικέτα που μπορεί να διαθέτει μία λέξη. Παραδείγματος χάριν, η λέξη *bad* μπορεί να έχει ετικέτα είτε */r* είτε */a* ανάλογα με το μέρους του λόγου στο οποίο ανήκει. Στην εργασία μου δεν εφάρμοσα σύγκριση του μέρους του λόγου για τη λέξη όταν βρίσκεται μέσα στην πρόταση και έπειτα ανάθεση του κατάλληλου σκορ, αλλά εξήγαγα το μέσο

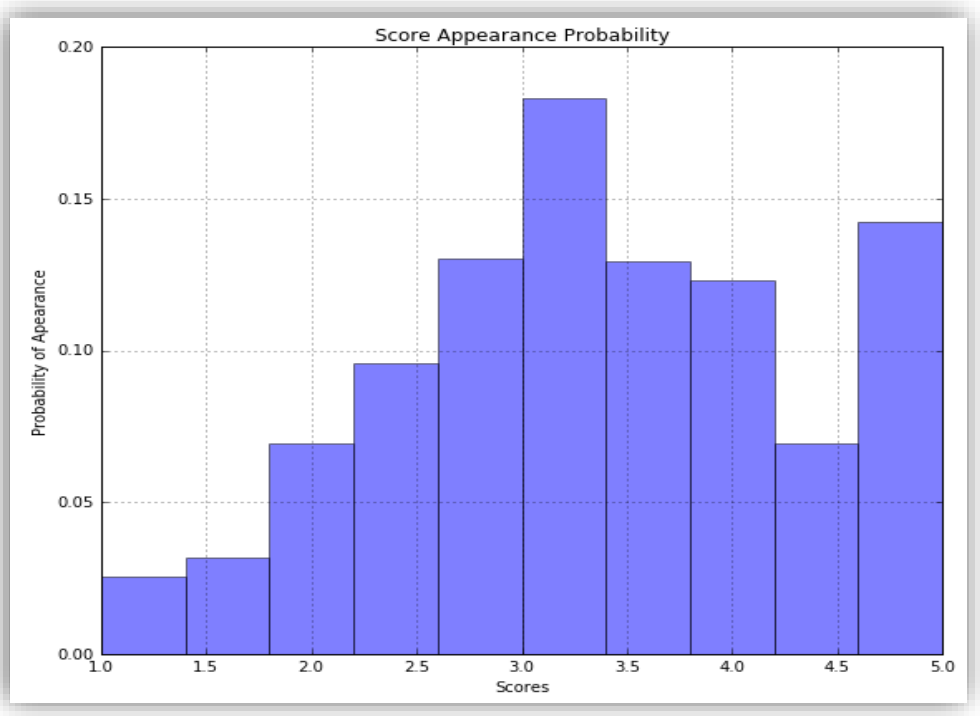
Στον πίνακα που παραθέτω παρακάτω δίνω δέκα (10) εγγραφές, στις οποίες διακρίνεται η λέξη, η βαθμολογία που έχω θέσει στη λέξη και στην τρίτη στήλη φαίνεται η μέση τιμή σε περίπτωση που υπάρχει διπλοεγγραφή. Στις περιπτώσεις που δεν υπάρχει διπλοεγγραφή θέτω τιμή της στην μετρική ίση με την τιμή του σκορ. Στον πίνακα που παραθέτω η μόνη λέξη που διαθέτει διπλοεγγραφή είναι η λέξη bad.

Word	**Expected Rate**	**Average of Duplicates**
bad	2.428297	2.635769
badly	2.164226	2.164226
baffled	3	3

baffling	5	5
baggy	1.697156	1.697156
baked	3.880473	3.880473
baking hot	2.554987	2.554987
balanced	3.072097	3.072097
bald	2.897826	2.897826
baldly	5	5

Πίνακας 10: Δέκα εγγραφές στην τελική έκδοση του λεξικού

Το λεξικό του Amazon/TripAdvisor μετά το τέλος της επεξεργασίας διαθέτει 9,686 εγγραφές οι οποίες λαμβάνουν σκορ στο εύρος [1,5]. Οι τιμές που λαμβάνουν οι λέξεις δεν είναι ακέραιες για αυτό το λόγο δεν μπορώ να παραθέσω έναν πίνακα με τη πλήθος εμφάνισης και συχνότητα ανά εγγραφή για κάθε τιμή, για αυτό το λόγο παραθέτω μόνο διάγραμμα με το τη πιθανότητα εμφάνισης των διαφόρων σκορ σε ποσοστό επί τοις εκατό. Η μέση τιμή που λαμβάνουν οι λέξεις είναι ίση με 3.368708 με διακύμανση 0.9815, συνολικά υπάρχουν 4,931 μοναδικά σκορ. Άξιο παρατήρησης είναι ότι υπάρχουν 1210 λέξεις οι οποίες λαμβάνουν σκορ ίσο με πέντε(5), δηλαδή 24.5% των εγγραφών βαθμολογούνται με το υψηλότερο δυνατό σκορ.



Διάγραμμα 18: Πιθανότητα εμφάνισης διαθέσιμων σκορ στο λεξικό του Amazon/TripAdvisor, ομαδοποιημένα σε 10 υποσύνολα

4.2.3 Χρήση λεξικού Goodreads

Το λεξικό που εξετάζω σε αυτό το υποκεφάλαιο προέρχεται από τη μελέτη δεδομένων της ιστοσελίδας Goodreads [43] και επομένως το ονομάζω λεξικό Goodreads. Το σκεπτικό δημιουργίας του είναι ίδιο με αυτό των λεξικών του imdb και του Amazon/TripAdvisor, καθώς επίσης βρίσκεται και στο ίδιο αρχείο. Οπότε η διαδικασία για την απομόνωση του λεξικού και την εξαγωγή βαθμολογίας είναι ακριβώς ίδια και για λόγους οικονομίας χώρου δεν την επαναλαμβάνω.

Στον πίνακα που ακολουθεί παραθέτω τον τρόπο χρησιμοποίησης της λέξης ‘bad’ στις κριτικές που γίνονται πάνω σε βιβλία και υπάρχουν στη συγκεκριμένη ιστοσελίδα.

Word	Tag	Category	Count	Total	Corpus
bad/a	1	-0.5	2100	3419923	Goodreads
bad/a	2	-0.25	1956	3912625	Goodreads
bad/a	3	0	2780	6011388	Goodreads
bad/a	4	0.25	2298	10187257	Goodreads
bad/a	5	0.5	2119	16202230	Goodreads

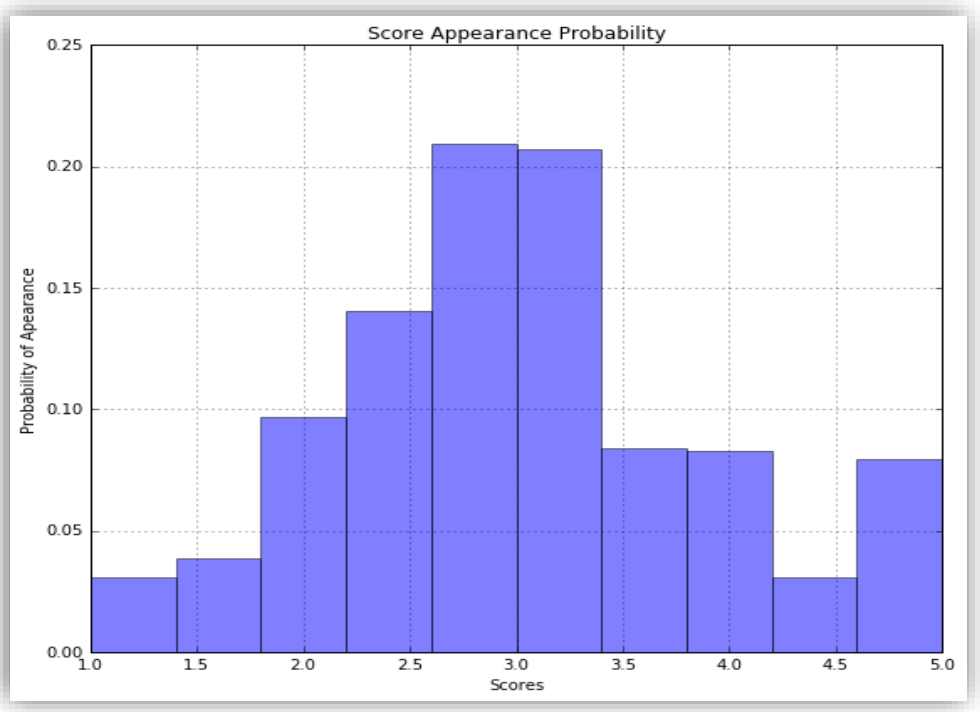
Πίνακας 11: Εγγραφές για τη λέξη 'bad' στο λεξικό του Goodreads

Στον πίνακα που παραθέτω παρακάτω δίνω δέκα (10) εγγραφές, όπως έκανα και στο λεξικό το imdb. Υπάρχουν τρεις στήλες στο λεξικό, στην πρώτη υπάρχει η λέξη της εγγραφής, στη δεύτερη η βαθμολογία αυτής και στην τρίτη η μέση τιμή των υφιστάμενων διπλοεγγραφών. Να σημειώσω ότι λέξη ‘bad’ την οποία χρησιμοποιώ ως σημείο αναφοράς το συγκεκριμένο λεξικό δεν υφίσταται ως διπλοεγγραφή.

Word	Expected Rate	Average of Duplicates
bad	2.357987	2.357987
badly	2.013399	2.013399
baffled	2.757086	2.757086
baffling	1.584345	1.584345
baggy	4	4
baked	2.093055	2.093055
balanced	3.187153	3.187153
bald	2.146235	2.146235
baldly	2.412892	2.412892
baleful	4	4

Πίνακας 12: Δέκα εγγραφές στην τελική έκδοση του λεξικού

Το λεξικό του Goodreads μετά το τέλος της επεξεργασίας διαθέτει 10,050 εγγραφές οι οποίες λαμβάνουν σκορ στο εύρος [1,5] και τιμές του σκορ δεν είναι ακέραιες. Η μέση τιμή για τα σκορ των εγγραφών είναι 3.036105 με διακύμανση ίση με 0.829019 και συνολικά υπάρχουν 5,286 μοναδικά σκορ.



Διάγραμμα 19: Πιθανότητα εμφάνισης διαθέσιμων σκορ στο λεξικό του Goodreads, ομαδοποιημένα σε 10 υποσύνολα

4.2.4 Χρήση λεξικού OpenTable

Το τελευταίο λεξικό αυτής της οικογένειας που μελετάω σε αυτή την εργασία δημιουργήθηκε από την ανάγνωση κριτικών σε εστιατόρια και υπάρχουν στην ιστοσελίδα του OpenTable [44]. Το συγκεκριμένο λεξικό στα πλαίσια αυτής της εργασίας το ονομάζω λεξικό OpenTable.

Το λεξικό του OpenTable βρίσκεται στο ίδιο αρχείο με τα υπόλοιπα τρία (3) λεξικά που έχω παρουσιάσει σε αυτό το υποκεφάλαιο και επομένως εφάρμοσα την ίδια τεχνική για να το απομονώνω και να θέσω σε κάθε εγγραφή του το κατάλληλο σκορ. Στον πίνακα που ακολουθεί παραθέτω τα σκορ που δίνει το λεξικό για τη λέξη του ‘bad’.

Word	**Tag**	**Category**	**Count**	**Total**	**Corpus**
bad/a	1	-0.5	1127	699695	OpenTable
bad/a	2	-0.25	2595	2507147	OpenTable
bad/a	3	0	2859	4207700	OpenTable

bad/a	4	0.25	2544	7789649	OpenTable
bad/a	5	0.5	1905	8266564	OpenTable

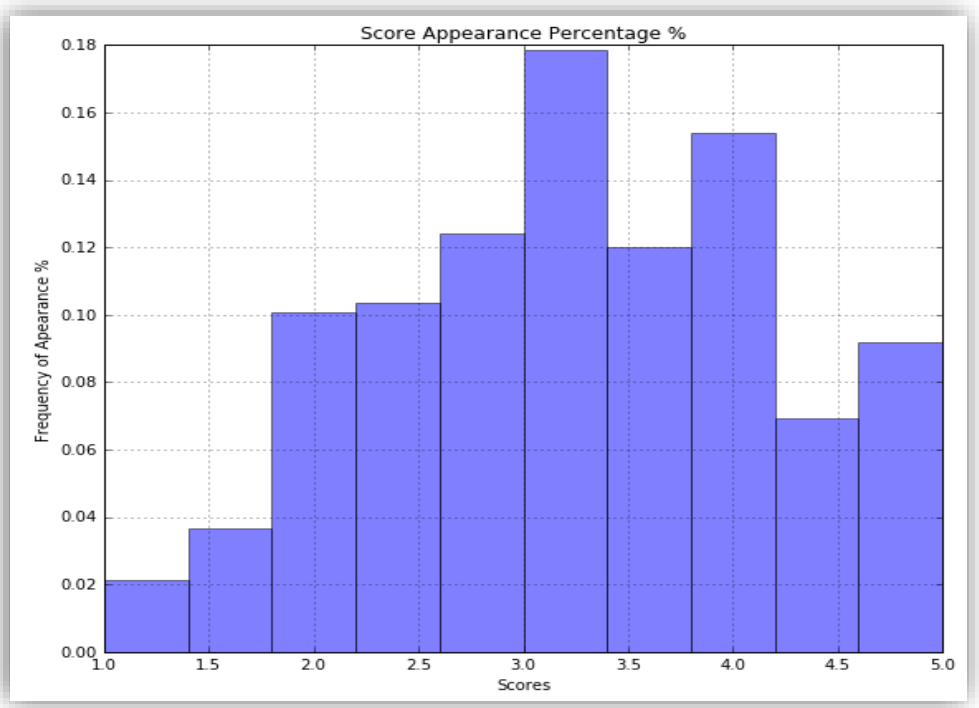
Πίνακας 13: Εγγραφές για τη λέξη 'bad' στο λεξικό του OpenTable

Μετά την επεξεργασία που πραγματοποιώ στο λεξικό του OpenTable, εξαγωγή σκορ και εκκαθάριση διπλοεγγραφών, έχουν απομείνει 7,575 από τις 7,724 εγγραφές. Δέκα (10) από αυτές τις εγγραφές παρουσιάζονται στον πίνακα που ακολουθεί.

Word	**Expected Rate**	**Average of Duplicates**
badly	1.865014	1.865014
baffled	3.760392	3.760392
baggy	5	5
bahamian	4	4
baked	2.82386	2.82386
balanced	3.796321	3.796321
bald-headed	2	2
bald	2.016125	2.016125
ball-shaped	5	5
balmy	3.712617	3.712617

Πίνακας 14: Δέκα εγγραφές στην τελική έκδοση του λεξικού

Η μέση τιμή για τα σκορ των εγγραφών που έχουν απομείνει είναι 3.252655 με διακύμανση ίση με 0.916501 και συνολικά υπάρχουν 4,001 μοναδικά σκορ. Οι συχνότητες με τις οποίες εμφανίζονται τα συγκεκριμένα σκορ στις εγγραφές του λεξικού φαίνονται στο παρακάτω διάγραμμα.



Πίνακας 15: Ποσοστό εμφάνισης διαθέσιμων σκορ στο λεξικό του OpenTable, ομαδοποιημένα σε 10 υποσύνολα

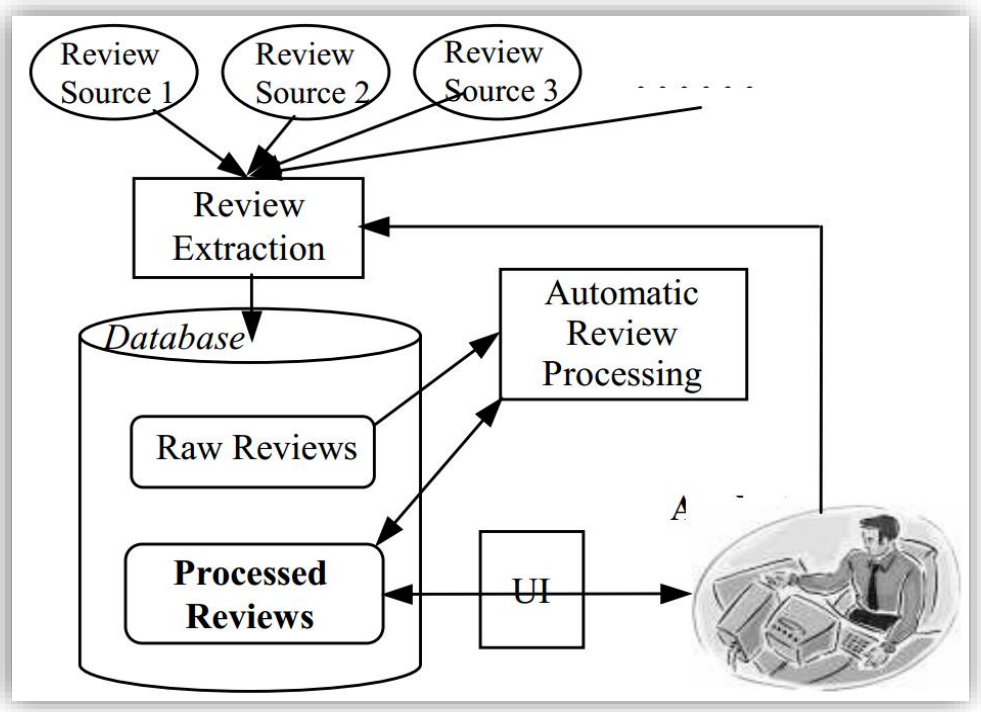
4.3 Χρησιμοποίηση Λεξικού Opinion Observer

Έχοντας ολοκληρώσει την μελέτη των λεξικών που προέρχονται από τέσσερις (4) συγκεκριμένες ιστοσελίδες (imdb, Amazon/TripAdvisor, Goodreads, Opentable) προχωράω στην μελέτη ενός ακόμη λεξικού. Το λεξικό που μελετάω σε αυτό το υποκεφάλαιο έχει δημιουργηθεί με σκοπό την κατηγοριοποίηση κριτικών από διαφορετικές ιστοσελίδες [45] [46].

Η βασική ιδέα πίσω από την δημιουργία του συγκεκριμένου λεξικού είναι η αυτόματη κατηγοριοποίηση των κριτικών σε θετικές και αρνητικές. Ο συγκεκριμένος συλλογισμός γεννήθηκε από την ανάγκη για γρήγορη ερμηνεία των δημοφιλών προϊόντων, τα οποία μπορούν να έχουν εκατοντάδες ή χιλιάδες κριτικές. Οι δημιουργοί του λεξικού

δημιούργησαν ένα ολοκληρωμένο λογισμικό οπτικοποίησης της ερμηνείας των κριτικών με το όνομα Opinion Observer [46], για αυτό το λόγο το συγκεκριμένο λεξικό στα πλαίσια αυτής της εργασίας το ονομάζω Opinion Observer.

Όσον αφορά την δημιουργία του λεξικού, αρχικά πραγματοποιήθηκε η εξόρυξη των κριτικών από διαφορετικές ιστοσελίδες, στη συνέχεια αυτές οι κριτικές επεξεργάστηκαν μέσα από εργαλεία εύρεσης μερών του λόγου, απομόνωσης χαρακτηριστικών προϊόντος, εύρεση συνωνύμων και δημιουργίας διάφορων κανόνων, προκειμένου να πραγματοποιηθεί σωστή κατηγοριοποίηση. Στη συνέχεια πρόσθεσαν και τη δυνατότητα χειροκίνητου ελέγχουν στις κριτικές που έχουν επεξεργαστεί και κατηγοριοποιηθεί. Για την καλύτερη κατανόηση του συστήματος που δημιουργήσαν και κατ' επέκταση του λεξικού που χρησιμοποιώ παραθέτω την παρακάτω εικόνα.



Εικόνα 2: Αρχιτεκτονική του λογισμικού Opinion Observer

Σημεία που πρέπει να τονιστούν είναι ότι το λεξικό του Opinion Observer έχει δημιουργηθεί για την κατηγοριοποίηση κριτικών και έχει επικεντρωθεί στην κατανόηση προτάσεων και όχι ολοκλήρου κειμένου και κατά την δημιουργία του δεν υπολογίστηκαν τρεις σημαντικοί παράγοντες:

- Αντιμετώπιση προτάσεων οι οποίες χρειάζονται ανάλυση αντωνυμιών (π.χ. 'it is quiet but powerful')
- Μη συμπερίληψη λέξεων που δεν είναι επίθετα, ως λέξεις που εκφράζουν συναίσθημα
- Μη προσδιορισμός βαρύτητας στις λέξεις που εκφράζουν συναίσθημα

Η δομή του λεξικού διαφέρει συγκριτικά με τα υπόλοιπα λεξικά που έχω μελετήσει. Όπως ανέφερα, στόχος των δημιουργών του λεξικού είναι να κατηγοριοποιήσουν μία κριτική προϊόντος ως θετική ή αρνητική, χωρίς να δίνουν διαφορετική βαρύτητα στις λέξεις, αλλά μόνο χροιά (θετική, αρνητική). Για αυτό το λόγο το λεξικό του Opinion Observer αποτελείται από δύο αρχεία, το πρώτο διαθέτει 2,006 εγγραφές με θετική χροιά και το δεύτερο 4,783 εγγραφές με αρνητική χροιά. Δεν δίνονται επιπλέον στοιχεία σε κάθε εγγραφή, παρά μόνο η λέξη. Επομένως το σκορ για κάθε λέξη είναι +1 αν υπάρχει στο αρχείο με τις λέξεις που διαθέτουν θετική χροιά ή -1 αν υπάρχει στο δεύτερο αρχείο.

Στον πίνακα που ακολουθεί τοποθετώ στην αριστερή στήλη 10 λέξεις με θετική χροιά και δέκα με αρνητική στην δεξιά στήλη.

Positive	**Negative**
good	backward
goodly	backwardness
goodness	backwood
goodwill	backwoods
goood	bad
gooood	badly
gorgeous	baffle
gorgeously	baffled
grace	bafflement
graceful	baffling

Πίνακας 16: Δέκα (10) εγγραφές με θετική και αρνητική χροιά στο λεξικό του Opinion Observer

4.4 Χρησιμοποίηση Λεξικού SentiWordNet

Το έβδομο στη σειρά λεξικό που μελετάω έχει δημιουργηθεί πάνω στο WordNet [47] [48], μία από τις μεγαλύτερες λεξιλογικές βάσεις δεδομένων, το οποίο ομαδοποιεί τις λέξεις με βάση την εννοιολογικής τους σημασία.

Για τη δημιουργία και ολοκλήρωση του λεξικού χρησιμοποιήθηκαν δύο διαφορετικές τεχνικές, σε δύο ξεχωριστά βήματα. Στο πρώτο βήμα χρησιμοποιήθηκε classifier με μερική επίβλεψη, κατά το οποίο αρχικά δόθηκαν 7 χαρακτηριστικά θετικές λέξεις και 7 αρνητικές, και μέσω των δυαδικών (binary) σχέσεων που υφίστανται με άλλες λέξεις (συνώνυμα/ανώνυμα) στο λεξικό του WordNet, αυτό το σύνολο επεκτάθηκε. Στην

συνέχεια αυτό το σύνολο, συν κάποιες ουδέτερες λέξεις, χρησιμοποιήθηκε ως training set και με τη χρήση classifier αξιολογήθηκαν όλες οι λέξεις του λεξικού.

Το δεύτερο βήμα στη δημιουργία του λεξικού είναι η εφαρμογή του αλγορίθμου Random Walk, για την επιβεβαίωση της κατηγορίας στην οποία ανήκει η λέξη. Ο συγκεκριμένος αλγόριθμος, λαμβάνει το λεξικό ως γράφημα, όπου μία λέξη *s1* συνδέεται με μία λέξη *s2*, όταν η πρώτη υπάρχει στην περιγραφή της δεύτερης. Έτσι μέσω επαναλήψεων εξάγεται το συμπέρασμα, ότι όσο περισσότερες θετικές (ή αρνητικές) λέξεις υπάρχουν στην περιγραφή μίας συγκεκριμένης λέξης, τότε είναι πιθανότερο η υπό εξέταση λέξη να είναι κι αυτή θετική (ή αντίστοιχα αρνητική).

Όσον αφορά την μορφή που έχει το λεξικό του SentiWordNet, η έκδοση 3.0 του λεξικού [49] [50], αυτή διαφέρει σε σχέση με τα υπόλοιπα λεξικά που έχω μελετήσει, διαθέτει συνολικά 5 στήλες και η πέμπτη στήλη είναι αυτή που το διαφοροποιεί σημαντικά από τα υπόλοιπα. Στην πρώτη στήλη υπάρχει το μέρος του λόγου στο οποίο ανήκει η εγγραφή, στη δεύτερη το μοναδικό ID της εγγραφής, στην τρίτη ένας αριθμός που υποδηλώνει το θετικό σκορ, η επόμενη στήλη δηλώνει το αρνητικό σκορ της εγγραφής, ακολουθεί η λέξη προς μελέτη και στην τελευταία στήλη υπάρχει λεξιλογική επεξήγηση της εγγραφής. Στον πίνακα που ακολουθεί παραθέτω πέντε (5) εγγραφές του λεξικού SentiWordNet.

POS	**ID**	**PosScore**	**NegScore**	**Word**	**SynsetTerms**
a	00001740	0.125	0	able#1	(usually followed by `to') having the necessary means or skill or know-how …
a	00002098	0	0.75	unable#1	(usually followed by `to') not having the necessary means or skill or know-how…
a	00002312	0	0	dorsal#2 abaxial#1	facing away from the axis of an organ or organism; …
a	00002527	0	0	ventral#2 adaxial#1	nearest to or facing toward the axis of an organ or organism; …
a	00002730	0	0	acroscopic#1	facing or on the side toward the apex
a	00002843	0	0	basiscopic#1	facing or on the side toward the base

Πίνακας 17: Πέντε (5) εγγραφές του λεξικού SentiWordNet

Μετά την λέξη ακολουθεί το σύμβολο της δίεσης(#) και ένας αριθμός. Ο αριθμός υποδηλώνει τη συχνότητα εμφάνισης της λέξης στο λεξικό, διότι μία λέξη ενδέχεται να έχει περισσότερες από μία ερμηνείες. Ο αριθμός που ακολουθεί μετά τη δίεση δεν είναι σταθερός για κάθε λέξη, αλλά αυξάνεται σε κάθε συνάντηση της λέξης. Δηλαδή στην πρώτη εμφάνιση της λέξης λαμβάνει την τιμή ένα (#1) στη δεύτερη τη τιμή δύο (#2) κ.ο.κ.

Επίσης υπάρχει το ενδεχόμενο σε μία γραμμή να υπάρχουν περισσότερες από μία λέξεις, οι οποίες είναι συνώνυμες και διαχωρίζονται απλά με δύο κενούς χαρακτήρες. Η τρίτη εγγραφή στον πίνακα ανήκει σε αυτή την κατηγορία. Να σημειώσω ότι το λεξικό είναι σε μορφή .txt, αλλά το παρουσιάζω εγώ σε μορφή πίνακα προκειμένου να γίνει ευκολότερο κατανοητό.

Ακόμη πρέπει να σημειώσω ότι για κάθε λέξη του λεξικού υπάρχει και η μετρική της ουδετερότητας (objective/ή neutrality). Η συγκεκριμένη μετρική δεν υφίσταται στο λεξικό, αλλά είναι ιδιαίτερο εύκολο να εξαχθεί, μέσω της συνάρτησης *obj = 1 – (pos + neg)*. Οι τρεις μετρικές κυμαίνονται στο εύρος [0,1] και το άθροισμα τους ισούται με τη μονάδα. Στην εργασία μου ως τελικό σκορ δεν χρησιμοποίησα τη μετρική του objectivity, αλλά αφαίρεσα από το θετικό σκορ το αρνητικό, έτσι ώστε όταν υπάρχει μόνο θετικό (ή αρνητικό) σκορ να είναι αυτό το τελικό σκορ ειδάλλως να είναι η διαφορά τους. Οι λέξεις με σκορ ίσο με το 0 είναι απόλυτα ουδέτερες, δηλαδή στην μετρική του objective έχουν σκορ ίσο με τη μονάδα.

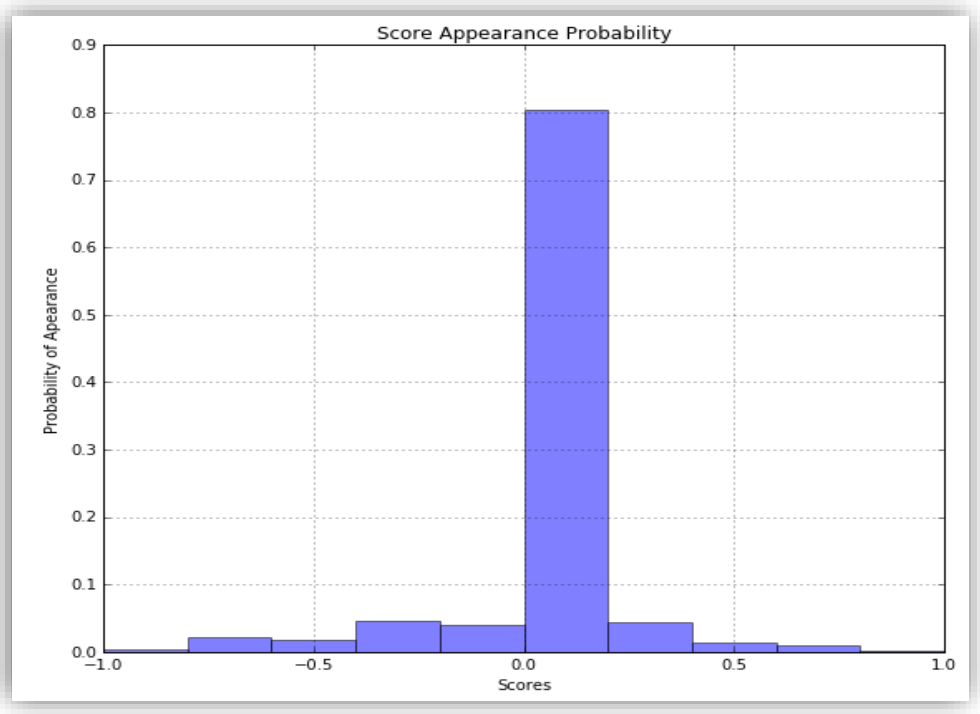
Το λεξικό με την αρχική του μορφή (αρχείο .txt, πεδία χωρίζονται με κενά) είναι ιδιαίτερα δύσκολο να συσχετιστεί με τη δομή της βάσης δεδομένων που έχω ήδη συλλέξει. Αρχικά αφαίρεσα τα πεδία από το λεξιλόγιο τα οποία δεν μπορούν να χρησιμοποιηθούν από τους classifiers και άφησα τη λέξη και το τελικό σκορ για κάθε εγγραφή.

Στις περιπτώσεις που μία εγγραφή (γραμμή) διαθέτει περισσότερες από μία λέξεις, τις έσπασα σε περισσότερες εγγραφές, έτσι ώστε κάθε εγγραφή να αποτελείται από μία λέξη και το τελικό σκορ της. Στην τελική μορφή το λεξικό διαθέτει δύο στήλες, στην πρώτη υπάρχει η λέξη και στη δεύτερη το σκορ τη λέξης. Συνολικά υπάρχουν 206940 εγγραφές, αλλά υπάρχουν και πολλές λέξεις οι οποίες υπάρχουν παραπάνω από μία φορά στο λεξικό, οι λέξεις που υπάρχουν, αφαιρώντας τις διπλοεγγραφές, είναι 147790.

Όσον αφορά το πρόβλημα με τις διπλοεγγραφές δημιούργησα δύο εναλλακτικές λύσεις. Στην πρώτη υπάρχουν διπλοεγγραφές στο λεξικό και η αναζήτηση της λέξης σταματάει στην εύρεση της πρώτης εγγραφής, όπου το σκορ της λέξης δεν ορίζεται από κάποιον κανόνα. Στην εναλλακτική λύση, απέρριψα τις διπλές εγγραφές από το λεξικό και κράτησα την εγγραφή με το υψηλότερο απόλυτο σκορ. Παραδείγματος χάριν, η λέξη greedy συναντάται 3 φορές, με σκορ 0, 0, 0.125 στην πρώτη περίπτωση η λέξη λαμβάνει τιμή 0, ενώ στη δεύτερη 0.125.

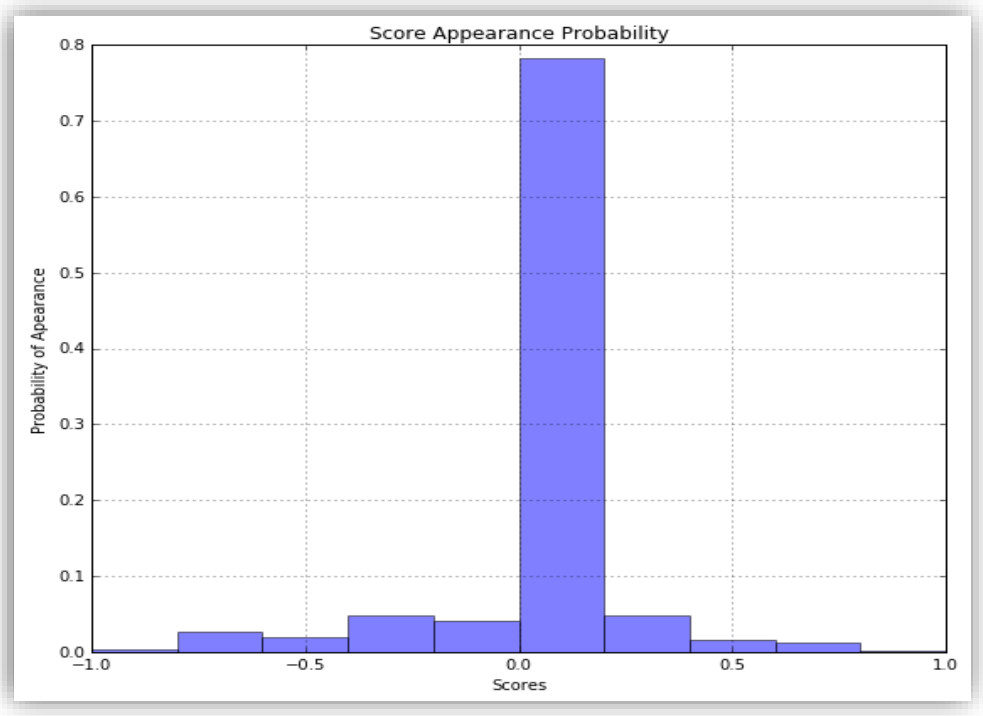
Ο λόγος που δημιουργώ δύο εναλλακτικά λεξικά είναι η διερεύνηση της συμπεριφοράς των classifiers σε αυτές τις δύο περιπτώσεις. Όπως παρατηρήθηκε στα προηγούμενα λεξικά, έτσι και στο λεξικό του SentiWordNet η πλειοψηφία των εγγραφών λαμβάνει μηδενική τιμή, κάνοντας την σωστή κατηγοριοποίηση τους ιδιαίτερα δύσκολη.

Το αρχείο έχει δύο στήλες, στην πρώτη στήλη υπάρχει η λέξη που μελετάται και στην δεύτερη το σκορ που αποδίδεται σε αυτή. Στην πρώτη έκδοση του λεξικού συνολικά υπάρχουν 206,940 εγγραφές, στις οποίες αποδίδεται σκορ στο εύρος [-1, 1]. Οι εγγραφές έχουν μέση τιμή -0.011491 και τυπική απόκλιση 0.034453 με 28 μοναδικές τιμές, αλλά να σημειώσω ότι οι λέξεις που βαθμολογούνται με μηδέν (0) είναι το 78% του συνόλου. Η κατανομή των σκορ που αποδίδεται στις λέξεις απεικονίζεται στο διάγραμμα που ακολουθεί.



Διάγραμμα 20: Πιθανότητα εμφάνισης διαθέσιμων σκορ στο λεξικό του SentiWordNet στην πρώτη έκδοση που δημιούργησα, ομαδοποιημένα σε 10 υποσύνολα

Στην δεύτερη έκδοση του λεξικού υπάρχουν 147,791 εγγραφές με μέση τιμή ίση με -0.011491 και τυπική απόκλιση 0.04039 με 26 μοναδικές τιμές, οι λέξεις που βαθμολογούνται με μηδέν (0) αποτελούν το 74% του συνόλου. Η κατανομή των σκορ που αποδίδοντα στις λέξεις απεικονίζεται στο διάγραμμα που ακολουθεί.



Διάγραμμα 21: Πιθανότητα εμφάνισης διαθέσιμων σκορ στο λεξικό του SentiWordNet στη δεύτερη έκδοση που δημιούργησα, ομαδοποιημένα σε 10 υποσύνολα

Από τα χαρακτηριστικά των λεξικών, σε συνδυασμό με τα δύο διαγράμματα που παρέθεσα δεν είναι εμφανής κάποια μεγάλη διαφορά στις δύο εκδόσεις του λεξικού που δημιούργησα.

4.5 Χρησιμοποίηση Λεξικού Subjectivity

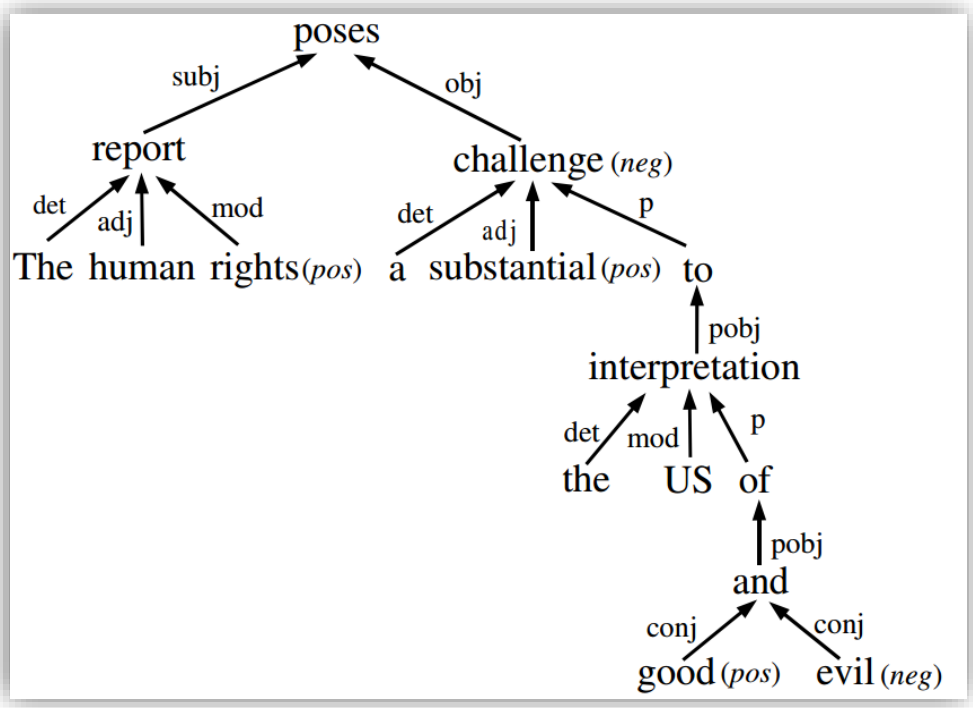
Το λεξικό που μελετώ σε αυτό το υποκεφάλαιο επικεντρώνεται στην εύρεση συναισθήματος στο φυσικό λόγο. Σύμφωνα με τη θεωρία πάνω στην οποία δημιούργησαν το λεξικό του το συναίσθημα στον γραπτό λόγο εκφράζεται μέσω λέξεων που συνδέονται με την υποκειμενικότητα (subjectivity) της άποψης. Για αυτό το λόγο στα πλαίσια αυτής της εργασίας το λεξικό το ονομάζω λεξικό Subjectivity

Η καινοτομία στην έρευνα τους για την εύρεση συναισθήματος στον γραπτό λόγο έγκειται στον τρόπο με τον οποίο θεωρούν ότι μία έκφραση συνδέεται με ένα συναίσθημα. Πρώτα αποφασίζεται αν η συγκεκριμένη έκφραση είναι ουδέτερη ή συναισθηματικά φορτισμένη και έπειτα με ποιο συναίσθημα συνδέεται. Μέσω αυτής της τεχνικής διαχωρίζεται το συναίσθημα στην λέξη όταν αυτή κρίνεται αυτόνομα με το νόημα της μέσα στο πλαίσιο λόγου που χρησιμοποιείται. Μέσω της συγκεκριμένης τεχνικής πραγματοποιούν μετρήσεις στις οποίες αποδεικνύεται ότι ο συνδυασμός των τεχνικών παρουσιάζει υψηλότερα σκορ, συγκριτικά με την εφαρμογή των επιμέρους τεχνικών ανεξάρτητα.

Από την μελέτη για την αξιοπιστία των σχολιαστών(annotators), το 82% των φράσεων θεωρήθηκαν ότι περιέχουν κάποια υποκειμενική έκφραση και από τους δύο annotators που χρησιμοποιήθηκαν. Το 18% των υποκειμενικών εκφράσεων, χαρακτηρίστηκαν από ένα τουλάχιστον annotator ως αβέβαιες (uncertain). Αφαιρώντας αυτές τις προτάσεις το ποσοστό συμφωνίας μεταξύ των annotators έφτασαν το 90%.

Ως βάση δεδομένων χρησιμοποιήθηκαν αρχεία που υπάρχουν στη βάση δεδομένων του MPQA [51]. Στη μελέτη της βάσης δεδομένων έχουν εξαχθεί κάποιες παρατηρήσεις οι οποίες πρέπει να σημειωθούν. Χρησιμοποιήθηκαν συνολικά 15991 υποκειμενικές εκφράσεις, το 28% των οποίων δεν περιέχει κάποια υποκειμενική έκφραση, το 25% περιέχει μόνο μία και το 47% τουλάχιστον δύο. Από τις 4247 προτάσεις που περιέχουν τουλάχιστον δύο υποκειμενικές εκφράσεις, το 17% περιέχει θετικές καθώς και αρνητικές εκφράσεις και το 62% περιέχει εκφράσεις που χαρακτηρίζονται από διαφορετική συναισθηματική φόρτιση.

Για να γίνει πιο εύκολα κατανοητή η βασική ιδέα πίσω από τη δημιουργία του λεξικού παραθέτω μία εικόνα που εξηγεί πως αξιολογούνται κάθε μία από τις λέξεις της πρότασης *‘The human rights report poses a substantial challenge to the US interpretation of good and evil’*.



Εικόνα 3: Το δέντρο εξάρτησης για την πρόταση 'The human rights report poses a substantial challenge to the US interpretation of good and evil'. Η χροιά των λέξεων σημειώνεται μέσα σε παρένθεση

Όσον αφορά τη συναισθηματική φόρτιση που εκφέρουν οι λέξεις όταν αυτές κρίνονται αυτόνομα το 92.8% αυτών χαρακτηρίζονται ότι έχουν συναισθηματική φόρτιση, 33.1% θετική και 59.7% αρνητική. Μόλις 0.3% χαρακτηρίζονται ως αμφισήμαντες στην εκφορά συναισθήματος και 6.9% ως ουδέτερες.

Το λεξικό διαθέτει 8,222 εγγραφές με την κάθε εγγραφή να έχει 6 στήλες, στην πρώτη αναγράφεται ο τύπος της λέξης, στη δεύτερη το μήκος της, στην τρίτη η ίδια η λέξη, στην τέταρτη το μέρος του λόγου στο οποίο ανήκει η λέξη, στην πέμπτη στήλη υπάρχει μία δυαδική μεταβλητή που δείχνει αν η λέξη είναι παράγωγο και στη τελευταία στήλη παρουσιάζεται η συναισθηματική φόρτιση της λέξης. Στον πίνακα που ακολουθεί παραθέτω πέντε (5) εγγραφές του λεξικού.

Type	**Length**	**Word**	**POS**	**Stemmed**	**Priorpolariy**
weaksubj	1	backbone	noun	n	positive
strongsubj	1	backward	adj	n	negative
Strongsubj	1	backwardness	noun	n	negative
Strongsubj	1	bad	adj	n	negative
Strongsubj	1	badly	adj	n	negative

Πίνακας 18: Πέντε (5) εγγραφές του λεξικού Subjectivity

Από τις παραπάνω εγγραφές του λεξικού γίνεται κατανοητό ότι δεν δίνεται κάποιο σκορ για να μπορέσω να εφαρμόσω αλγορίθμους κατηγοριοποίησης στη συνέχεια, έτσι πρέπει να αναθέσω σκορ στην κάθε εγγραφή. Η ανάθεση του σκορ λαμβάνει υπόψιν δύο

πεδία. Το πρώτο είναι ο τύπος της λέξης και το δεύτερο είναι η συναισθηματικότητα. Θέτω σκορ ίσο με το μηδέν στις εγγραφές που έχουν ουδέτερη συναισθηματικότητα, στις εγγραφές με αδύναμη υποκειμενικότητα σκορ ίσο με +1 αν υπάρχει θετική χροιά και -1 αν υπάρχει αρνητική και για της εγγραφές με δυνατή υποκειμενικότητα θέτω +2 σε αυτές που έχουν θετική χροιά και -2 στις λέξεις με αρνητική χροιά.

Type	**Priorpolarity**	**Score**
Weak	Neutral	0
Strong	Neutral	0
Weak	Positive	1
Strong	Positive	2
Weak	Negative	-1
Strong	Negative	-2

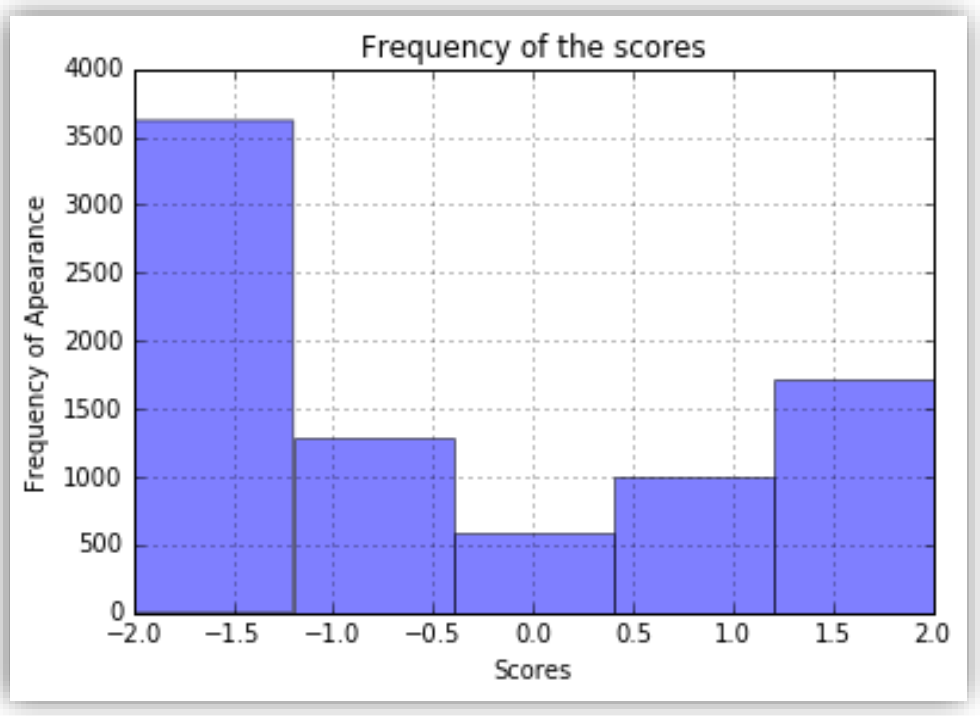
Πίνακας 19: Ο τρόπος ανάθεσης σκορ στις εγγραφές του λεξικού Subjectivity

Οπότε στο λεξικό προσθέτω ακόμα ένα πεδίο και οι πέντε εγγραφές που παρέθεσα παραπάνω παίρνουν τα σκορ που δείχνει ο πίνακας στη συνέχεια.

Type	**Length**	**Word**	**POS**	**Stemmed**	**Priorpolariy**	**Score**
weaksubj	1	backbone	noun	n	positive	1
strongsubj	1	backward	adj	n	negative	-2
Strongsubj	1	backwardness	noun	n	negative	-2
Strongsubj	1	bad	adj	n	negative	-2
Strongsubj	1	badly	adj	n	negative	-2

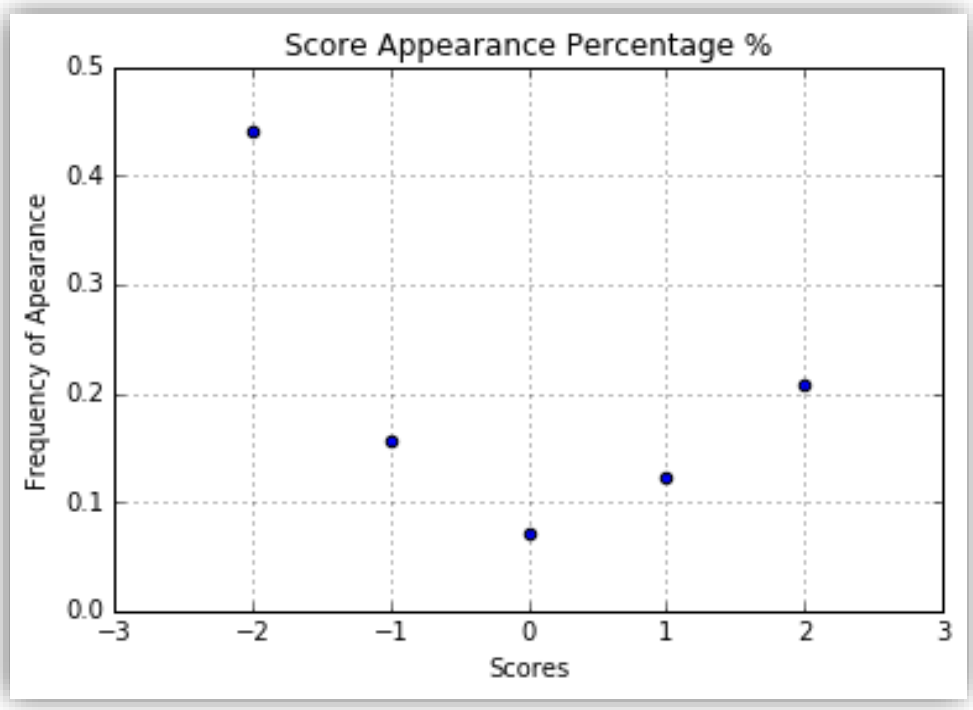
Πίνακας 20: Πέντε (5) εγγραφές του λεξικού Subjectivity μετά την ανάθεση σκορ

Οπότε το λεξικό του Subjectivity λαμβάνει για σκορ ακέραιες τιμές στο εύρος [-2,2]. Μετά το λεξικό του AFINN είναι το δεύτερο λεξικό το οποίο λαμβάνει ακέραιες τιμές, αλλά δεν έχουν καμία ομοιότητα στην κατανομή, καθώς το λεξικό του AFINN έχει κανονική κατανομή, ενώ το λεξικό του Subjectivity θέτει στην πλειοψηφία των εγγραφών, συγκεκριμένα το 40%, σκορ ίσο με -2. Ακολουθούν τα διαγράμματα που δείχνουν την κατανομή των τιμών στο λεξικό του Subjectivity.



Διάγραμμα 22: Πλήθος εμφάνισης των διαθέσιμων σκορ στο λεξικό του Subjectivity

Η κατανομή των τιμών στο λεξικό του Subjectivity δεν θυμίζει κάποια κατανομή από τα λεξικά που έχω μελετήσει ως τώρα. Υπάρχει συσσώρευση τιμών στο σκορ του -2, την μικρότερη δυνατή τιμή, ενώ παράλληλα σκορ ίσο με μηδέν δίνεται σε λίγες εγγραφές. Το λεξικό έχει έντονη αρνητική χροιά και συγκριτικά με τα υπόλοιπα λεξικά έχει λίγες λέξεις με ουδέτερη χροιά. Η κατανομή του σίγουρα προκαλεί ενδιαφέρον και η σωστή πρόβλεψη μέσω classifiers θα είναι ιδιαίτερα δύσκολη.



Διάγραμμα 23: Πιθανότητα εμφάνισης % των διαθέσιμων σκορ στο λεξικό του Subjectivity

4.6 Χρησιμοποίηση Λεξικού inquirer

Το τελευταίο λεξικό που μελετάω είναι δημιούργημα του ίδιου ατόμου που δημιούργησε τα λεξικά των imdb, Amazon/TripAdvisor, Goodreads και OpenTable, αλλά το δημιουργήθηκε βασισμένο σε διαφορετική ιδέα. Είναι υλοποίηση [52] [53] ενός αλγορίθμου [54], ο οποίος ως κεντρική ιδέα έχει την όσο το δυνατόν μεγαλύτερη αξιοποίηση ήδη γνωστών δεδομένων (a priori).

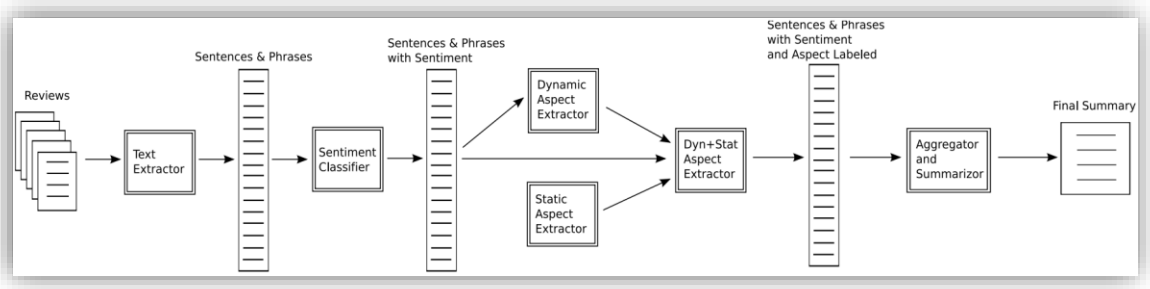
Το πρώτο βήμα του αλγορίθμου είναι η συλλογή δεδομένων, στην αρχική τους ιδέα τα δεδομένα προέρχονταν από κριτικές σε εστιατόρια και ξενοδοχεία. Τα δεδομένα αυτά χρησιμοποιούνται για τη δημιουργία ενός συνόλου που να συσχετίζεται με τις βαθμολογίες

για κάθε επιμέρους τμήμα της αρχικής κριτικής. Ακόμη αυτά τα δεδομένα είναι υποψήφια για την τελική αξιολόγηση, η οποία αποτελείται από ελεγμένα σωστές πληροφορίες.

Το δεύτερο στάδιο του αλγορίθμου είναι η κατηγοριοποίηση των προτάσεων ως θετικές, αρνητικές ή ουδέτερες. Το μοντέλο που χρησιμοποιήθηκε για την κατηγοριοποίηση των προτάσεων είναι ένα υβριδικό μοντέλο το οποίο χρησιμοποίει προσέγγιση μέσω λεξικού και αλγορίθμους μηχανικής μάθησης. Στην παρουσίαση του αλγορίθμου έδειξαν ότι βελτιώνεται η επιτυχία της κατηγοριοποίησης όταν αξιοποιείται όχι μόνο το περιεχόμενο της πρότασης αλλά και γενικές πληροφορίες που δίνονται από τον χρήστη, όπως μία γενική βαθμολογία στην κριτική.

Το τελευταίο βήμα του αλγορίθμου είναι η εξαγωγή των επιμέρους χαρακτηριστικών μέσα από τις κριτικές. Για τη συγκεκριμένη διεργασία εφαρμόζεται μία υβριδική μέθοδος, η οποία συνδυάζει δεδομένα τα οποία συλλέχθηκαν μέσα από το κείμενο προς επεξεργασία, δυναμική μέθοδος, αλλά και δεδομένα από τα οποία έχουν εξαχθεί από δεδομένα τα οποία έχουν αξιολογηθεί χειροκίνητα ή ανήκουν σε συγκεκριμένες ομάδες (τεχνική clustering).

Για να γίνει πιο εύκολα κατανοητή η αρχιτεκτονική του συστήματος παραθέτω την παρακάτω εικόνα.



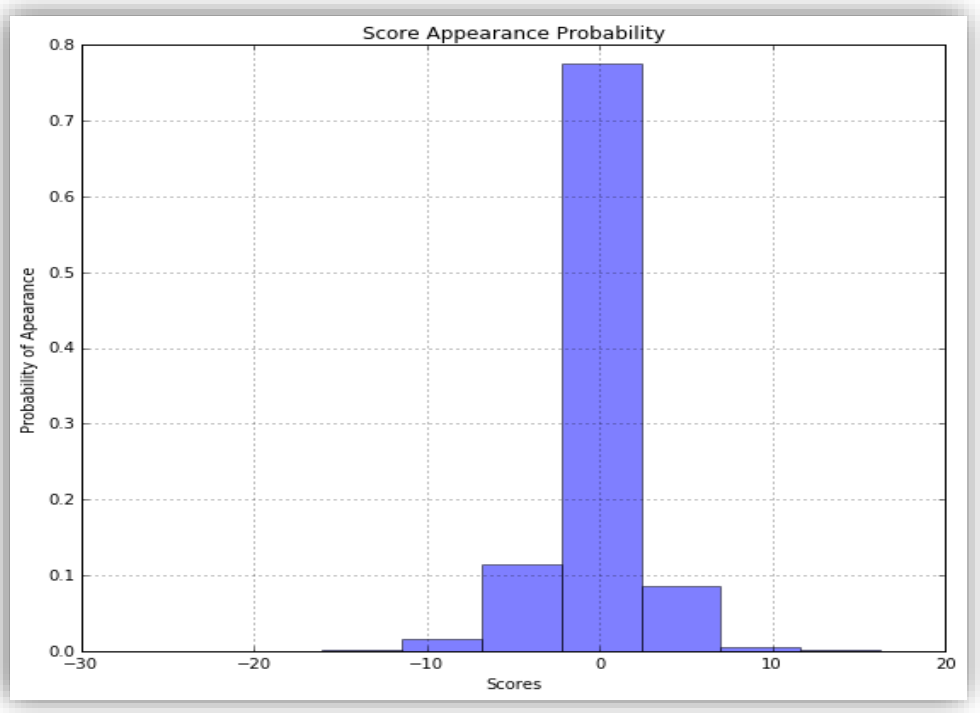
Εικόνα 4: Επισκόπηση του αλγορίθμου πάνω στον οποίο βασίστηκε το λεξικό του inquirer *[54]*

Για τη υλοποίηση του αλγορίθμου χρησιμοποιήθηκαν ως αρχικές λέξεις (seeds) του λεξικού 20 λέξεις με αρνητική χροιά, 47 με θετική και 293 με ουδέτερη. Εκτελώντας τον αλγόριθμο για επέκταση του αρχικού λεξικού με τη χρήση συνωνύμων και αντώνυμων δημιουργήθηκε λεξικό με 5,705 θετικές λέξεις και με 6,605 αρνητικές λέξεις. Η χρησιμότητα του αλγορίθμου δεν είναι η γνώση ενός καινούργιου λεξικού, αλλά η γνώση για τη σχέση ανάμεσα σε λέξεις με την ίδια πολικότητα (sentiment propagation).

Το λεξικό διαθέτει 156,585 εγγραφές. Η κάθε εγγραφή διαθέτει τρία πεδία, το πρώτο είναι η λέξη, το δεύτερο πεδίο δηλώνει το μέρος του λόγο στο οποίο ανήκει η λέξη και το τρίτο πεδίο είναι το σκορ που αποδίδεται στη λέξη. Η δομή του λεξικού είναι απλή, παρόμοιας με αυτής που συνάντησα στο λεξικό του AFINN και δεν απαιτεί ιδιαίτερη επεξεργασία για να γίνει χρήση του.

Το λεξικό διαθέτει πολλές εγγραφές, μόνο το λεξικό του SentiWordNet διαθέτει περισσότερες, αλλά διαθέτει και πολλές εγγραφές οι οποίες βαθμολογούνται με 0. Αφαιρώντας τις μηδενικές εγγραφές και κρατώντας τη μεγαλύτερη τιμή από τις διπλοεγγραφές, το λεξικό διαθέτει 8698 εγγραφές με μέση τιμή των σκορ να είναι ίση με -0.113891 και η διακύμανση ίση με 5.38880534103. Ακόμη υπάρχουν 1,673 μοναδικές

τιμές, υπάρχουν τόσες πολλές μοναδικές τιμές, γιατί ορίζονται με ακρίβεια 7^{ου} δεκαδικού ψηφίου. Η κατανομή των σκορ που αποδίδεται στις λέξεις απεικονίζεται στο διάγραμμα που ακολουθεί.



Διάγραμμα 24: Πιθανότητα εμφάνισης διαθέσιμων σκορ στο λεξικό του inquirer

Το διάγραμμα για την πιθανότητα εμφάνισης των σκορ δίνει μία αρκετά κατατοπιστική εικόνα για το σκορ που αποδίδεται στις εγγραφές του λεξικού του inquirer. Υπάρχει πολύ μεγάλη συγκέντρωση, ανώτερη του 75% στο εύρος [-2.5, 2.5].

5. Τεχνικές Μηχανικής Μάθησης

Σε αυτό το σημείο της εργασίας μου έχω πραγματοποιήσει τη συλλογή, την προεπεξεργασία και την οπτικοποίηση των δεδομένων και έχω περιγράψει τα λεξικά που έχω χρησιμοποιήσει για την αξιολόγηση των δεδομένων που έχω συλλέξει. Σε αυτό το κεφάλαιο παραθέτω τις βασικές αρχές για τις τεχνικές μηχανικής μάθησης και παρουσιάζω τον τρόπο με τον οποίο θα τις χρησιμοποιήσω για να ελέγξω την αποτελεσματικότητα των λεξικών στην αξιολόγηση των δεδομένων που έχω συλλέξει.

Ο όρος της μηχανικής μάθησης έχει δοθεί σε κάποιους γενικευμένους αλγορίθμους, οι οποίοι καθιστούν δυνατή την επεξεργασία ενός προβλήματος από έναν υπολογιστή, απλώς εξετάζοντας τα δεδομένα. Αυτοί οι αλγόριθμο έρχονται σε αντίθεση με την κλασσική έννοια του προγραμματισμού, όπου ο προγραμματιστής παρέχει λεπτομερείς οδηγίες για την εκπλήρωση ενός προβλήματος. Με άλλα λόγια μηχανική μάθηση είναι η ικανότητα ενός υπολογιστή να βρίσκει τη λύση σε ένα πρόβλημα χωρίς να έχει προγραμματιστεί ειδικά για αυτό, αλλά έχοντας εκπαιδευτεί σε κάποια δεδομένα.

Με άλλα λόγια τα δεδομένα είναι υπεύθυνα κι όχι ο αλγόριθμος για το αποτέλεσμα του προβλήματος. Το πρόβλημα θα μπορούσε να είναι οτιδήποτε, τα δεδομένα που εισάγουμε σε αυτό είναι αυτά που δημιουργούν την εκάστοτε λύση.

Ο πιο συνηθισμένος διαχωρισμός των αλγορίθμων είναι ανάλογα με το είδος των δεδομένων που δέχονται ή την ανταπόκριση που προσφέρει το κάθε σύστημα εκπαίδευσης. Οι τρεις πιο σύνηθες κατηγορίες είναι:

- Supervised learning: Ο αλγόριθμος δέχεται δεδομένα εισόδου και επιθυμητά αποτελέσματα. Ο στόχος είναι να δημιουργηθεί ένας κανόνας, ο οποίος ορίζει κάποια δεδομένα εισόδου σε κάποιες συγκεκριμένες εξόδους.
- Unsupervised learning: Δεν δίνονται επιθυμητά αποτελέσματα εξόδου, αφήνοντας τον αλγόριθμο να βρει κάποιο μοτίβο στα δεδομένα εισόδου. Πολλές φορές όταν εφαρμόζεται unsupervised learning, ο στόχος της ανάλυσης είναι η εύρεση μοτίβων στα δεδομένα.
- Reinforcement learning: Το λογισμικό αλληλοεπιδρά σε ένα δυναμικό περιβάλλον μέσα στο οποίο πρέπει να πραγματοποιήσει μία συγκεκριμένη ενέργεια, χωρίς ο προγραμματιστής να έχει δηλώσει ρητά πότε αυτή η ενέργεια ολοκληρώνεται.

Σε αυτή την εργασία εφαρμόζω μόνο τεχνικές supervised learning. Ο λόγος είναι ότι έχω στην κατοχή μου την επιθυμητή έξοδο, σκορ λεξικού, και θέλω να έχω μετρικές για

την αποδοτικότητα του συγκεκριμένου λεξικού. Καθώς επίσης εξετάζεται και η ύπαρξη σχέσης μεταξύ των διαφορετικών πεδίων που υπάρχουν στη βάση δεδομένων μου.

Οι αλγόριθμοι που χρησιμοποιώ είναι υλοποιημένοι σε Python και παρέχονται από τη βιβλιοθήκη του scikit-learn [55]. Η βιβλιοθήκη παρέχει υλοποιήσεις των περισσοτέρων αλγόριθμών μηχανικής μάθησης και έχει δημιουργηθεί χρησιμοποιώντας SciPy [56] [57], NumPy [58] και matplotlib [59]. Το scikit-learn αποτελεί μέρος της πλατφόρμας Anaconda.

Ο λόγος που επέλεξα να χρησιμοποιήσω τη συγκεκριμένη βιβλιοθήκη είναι ότι είναι ανοιχτού κώδικα, γραμμένη σε Python, παρέχει καλό documentation και έχει μεγάλη κοινότητα. Στην ακαδημαϊκή κοινότητα είναι ιδιαίτερα διαδομένο το Weka [60] [61] ως βιβλιοθήκη για τεχνικές μηχανικής μάθησης, λόγω του γραφικού περιβάλλοντος που προσφέρει.

Το Weka όμως δεν προσφέρει documentation το οποίο είναι εύκολο κατανοητό και είναι σχεδόν αδύνατο κάποιος να προσθέσει καινούργιους ή τροποποιημένους αλγορίθμους. Τέλος η επεξεργασία φυσικής γλώσσας απαιτεί μεγάλη και συνεχή προεπεξεργασία η οποία επιτυγχάνεται πιο εύκολα χρησιμοποιώντας μία ευρέως χρησιμοποιούμενη δομή δεδομένων, όπως τα Pandas, αντί διαφόρων πειραματισμών σε γραφικό περιβάλλον.

Τέλος, η διεπαφή που προσφέρει το scikit [62] για τους διάφορους αλγορίθμους είναι σχετικά απλή και η διαδικασία που πρέπει να ακολουθηθεί πριν εισαχθούν τα δεδομένα στον εκάστοτε αλγόριθμο είναι σύντομη και πάντοτε η ίδια.

5.1 Classification

Ως classification (κατηγοριοποίηση) ορίζεται η διαδικασία ανάθεσης εγγραφών σε ομάδες με κοινά χαρακτηριστικά. Ο προγραμματιστής έχει θέσει τις διαφορετικές ομάδες και ο αλγόριθμος προσπαθεί να ταξινομήσει τις διάφορες εγγραφές στις υπάρχουσες ομάδες. Το πιο τυπικό παράδειγμα classification είναι τα λογισμικά ανίχνευσης ανεπιθύμητης αλληλογραφίας(spam filter). Το λογισμικό έχει δύο κατηγορίες, επιθυμητό και ανεπιθύμητο, και η λειτουργία του είναι να τοποθετήσει τα νέα εισερχόμενα μηνύματα, εγγραφές, σε μία από τις δύο κατηγορίες.

5.1.1 K-Nearest Neighbors

Η λογική πίσω από τον αλγόριθμο του K-Nearest Neighbors (Κ-Πλησιέστεροι Γείτονες -ΚΝΝ) [63] είναι ότι σχεδόν όλα τα δεδομένα μπορούν να μοντελοποιηθούν σε διαφορετικές κατηγορίες, αν υπάρχει σωστή εστίαση στα κατάλληλα πεδία.

Το πρώτο βήμα του του ΚΝΝ είναι να αποθηκεύσει απλά τα δεδομένα που έχω στην κατοχή μου. Έπειτα, στην προσπάθεια κατηγοριοποίηση μίας νέας εγγραφής, βρίσκει τις πλησιέστερες 'Κ' εγγραφές στο δείγμα. Τα πεδία των δειγμάτων πρέπει να αριθμητικά έτσι ώστε να έχει νόημα το πλησιέστερες ως προσδιοριστικό των εγγραφών. Η μέτρηση της απόσταση μπορεί να γίνει με διαφορετικές μετρικές, η πιο διαδεδομένη είναι η Ευκλείδεια απόσταση.

Ένα ιδιαίτερο χαρακτηριστικό των supervised αλγορίθμων γενικά, κι όχι μόνο του ΚΝΝ, είναι η οριοθέτηση ορίων. Τα όρια των κατηγοριών δεν είναι πάντοτε σφαιρικά, αλλά μπορούν να πάρουν διαφορετικά σχέδια, από γραμμικά μέχρι πολύπλευρα με άνισες πλευρές και διαφορετικές γωνίες. Για το αλγόριθμο του ΚΝΝ ισχύει ο γενικός κανόνας ότι όσο μεγαλύτερο είναι το Κ, που ορίζει το πλήθος των γειτονικών εγγραφών, τόσο πιο στρωτά γίνονται τα όρια με λιγότερο θόρυβο. Πρέπει να σημειωθεί όμως ότι όσο μεγαλύτερο είναι το Κ, ο αλγόριθμος γίνεται όλο και λιγότερο ευαίσθητος σε τοπικές διακυμάνσεις, από τη στιγμή που λαμβάνονται υπόψιν πολλές περισσότερες εγγραφές.

5.1.2 Decision Trees

Τα Decision Trees (Δέντρα Απόφασης) [64] είναι μία supervised τεχνική μηχανικής μάθησης, η οποία είναι ικανή να προβλέψει την κατηγορία στην οποία ανήκει μία εγγραφή. Αυτή η πρόβλεψη επιτυγχάνεται μέσω της εξέτασης των πιθανοτικών αποτελεσμάτων των εγγραφών.

Τα δέντρα απόφασης κατηγοριοποιούν τα δεδομένα πιθανοτικά χρησιμοποιώντας ως μετρική την εντροπία, η οποία μπορεί να θεωρηθεί ως 'καθαρότητα' των δεδομένων. Τα δέντρα απόφασης αναζητούν για ένα σύνολο κανόνων/ερωτήσεων, σύμφωνα με το οποίο θα πραγματοποιηθεί η πιο γρήγορη κατηγοριοποίηση των δεδομένων. Όσο πιο ορθοί είναι οι κανόνες, τόσο πιο γρήγορα ο αλγόριθμος θα καταλήξει στη λύση. Τα δέντρα απόφασης διαθέτουν κάποια χαρακτηριστικά που τα κάνουν μοναδικά:

- Είναι δομημένα όπως ένα διάγραμμα ροής. Διαθέτουν έναν αρχικό κόμβο και μπορούν να διαθέτουν ένα ή παραπάνω φύλα και εσωτερικούς κόμβους.
- Κάθε κόμβος αντιπροσωπεύει έναν κανόνα.

- Κάθε διακλάδωση συνδέει το κόμβο-πατέρα με τον κόμβο-παιδί. Ακόμη δείχνει το αποτέλεσμα στο ερώτημα του κόμβου-πατέρα
- Κάθε φύλο του δέντρου αντιπροσωπεύει μία κατηγορία.

Τα δέντρο απόφασης μοντελοποιούν τις συνεχόμενες πιθανές πράξεις βασιζόμενα στην πιθανότητα εμφάνισης, το κόστος του μονοπατιού και το κέρδος της πληροφορίας που θα κερδίζουν. Ο στόχος του αλγορίθμου είναι η μεγιστοποίηση της συνολικής ομοιογένειας της κάθε κατηγορίας, η οποία είναι αποθηκευμένη στα φύλλα του δέντρου.

5.1.3 Random Forest

Ο αλγόριθμος του Random Forest (Τυχαία Δάση) [65] [66], διορθώνει τη μοναδική αδυναμία των δέντρων απόφασης. Στα δέντρα απόφασης αν υπάρξει πληθώρα ερωτημάτων-κόμβων, θα οδηγήσει σε μάθηση σε βάθος (deep learning) με λάθος μοτίβα και αποδοχή outliers στα δεδομένα. Σε αυτή την περίπτωση θα υπάρξει υπερκάλυψη δεδομένων, οδηγώντας σε εξαιρετική ανάκληση δεδομένων (data recall), αλλά με πολύ φτωχές επιδόσεις σε πρόβλεψη δεδομένων.

Τα δέντρα απόφασης υπάρχει πιθανότητα να μην λειτουργούν σωστά εξαιτίας των outliers και το μήκος και πλάτος των δεδομένων. Έτσι, ο αλγόριθμος δεν βασίζεται σε ένα δέντρο απόφασης, αλλά σε ένα δάσος από διαφορετικά δέντρα απόφασης. Κάθε δέντρο στο Random Forest, ειδικεύεται σε μία ειδική περιοχή, αλλά συνεχίζει να έχει μία γενική γνώση για τις περισσότερες περιοχές.

Όπως κάθε αλγόριθμος έτσι κι ο random forest, χρειάζεται μία βάση δεδομένων ως είσοδο, καθώς κι ένα πεδίο σε αυτή τη βάση που θα ορίζει σε ποια κλάση ανήκει η κάθε εγγραφή. Ο random forest όμως χρησιμοποιεί δύο ειδικές τεχνικές που τον διαφοροποιούν, μία σε επίπεδο δέντρου και μία στο επίπεδο δάσους.

Ο συγκεκριμένος αλγόριθμος αντί να χρησιμοποιεί ολόκληρη τη βάση δεδομένων για κάθε δέντρο απόφασης, χρησιμοποιεί μία διαίρεση στα συνολικά δεδομένα. Με αυτό τον τρόπο, κάθε δέντρο εκπαιδεύεται σε μία ανεξάρτητη βάση δεδομένων και η διαφοροποίηση μεταξύ των δέντρων είναι ελεγχόμενη. Αυτή η τεχνική ονομάζεται tree bagging, ή bootstrap aggregating.

Η δεύτερη τεχνική που χρησιμοποιεί ο αλγόριθμος εφαρμόζεται σε επίπεδο δάσους. Κάθε κόμβος του δέντρου χρησιμοποιεί ένα σύνολο από συγκεκριμένα χαρακτηριστικά για κάθε διακλάδωση. Ο λόγος ύπαρξης αυτής της τεχνικής είναι ότι είναι πιθανό να υπάρχουν ένα ή περισσότερα πεδία που έχουν μεγάλη συσχέτιση με την μεταβλητή y, η οποία εκφράζει την κλάση της εγγραφής. Επιλέγοντας ένα τυχαίο δείγμα

χαρακτηριστικών, τέτοια πεδία δεν θα δημιουργούσαν τόσο πολλά δέντρα και θα υπήρχε μεγαλύτερη ποικιλία στα πεδία που εξετάζονται.

5.1.4 Logistic Regression

Ο αλγόριθμος του logistic regression [67] είναι, αντίθετα με ότι δηλώνει το όνομα του, είναι ένας αλγόριθμος classification. Συγκεκριμένα εγώ θα χρησιμοποιήσω τον multinomial logistic regression, ο οποίος γενικεύει τον αλγόριθμος του logistic regression σε πρόβλημα πολλών κλάσεων, με περισσότερες από δύο πιθανές εξόδους [68]. Ο συγκεκριμένος αλγόριθμος είναι γνωστός ακόμα ως polytomous LR [69], multiclass LR, softmax regression, multinomial logit, maximum entropy(MaxEnt) classifier, conditional maximum entropy model.

Ο multinomial logistic regression αποτελεί λύση στο πρόβλημα ταξινόμησης, όπου υποθέτει ότι ένας γραμμικός συνδυασμός των πεδίων της βάσης δεδομένων και κάποιοι παράμετροι σχετικοί με το πρόβλημα μπορούν να χρησιμοποιηθούν για να αποφασιστεί η πιθανότητά εμφάνισης κάθε πιθανής εξόδου.

Οι πιθανότητες που περιγράφουν τα πιθανά αποτελέσματα του αλγορίθμου προκύπτουν χρησιμοποιώντας τη συνάρτηση logistic function. Η κεντρική ιδέα πίσω από τη logistic function είναι η χρησιμοποίηση μίας λογαριθμικής συνάρτησης, έτσι ώστε να περιοριστούν οι πιθανότητες στο εύρος (0,1).

5.2 Support Vector Machine

Το επόμενο στάδιο της εργασίας είναι η μελέτη των αλγορίθμων Support Vector Machines, εφόσον ολοκλήρωσα την μελέτη των αλγορίθμων Classification και Regression. Οι αλγόριθμοι που ανήκουν στην οικογένεια των SVM, αποτελούν ένα σύνολο

supervised μηχανικών μάθησης που μπορούν να χρησιμοποιηθούν για classification, regression και εντοπισμό outliers.

Σύμφωνα με τη φύση των δεδομένων που έχω συλλέξει οι τεχνικές του regression είναι φυσικό επόμενο να μην έχουν ικανοποιητική απόδοση, για αυτό το λόγο χρησιμοποιώ μόνο τις τεχνικές του SVM για classification.

Χρησιμοποιώ δύο διαφορετικούς αλγορίθμους SVM classification, τον SVC (C-Support Vector Classification) [73] [74] και τον LinearSVC (Linear Support Vector Classification) [75]

Η βασική ιδέα πίσω από τη λειτουργία των αλγορίθμων SVC είναι απλή, διαχωρισμός των διαφορετικών κλάσεων με τέτοιο τρόπο ώστε να μεγιστοποιείται η απόσταση μεταξύ των δειγμάτων. Μια διαφορετική προσέγγιση στην ερμηνεία των SVM είναι η εύρεση της εξίσωσης του hyperplane (υπερεπιπέδου) το οποίο οδηγεί στο μεγαλύτερο δυνατό διαχωρισμό ανάμεσα στις κλάσεις των δειγμάτων

Η δημιουργία αυτού του hyperplane, εξαρτάται από τη συνάρτηση kernel. Ο kernel ανάλογα με το πως ορίζεται, έχει τη δυνατότητα να αλλάξει διάσταση τα δεδομένα, προκειμένου να επιτευχθεί ο βέλτιστος διαχωρισμός κλάσεων.

Το μεγαλύτερο πλεονέκτημα των αλγορίθμων SVM συγκριτικά με τους κλασσικούς αλγορίθμους κατηγοριοποίησης (KNN, Decision Tree, Random Forest) είναι ο μικρός χρόνος εκτέλεσης των αλγορίθμων. Ο χρόνος εκπαίδευσης των αλγορίθμων είναι μεγάλος, καθώς και η διαδικασία εύρεσης βέλτιστων παραμέτρων, αλλά όχι η εκτέλεση. Για αυτό το λόγο, οι αλγόριθμοι SVM χρησιμοποιούνται συχνά σε προβλήματα πραγματικού χρόνου

5.2.1 SVC

Ο πρώτος αλγόριθμος SVM που μελετάω είναι ο SVC, ένας classifier ο οποίος έχει υλοποιηθεί πάνω στη βιβλιοθήκη του libsvm [73]. Η πολυπλοκότητα του χρόνου εκπαίδευσης είναι O(samples^4), ένα γεγονός το οποίο καθιστά δύσκολη την εφαρμογή του σε βάση δεδομένων με περισσότερα από 10,000 εγγραφές.

Στην περίπτωση των δεδομένων μου, όπου έχω multiclass δεδομένα, το μοντέλο λειτουργεί με την τεχνική one-vs-one. Στην τεχνική one-vs-one εκπαιδεύεται ξεχωριστός classifier για κάθε τιμή του y που υπάρχει στη βάση δεδομένων. Αυτή η τεχνική οδηγεί σε $\frac{N(N-1)}{2}$ classifiers, ένα μοντέλο το οποίο δεν είναι ευαίσθητο όταν συναντάει βάση δεδομένων με ανισορροπίες, αλλά την ίδια στιγμή είναι υπολογιστικά ακριβό.

Ο SVC προσφέρει τη δυνατότητα για δημιουργία τροποποιημένου kernel, αλλά και ένα σύνολο από 4 διαφορετικούς για επιλογή:

- linear(γραμμικό): (x,x’)
- polynomial (πολυωνυμικό): $(\gamma(x,x') + r)^d$, οι μεταβλητές r και d μπορούν να οριστούν χειροκίνητα
- rbf (Γκαουσιανή ακτινική συνάρτηση βάσης): $\exp(-\gamma|x-x'|^2)$, η μεταβλητή γ ορίζεται από το χρήστη και πρέπει να είναι μεγαλύτερη του 0.
- sigmoid (σιγμοειδής): $(\tanh(\gamma(x,x')+r))$, με την μεταβλητή r να μπορεί να οριστεί από το χρήστη

5.2.2 Linear SVC

Ο Linear SVC είναι ένας αλγόριθμος classification, παρόμοιος με τον SVC αν θέσουμε ως kernel ‘linear’. Η διάφορα τους έγκειται στο γεγονός ότι ο linear SVC είναι υλοποιημένος πάνω στη βιβλιοθήκη του liblinear [75] και όχι πάνω στην libsvm.

Η συγκεκριμένη υλοποίηση προσφέρει περισσότερες επιλογές στην επιλογή ποινών και συναρτήσεων απώλειας, καθώς επίσης είναι πιο εύκολο να εφαρμοστεί σε βάση δεδομένων με πολλές εγγραφές.

Η δεύτερη διαφορά του linear SVC με το SVC είναι ότι στην περίπτωση που υπάρχουν multiclass δεδομένα, αυτά διαχειρίζονται μέσω της τεχνικής one-vs-the-rest. Η τεχνική one-vs-the-rest ή αλλιώς γνωστή κι ως one-vs-all εκπαιδεύει ένα classifier ανά κλάση, δηλαδή υπάρχουν τόσο classifiers όσες κλάσεις. Για την κλάση ι υποθέτουμε ότι υπάρχουν i-κλάσεις οι οποίες είναι θετικές και οι υπόλοιπες αρνητικές. Αυτή η υπόθεση σε συνδυασμό με πιθανή ανισορροπία στα δεδομένα εισόδου, είναι πιθανό να οδηγήσει σε αδυναμία λειτουργίας του SVM.

5.3 Neural Networks

Εφόσον έχω ολοκληρώσει την μελέτη για τους αλγορίθμους classification, regression και SVM, πραγματοποιώ μια εισαγωγή για τα νευρωνικά δίκτυα (Neural Networks) [76]. Τα νευρωνικά δίκτυα, μπορούν να συναντηθούν στη βιβλιογραφία και ως τεχνητά νευρωνικά δίκτυα (artificial neural network), είναι δίκτυα που αποτελούνται από απλούς υπολογιστικούς κόμβους (νευρώνες ή νευρώνια) διασυνδεδεμένους μεταξύ τους. Τα νευρωνικά δίκτυα είναι εμπνευσμένα από τη λειτουργία του ανθρώπινου εγκεφάλου.

Οι νευρώνες είναι τα δομικά στοιχεία του δικτύου, δέχονται ένα σύνολο αριθμητικών εισόδων από διαφορετικές πηγές, είτε άλλους νευρώνες είτε το περιβάλλον, επιτελούν έναν υπολογισμό με βάση τις εισόδους και παράγουν μία έξοδο. Αυτή η έξοδος είτε κατευθύνεται στο περιβάλλον είτε σε άλλους νευρώνες.

Από τις διαφορετικές προσεγγίσεις και υλοποιήσεις που υπάρχουν για τα νευρωνικά δίκτυα, η βιβλιοθήκη του scikit-learn προσφέρει τον αλγόριθμο του Multi-layer Perceptron (MLP) [77]. Η λειτουργία του MLP είναι παρόμοια με τη λειτουργία οποιουδήποτε αλγορίθμου που ανήκει στην οικογένεια των νευρωνικών δικτύων. Είναι ένας αλγόριθμος που εκπαιδεύει μία συνάρτηση *f* μέσω ενός συνόλου δεδομένων, έτσι ώστε $f(\cdot): R^m \rightarrow R^o$, με τη μεταβλητή m να αντιπροσωπεύει τον αριθμό των διαστάσεων εισόδου και τη μεταβλητή o τον αριθμό διαστάσεων εξόδου.

Η διαφορά του με τον αλγόριθμο του logistic regression έγκειται στο γεγονός ότι ανάμεσα στο επίπεδο εισόδου και επίπεδο εξόδου, μπορούν να υπάρχουν ένα ή περισσότερα μη γραμμικά επίπεδα, που ονομάζονται υπολογιστικοί νευρώνες(hidden layers). Οι νευρώνες εισόδου, είναι ένα σύνολο νευρώνων $\{x_i \mid x_1, x_2, \ldots, x_m\}$ δέχονται απλά την είσοδο από το περιβάλλον και την προωθούν στους υπολογιστικούς νευρώνες. Οι υπολογιστικοί νευρώνες μετατρέπουν την τιμή που λαμβάνουν από το προηγούμενο επίπεδο με μία σταθμισμένη γραμμική άθροιση $w_1x_1 + w_2x_2 + \ldots + w_mx_m$, ακολουθούμενη από μια μη-γραμμική συνάρτηση ενεργοποίησης $g(\cdot): R \rightarrow R -$, όπως η υπερβολική συνάρτηση εφαπτομένης. Οι νευρώνες εξόδου λαμβάνουν τις τιμές από τον τελευταίο υπολογιστικό νευρώνα και το μεταφέρουν στο περιβάλλον ως μεταβλητή εξόδου.

Το μοντέλο του MLP Classifier βρίσκει τα βάρη μεταξύ των νευρώνων βελτιστοποιώντας τη συνάρτηση log-loss χρησιμοποιώντας τη μέθοδο LBFGS [78], μία μέθοδο που ανήκει στην οικογένεια των quasi-Newton, ή βελτιστοποιώντας τη συνάρτηση stochastic gradient descent (SGD) [79]. Ακόμη ο αλγόριθμος δίνει τη δυνατότητα για εισαγωγή μέσω παραμέτρων από τον προγραμματιστή μίας λίστας με πίνακες βαρών και μίας λίστας με διανύσματα πόλωσης.

5.4 Εφαρμογή των Classifiers και Μετρικές Απόδοσης

Εφόσον πραγματοποίησα τη θεωρητική εισαγωγή για τους αλγορίθμους classification που θα χρησιμοποιήσω, θα προχωρήσω στην εφαρμογή των αλγορίθμων στα δεδομένα μου. Πρέπει να αναφέρω για ακόμη μία φόρα ότι η υλοποίηση των αλγορίθμων δεν πραγματοποιήθηκε από μένα, αλλά από τη βιβλιοθήκη του scikit-learn. Ακόμη ισχύει ότι έχει αναφερθεί για τους κώδικες στο πρόγραμμα συλλογής δεδομένων, δεν παραθέτω ολόκληρο τον κώδικα, παρά μόνο μερικές γραμμές οι οποίες είναι απαραίτητες για την κατανόηση της λειτουργίας των αλγορίθμων.

Η πρώτη διεργασία που πρέπει να πραγματοποιηθεί είναι η ανάγνωση των δεδομένων. Για τη βάση δεδομένων μου χρησιμοποιώ MySQL, όπως έχω αναφέρει στο κομμάτι για πρόγραμμα συλλογής δεδομένων, αλλά είναι πιο εύχρηστο να διαβάζω αρχεία .csv για ανάλυση δεδομένων. Οπότε έχω εξάγει τη βάση δεδομένων σε αρχείο .csv.

```
mysql> SELECT *
FROM my_collected_data_table
INTO OUTFILE 'my_collected_data.csv'
FIELDS ENCLOSED BY ""
TERMINATED BY ';'
ESCAPED BY ""
LINES TERMINATED BY '\r\n';
```

Κώδικας 22: Εντολή εξαγωγής της βάσης δεδομένων σε αρχείο .csv

Ο λόγος που προτιμώ το αρχείο .csv είναι η ταχύτερη προσπέλαση του κατά την ανάγνωση σε σχέση με κλήσεις στη βάση δεδομένων, ενώ προτιμώ την αποθήκευση των δεδομένων σε βάση δεδομένων MySQL για λόγους συμβατότητας με όλα τα συστήματα.

Το αρχείο .csv που έχω εξάγει διαβάζεται ως pandas data frame, ο λόγος είναι ότι η συγκεκριμένη δομή δεδομένων είναι εύκολη στην επεξεργασία, καθώς επίσης η βιβλιοθήκη του scikit-learn έχει δημιουργηθεί για να δουλεύει μαζί με τη συγκεκριμένη δομή δεδομένων. Η μεταβλητή που περιέχει τα δεδομένα δεν διαθέτει τους απαραίτητους τίτλους για κάθε στήλη, οπότε τους θέτω χειροκίνητα και στη συνέχεια μετατρέπω τη μεταβλητή του για το είδος της δημοσίευσης σε αριθμητική μεταβλητή. Στη συνέχεια διαγράφω τα πεδία από τα οποία δεν μπορώ να εξάγω κάποια πληροφορία.

Το επόμενο βήμα είναι να χωρίσω τα δεδομένα μου σε υποσύνολο εκπαίδευσης (train set) και υποσύνολο ελέγχου (test set). Αυτή η ενέργεια είναι απαραίτητη, καθώς άμα δεν πραγματοποιηθεί έχει γίνει ένα θεμελιώδης λάθος, ο αλγόριθμος θα συναντάει εγγραφές που είναι ήδη γνωστές, με αποτέλεσμα να δίνει τέλειο σκορ, αλλά θα αποτυγχάνει να προβλέψει σωστά νέες εγγραφές.

Η δημιουργία των μοντέλων κάνοντας κλήση των classifier που θα χρησιμοποιήσω έπεται. Τα μοντέλα που έχω δημιουργήσει τα εκπαιδεύω με το υποσύνολο εκπαίδευσης και πραγματοποιώ πρόβλεψη στο υποσύνολο ελέγχου. Τελευταίο βήμα του προγράμματος

είναι η κλήση διαφόρων μετρικών για τον έλεγχο της απόδοσης των διαφορετικών classifiers. Η κλήση αυτών των μετρικών γίνεται μέσω της συνάρτησης benchmark, την οποία παραθέτω στη συνέχεια.

```
# read data
X = pd.read_csv('Datasets/thesis_facebook_data_v3.csv',header=None)

# name the columns
X.columns = ['status_id', 'status_message', 'link_name', 'status_type', 'status_link', 'status_published', 'num_reactions', 'num_comments', 'num_shares', 'num_likes', 'num_loves', 'num_wows', 'num_hahas', 'num_sads', 'num_angrys', 'num_cc', 'num_cd', 'num_dt', 'nun_ex', 'num_fw', 'num_in', 'num_jj', 'num_jjr', 'num_jjs', 'num_ls', 'num_md', 'num_nn', 'num_nns', 'num_nnp', 'num_nnps', 'num_pdt', 'num_pos', 'num_prp', 'num_prp6', 'num_rb', 'num_rbr', 'num_rbs', 'num_rp', 'num_sym', 'num_to', 'num_uh', 'num_vb', 'num_vbd', 'num_vbg', 'num_vbn', 'num_vbp', 'num_vbz', 'num_wdt', 'num_wp', 'num_wp6', 'num_wrb', 'human_score', 'computer_score']

X.status_type = X.status_type.map({'link':1, 'photo':2, 'status':3, 'video':4 })

# drop possible missings
X = X.dropna()
# name the y variable
y = X['computer_score']
# drop y from X
X = X.drop('computer_score',1)
# drop categorical data
X = X.drop(['status_id', 'status_message', 'link_name', 'status_link', 'status_published'], axis=1)
                        .
                        .
                        .
# split the data to test and train
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state=7)

# call the classifiers
knn = KNeighborsClassifier()

# call the predicts
y_pred_knn = knn.fit(X_train, y_train).predict(X_test)
                        .
                        .
                        .
# create the confusion matrixes
cm_knn = confusion_matrix(y_test, y_pred_knn)

# get the benchmark scores
benchmark(knn, cm_knn, y_pred_knn, 'KNeighbors')
                        .
                        .
                        .
```

Κώδικας 23: Μέρος του κώδικα της κυρίως συνάρτησης για την εφαρμογή των classifiers. Διαβάζει, επεξεργάζεται τα δεδομένα και καλεί τους classifiers.

Στη συνάρτηση *benchmark()* οι πρώτες γραμμές εξετάζουν το χρόνο εκτέλεσης και εκπαίδευσης για κάθε αλγόριθμο. Ίσως προκαλεί εντύπωση ότι η εκπαίδευση και η αξιολόγηση των αλγορίθμων βρίσκονται μέσα σε μία επανάληψη. Αυτή είναι μία τεχνική, η οποία αυξάνει την αξιοπιστία των μετρήσεων.

Η πρώτη μετρική είναι η default μετρική που προσφέρει το scikit-learn για τον κάθε αλγόριθμο, για τους 4 αλγόριθμους classification που χρησιμοποιώ η μετρική είναι η mean accuracy(μέση ακρίβεια) [80]. Σε περιπτώσεις όπου έχουμε multi-label classification, όπως στα δεδομένα μου, η μετρική είναι mean accuracy (μέση ακρίβεια) η οποία θεωρείται harsh μετρική [81], από τη στιγμή που απαιτεί για εγγραφή κάθε πεδίο να προβλεφθεί σωστά.

Η δεύτερη μετρική που χρησιμοποιώ για την αξιολόγηση της απόδοσης ενός αλγορίθμου είναι το cross-validation score. Το cross-validation score είναι η default μετρική για τον αλγόριθμο, στην περίπτωση που εξετάζουμε η mean accuracy, με τη διαφορά ότι πραγματοποιείται με την τεχνική του cross-validation. Το cross-validation είναι μία γκάμα από διαφορετικές επιμέρους τεχνικές, οι οποίες αποσκοπούν στην εξάλειψη του φαινομένου του overfitting [82]. Η παράμετρος *cv* που δέχεται δηλώνει τον αριθμό των διαιρέσεων που θα χρησιμοποιηθούν στα δεδομένα για να εξαχθεί η τελική βαθμολογία.

Precision είναι η μετρικής της ικανότητας του classifier να μην ταξινομεί ας θετικό μία εγγραφή της βάσης που είναι αρνητική [80]. Πιο συγκεκριμένα είναι η το κλάσμα *precision = tp÷(tp+fp)*, όπου *tp* είναι ο αριθμός των ορθών θετικών(true positive) και *fp*, ο αριθμός των ψευδών θετικών.

Το classification report είναι ένας πίνακας που συγκεντρώνει τις τέσσερις σημαντικότερες μετρικές για αλγορίθμους classification, precision, recall, f1-score και support. Πλην τη μετρική του support, τις υπόλοιπες μετρικές τις έχω παρουσιάσει ξεχωριστά. Η μετρική του support δηλώνει τον αριθμό εμφανίσεων κάθε κλάσης στο σύνολο των ορθών θετικών κατηγοριοποιήσεων.

Μια ακόμη μετρική που χρησιμοποιώ είναι η F1 score, γνωστή επίσης με το όνομα balanced F-score και F-measure. Αυτή η μετρική ερμηνεύεται ως ένας σταθμισμένος μέσος του precision και του recall [83], όπου η F1 κυμαίνεται στο εύρος [0,1], με 0 το χειρότερο δυνατό αποτέλεσμα και 1 το καλύτερο. Στην περίπτωση των δεδομένων μου, που είναι multiclass, η F1 υπολογίζεται ξεχωριστά για κάθε κλάση. Ο τύπος που δίνει την F1 είναι: *F1 = 2 * (precision * recall) ÷ (precision + recall)*.

Η μετρική της fbeta είναι ο σταθμισμένος αρμονικός μέσος μεταξύ του precision και του recall [84] και κυμαίνεται επίσης στο εύρος [0,1]. Η παράμετρος beta στον αλγόριθμο δηλώνει πόσο βάρος θα δοθεί στο precision και πόσο στο recall.

Για τις μετρικές precision, recall, f1 και fbeta χρησιμοποιώ τρεις διαφορετικούς μέσους(averages), macro, micro και average. Η χρήση μέσων είναι απαραίτητη για δεδομένα multiclass και multilabel, γιατί αν δεν χρησιμοποιηθεί κάποιος μέσος η μετρική θα επιστρέψει το σκορ ξεχωριστά για κάθε κλάση. Ο μέσος micro, υπολογίζει τη μετρική σε καθολικό επίπεδο υπολογίζοντας το συνολικό αριθμό των ορθά θετικών, ψευδών

αρνητικών και ψευδών θετικών. Ο macro πραγματοποιεί υπολογισμούς για κάθε κλάση και βρίσκει τους μη σταθμισμένους μέσους, πρέπει να σημειωθεί ότι με αυτό τον τρόπο δεν λαμβάνονται υπόψιν πιθανές ανισορροπίες στα πεδία. Τέλος ο μέσος weighted υπολογίζει τη μετρική για κάθε κλάση και βρίσκει τον μέσο, σταθμίζοντας με τη μετρική support, έτσι είναι σαν τον μέσο macro, λαμβάνοντας υπόψιν πιθανές ανισορροπίες. Η μετρική του classification report χρησιμοποιεί ως μέσο 'weighted'.

Σε αυτή την εργασία αν παραθέσω κάποια από τις 4 προαναφερθείσες μετρικές, χωρίς να προσδιορίσω μέσο, αυτός εννοείται ότι είναι micro. Ο λόγος που διάλεξα micro, ως παράμετρο του μέσου είναι επειδή η εναλλακτική παράμετρος macro, δεν λαμβάνει υπόψιν τυχόν ανισορροπίες στα label των δεδομένων μου και στα δεδομένα που έχω συλλέξει υπάρχουν ανισορροπίες. Η εναλλακτική του weighted είναι προτιμότερη από αυτή του macro, αλλά ενδέχεται το βάρος (support) που θέτει ο αλγόριθμος να οδηγήσει σε αστοχία. Σε αρκετές περιπτώσεις, weighted και micro έχουν το ίδιο σκορ.

Ως hamming loss θεωρούμε το κλάσμα που υπολογίζει το συνολικό αριθμό των κατηγοριών που έχουν υπολογιστεί λανθασμένα [85]. Με άλλα λόγια μετράει το ποσοστό εύστοχων αναθέσεων κλάσεων στις εγγραφές.

Η μετρική του Jaccard similarity, ορίζεται ως το κλάσμα της τομής μεταξύ δύο κλάσεων προς την ένωση των δύο κλάσεων [80]. Χρησιμοποιείται για να συγκρίνει το σύνολο των προβλεπόμενων κλάσεων με το σύνολο των εγγραφών που είναι χαρακτηρισμένα ως ορθά αληθή.

Επειδή έχω συλλέξει multilabel classification χρησιμοποιώ και τη μετρική του zero one loss, η οποία βαθμολογεί με άσσο ένα υποσύνολο εάν η πρόβλεψη είναι σωστή, και με 0 αν υπάρχουν κάποια λάθη και στο τέλος επιστρέφει το ποσοστό των λανθασμένων προβλέψεων [86]. Δηλαδή αν y_i είναι προβλεπόμενη τιμή και y_j η πραγματική τιμή, η μετρική zero one loss ισούται με $L_{0\text{-}1}(y_i,y_j) = 1(y_i \neq y_j)$.

Η τελευταία μετρική που χρησιμοποιώ είναι το confusion matrix. Confusion matrix είναι ένας πίνακας με τη βοήθεια του οποίου αξιολογούμε την ακρίβεια ενός classification. Ο πίνακας παρουσιάζει τον αριθμό των παρατηρήσεων που ανήκουν σε μία κλάση, αλλά έχουν προβλεφθεί μία άλλη. Το κελί $C_{i,j}$ δείχνει τον αριθμό των εγγραφών που ανήκουν στην κλάση i, αλλά έχουν προβλεφθεί στην κλάση j. Αν ο classifier που χρησιμοποιούμε παρουσιάζει τέλεια ακρίβεια, τότε θα υπάρχουν τιμές μόνο στη διαγώνιο.

Τις μετρικές των classification report και confusion matrix δεν τις εμφανίζω σε κάθε εκτέλεση αλγορίθμου, αλλά μόνο όταν ένα λεξικό παρουσιάζει απόδοση η οποία απαιτεί περαιτέρω μελέτη.

```
# function benchmark, metrics for the classifiers
def benchmark(model, cm, y_pred, wintitle='Figure 1'):

    print '\n\n' + wintitle + ' Results'
    s = time.time()
    for i in range(iterations):
        model.fit(X_train, y_train)
    print "{0} Iterations Training Time: ".format(iterations), time.time() - s

    s = time.time()
    for i in range(iterations):
       score =model.score(X_test, y_test)

    print ("{0} Iterations Scoring Time: ".format(iterations), time.time() - s)
    print ("Score is:", score)
    fold_cross_scor = cval.cross_val_score(model, X_train, y_train, cv=10).mean()
    print ("Fold cross scor: ", fold_cross_scor)
    print ("High-Dimensionality Score: ", round((score*100), 3))

    print ("---------------------------------------")
    print ("precision score, micro : ",precision_score(y_test, y_pred, average='micro'))
    print ("precision score, macro : ",precision_score(y_test, y_pred, average='macro'))
    print ("precision score, weighted : ",precision_score(y_test, y_pred, average='weighted'))

    print ("---------------------------------------")
    print ("recall score, weighted : ",recall_score(y_test, y_pred, average='weighted'))
    print ("recall score, macro : ",recall_score(y_test, y_pred, average='macro'))
    print ("recall score, micro : ",recall_score(y_test, y_pred, average='micro'))

    print ("---------------------------------------")
    print (classification_report(y_test, y_pred))

    print ("---------------------------------------")
    print ("f1 score, weighted : ",f1_score(y_test, y_pred, average='weighted'))
    print ("f1 score, macro : ",f1_score(y_test, y_pred, average='macro'))
    print ("f1 score, micro : ",f1_score(y_test, y_pred, average='micro'))

    print ("---------------------------------------")
    print ("fbeta score, micro : ", fbeta_score(y_test, y_pred, average='micro', beta=0.5))
    print ("fbeta score, macro : ", fbeta_score(y_test, y_pred, average='macro', beta=0.5))
    print ("fbeta score, weighted : ", fbeta_score(y_test, y_pred, average='weighted', beta=0.5))

    print ("---------------------------------------")
    print ("hamming loss : ", hamming_loss(y_test, y_pred))

    print ("---------------------------------------")
    print ("jaccard similarity, normalized: ", jaccard_similarity_score(y_test, y_pred))

    print ("---------------------------------------")
    print ("zero one loss: ", zero_one_loss(y_test, y_pred))
```

```
    np.set_printoptions(precision=2)
    print('Confusion matrix, without normalization')
    print(cm)
```

Κώδικας 24: Η συνάρτηση benchmark, περιέχει όλες τις μετρικές για την αξιολόγηση των classifiers

6. Παράθεση Αποτελεσμάτων

Έχοντας ολοκληρώσει την ανάλυση για τους αλγορίθμους του classification και του regression, αλλά και των μετρικών που χρησιμοποιούνται για την αξιολόγηση τους, απομένει η παρουσίαση των αποτελεσμάτων τους. Για λόγους οικονομίας χώρου τα αποτελέσματα τα παρουσιάσω σε μορφή πινάκων, καθώς επίσης παρουσιάζω και μερικά διαγράμματα για τη καλύτερη κατανόηση.

Τους αλγορίθμους δεν τους εκτελώ μόνο με τη χρήση των default παραμέτρων, αλλά εκτελώ και αναζήτηση βέλτιστων παραμέτρων. Οι βέλτιστοι παράμετροι για κάθε αλγόριθμο αλλάζουν ανάλογα με τα δεδομένα εισόδου, στα δεδομένα εισόδου συμπεριλαμβάνεται και η αξιολόγηση των δεδομένων από τα διαφορετικά λεξικά. Δηλαδή ο KNN έχει διαφορετικούς βέλτιστους παραμέτρους όταν τα δεδομένα εισόδου αξιολογούνται από το λεξικό του AFINN και διαφορετικούς όταν αξιολογούνται από το λεξικό του imdb.

6.1 Παράμετροι εισόδου

Το scikit-learn προσφέρει δύο διαφορετικούς τρόπους αναζήτησης βέλτιστων παραμέτρων, ο πρώτος είναι μέσω τυχαίας επιλογής από ένα σύνολο (Random Search) και ο δεύτερος μέσω εξαντλητικής αναζήτησης χρησιμοποιώντας όλες τις πιθανές μεταβλητές (Grid Search). Πρέπει να σημειωθεί ότι η εξαντλητική αναζήτηση ενδέχεται να διαρκέσει ιδιαίτερα μεγάλο χρονικό διάστημα, μέχρι και μέρες, ανάλογα με τον αλγόριθμο και το σύνολο των παραμέτρων που δοκιμάζει, σε αντίθεση με την τυχαία αναζήτηση που πραγματοποιεί όσους συνδυασμούς θέσει ο χρήστης.

Σε αυτό το υποκεφάλαιο θα παρουσιάσω τα σύνολα των αλγορίθμων μέσα από τα οποία γίνεται η αναζήτηση βέλτιστων παραμέτρων, καθώς επίσης και μέρη κώδικα τα οποία πραγματοποιούν τη αναζήτηση βέλτιστων παραμέτρων.

Ο κώδικας που πραγματοποιεί την αναζήτηση βέλτιστων παραμέτρων διαφοροποιείται ελαφρώς από αλγόριθμο σε αλγόριθμο, ανάλογα με τις παραμέτρους εισόδου που λαμβάνει ο καθένας. Για λόγους οικονομίας χώρου θα παρουσιάσω μέρος του κώδικα που πραγματοποιεί την αναζήτηση βέλτιστων παραμέτρων για τον classifier του KNN.

Το μέρος του κώδικα το οποίο διαβάζει και επεξεργάζεται τα δεδομένα παραμένει ίδιο, για αυτό το λόγο στο στιγμιότυπο του κώδικα που ακολουθεί, το παραλείπω. Στη συνέχεια δημιουργώ τον classifier, δηλώνω το σύνολο των παραμέτρων εισόδου, ορίζω τον αριθμό αναζητήσεων για το Random Search, πραγματοποιώ την αναζήτηση και την πρόβλεψη και καλώ τη συνάρτηση που ταξινομεί τις εκτελέσεις με τις συγκεκριμένες παραμέτρου που δίνουν το υψηλότερο σκορ. Επίσης σημειώνω τον χρόνο εκτέλεσης των αναζητήσεων, έτσι ώστε να μπορώ να εκτιμήσω κόστος για την βελτίωση των μετρικών.

```
                        .
                        .
                        .
# build a classifier
clf = KNeighborsClassifier()
                        .
                        .
                        .
# specify parameters and distributions to sample from
param_dist = {
            "algorithm": ["ball_tree", "kd_tree", "brute", "auto"],
            "weights": ["uniform", "distance"],
            "n_neighbors" : sp_randint(1, 15),
            "leaf_size" :  sp_randint(15, 60),
            "metric": ["euclidean","manhattan", "chebyshev"]
            }

# run randomized search
n_iter_search = 20
random_search = RandomizedSearchCV(clf, param_distributions=param_dist,
                           n_iter=n_iter_search)

start = time()
random_search.fit(X_train, y_train)
y_pred_knn = random_search.fit(X_train, y_train).predict(X_test)

print("RandomizedSearchCV took %.2f seconds for %d candidates"
      " parameter settings." % ((time() - start), n_iter_search))
report(y_pred_knn, random_search.grid_scores_)
                        .
                        .
                        .
# use a full grid over all parameters
```

```
param_grid = {
            "algorithm": ["ball_tree", "kd_tree", "brute", "auto"],
            "weights": ["uniform", "distance"],
            "n_neighbors": np.arange( 1, 30, 1 ).tolist(),
            "leaf_size": np.arange( 5, 30, 1 ).tolist(),
            "metric": ["euclidean","manhattan", "chebyshev"]
           }

clf.get_params().keys()

# run grid search
grid_search = GridSearchCV(clf, param_grid=param_grid)
start = time()
grid_search.fit(X, y)
y_pred_knn = grid_search.fit(X_train, y_train).predict(X_test)

print("GridSearchCV took %.2f seconds for %d candidate parameter settings."
     % (time() - start, len(grid_search.grid_scores_)))
report(y_pred_knn, grid_search.grid_scores_)
```

Κώδικας 25: Μέρος της κυρίας συνάρτησης που πραγματοποιεί αναζήτηση βέλτιστων παραμέτρων

Η συνάρτηση report, η οποία αξιολογεί τα ευρήματα των αναζητήσεων, αρχικά πραγματοποιεί μία ταξινόμηση των αποτελεσμάτων, καλεί τις μετρικές απόδοσης που έχω χρησιμοποιήσει στη συνάρτηση benchmark και τέλος παρουσιάζει τις παραμέτρους εισόδου που παρουσιάζουν τα καλύτερα αποτελέσματα.

```
# Utility function to report best scores
def report(y_pred, grid_scores, n_top=3):
   top_scores = sorted(grid_scores, key=itemgetter(1), reverse=True)[:n_top]
   for i, score in enumerate(top_scores):
      print("Model with rank: {0}".format(i + 1))
      print("Mean validation score: {0:.3f} (std: {1:.3f})".format(
           score.mean_validation_score, np.std(score.cv_validation_scores)))
                        .
                        .
                        .
      print("Parameters: {0}".format(score.parameters))
      print("")
```

Κώδικας 26: Η συνάρτηση report, ταξινομεί τις εκτελέσεις αλγορίθμων ανάλογα με την απόδοση τους και παρουσιάζει τις αντίστοιχες παραμέτρους εισόδου

Είναι πολύ σημαντικό να σημειωθεί ότι η συνάρτηση που προσφέρει το scikit-learn για την αναζήτηση βέλτιστων παραμέτρων θεωρεί ως κριτήριο αξιολόγησης του εκάστοτε υποσυνόλου το cross fold validation score. Έτσι, ενδέχεται το νέο βέλτιστο σύνολο παραμέτρων για κάποιον αλγόριθμο να παρουσιάζει χειρότερο σκορ για κάποιες μετρικές.

Έχοντας παρουσιάσει και τον κώδικα ο οποίος πραγματοποιεί την αναζήτηση βέλτιστων παραμέτρων τόσο μέσω Random Search, όσο και μέσω Grid Search στη

συνέχεια παρουσιάζω τα δυνατά σύνολα παραμέτρων για όλους τους αλγορίθμους που χρησιμοποιώ.

Ο αλγόριθμος του KNN έχει τη δυνατότητα να χρησιμοποιήσει 4 διαφορετικές αποστάσεις, ευκλείδεια, manhattan, chebyshev και minkowski. Η αναγκαία χρησιμοποίηση δύναμης για τον υπολογισμό με βάση την απόσταση minkowski, με οδήγησε στη χρησιμοποίηση δύο διαφορετικών συνόλων παραμέτρων για την εκπλήρωση της εξαντλητικής αναζήτησης παραμέτρων. Στη Random Search αγνόησα τη μετρική minkowski.

Το πρώτο σύνολο παραμέτρων εξετάζει τις αποστάσεις ευκλείδεια, manhattan και chebyshev και ακόμα 4 διαφορετικές παραμέτρους. Συγκεκριμένα το υπό εξέταση σύνολο φαίνεται παρακάτω:

"algorithm": ["ball_tree", "kd_tree", "brute", "auto"],
"weights": ["uniform", "distance"],
"n_neighbors" : np.arange(1, 30, 1).tolist(),
"leaf_size" : np.arange(5, 30, 1).tolist(),
"metric": ["euclidean","manhattan", "chebyshev"]

Το δεύτερο σύνολο παραμέτρων εξετάζει τις μετρικές που εξάγονται χρησιμοποιώντας τη μετρική του minkowski, οι τιμές για τις παραμέτρους εισόδου στις πρώτες τρεις μεταβλητές έχουν παραμείνει ίδιες και έχουν αλλάξει οι μεταβλητές του metric και metric_params σε :

"metric": ["minkowski"],
'metric_params':[{'p':1.5}, {'p':2.5}, {'p':3.5}, {'p':3.0}]

Όσον αφορά τον αλγόριθμο του Decision Tree, το σύνολο των πιθανών παραμέτρων είναι:

"criterion": ["gini", "entropy"],
"splitter": ["best", "random"],
"max_depth": [3, None],
"max_features": ["auto","sqrt","log2",None],
"min_samples_split": sp_randint(1, 55),
"min_samples_leaf": sp_randint(1, 11),
"max_leaf_nodes":[None, 3, 5, 7, 10],
"presort":[True, False]

Ο τρίτος classifier που εξετάζω για να βρω βέλτιστες παραμέτρους είναι ο Random Forest. Οι παράμετροι που χρησιμοποίησα είναι από το παρακάτω σύνολο:

"n_estimators" : sp_randint(2,19),
"criterion": ["gini", "entropy"],
"max_features": sp_randint(1, 7),
"max_depth": [3, None],
"min_samples_split": sp_randint(1, 11),
"min_samples_leaf": sp_randint(1, 11),

"max_leaf_nodes":[None, 3, 5, 7, 10],
"oob_score": [True, False],
"class_weight": ["balanced", "balanced_subsample", None]

Ο τέταρτος και τελευταίος απλός classifier στον οποίο αναζητώ βέλτιστους παραμέτρους είναι ο Logistic Regression. Ο συγκεκριμένος αλγόριθμος έχει λιγότερες παραμέτρους από τους υπόλοιπους, συγκεκριμένα το σύνολο των παραμέτρων που δοκιμάζονται, φαίνονται παρακάτω:

"C": [0.3, 0.5, 0.8, 1.0, 1.5, 2.0, 2.5],
"solver": ['newton-cg', 'lbfgs', 'liblinear', 'sag']

Ο πρώτος αλγόριθμος της οικογένειας των SVM που μελετάω το σύνολο των δυνατών παραμέτρων είναι ο SVC. Ο SVC έχει τη δυνατότητα να λάβει τιμές σε τέσσερις διαφορετικές μεταβλητές, το σύνολο των δυνατών παραμέτρων φαίνεται παρακάτω:

"C": [0.1, 0.3, 0.4, 0.6, 0.8, 1, 1.2, 1.5, 1.8, 2.1],
"kernel" : ["rbf", "sigmoid"],
"probability": [True, False],
"shrinking": [True, False]

Μετά την παράθεση των δυνατών τιμών για τις παραμέτρους εισόδου στον αλγόριθμο του SVC, προχωράω στην αναζήτηση βέλτιστων παραμέτρων για τον αλγόριθμο του Linear SVC. Ο Linear SVC διαθέτει αρκετές μεταβλητές εισόδου, αλλά επειδή συγκεκριμένες τιμές σε κάποιες μεταβλητές απαιτούν συγκεκριμένες τιμές σε κάποιες άλλες μεταβλητές, μείωσα το σύνολο των δυνατών παραμέτρων. Το σύνολο που χρησιμοποίησα φαίνεται παρακάτω:

"C":[0.1, 0.3, 0.4, 0.6, 0.8, 1, 1.2, 1.5, 1.8, 2.1],
"loss":['hinge', 'squared_hinge'],
"multi_class" : ['ovr', 'crammer_singer']

Ο MLP είναι ο τελευταίος classifier για τον οποίο παρουσιάζω το σύνολο παραμέτρων μέσα από το οποίο πραγματοποιώ αναζήτηση βέλτιστων παραμέτρων. Ο αλγόριθμος του MLP δέχεται πάρα πολλούς παραμέτρους εισόδου, συγκεκριμένα το σύνολο παραμέτρων που δέχεται φαίνεται παρακάτω:

"hidden_layer_sizes" : [(random.randint(1, 100), random.randint(1, 100)) for k in range(100)],
"activation" : ["identity", "logistic", "tanh", "relu"],
"solver" : ["lbfgs", "sgd", "adam"],
"alpha" : [0.0000001, 0.00001, 0.0001, 0.001, 0.01, 0.1],
"batch_size" : ["auto", 100, 125, 150, 200, 250, 300, 350],
"learning_rate" : ["constant", "invscaling", "adaptive"],
"max_iter" : [100, 150, 200, 250, 300],
"shuffle" : [True, False],
"tol" : [0.00001, 0.0001, 0.001, 0.01, 0.1],
"learning_rate_init" : [0.0001, 0.001, 0.01, 0.1],

"power_t" : [0.25, 0.5, 0.75, 1, 1.25, 1.5],
"momentum" : [0.1, 0.3, 0.5, 0.7, 0.9],
"nesterovs_momentum": [True, False],
"early_stopping" : [True, False],
"validation_fraction" : [0.1, 0.3, 0.5, 0.7, 0.9],
"beta_1" : [0.1, 0.3, 0.5, 0.7, 0.9],
"beta_2" : [0.1, 0.3, 0.5, 0.7, 0.9, 0.999],
"epsilon" : [1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1]

Οι δυνατοί συνδυασμοί είναι πάρα πολλοί θέτοντας απαγορευτική την αναζήτηση μέσω Grid Search σε ένα απλό υπολογιστικό μηχάνημα. Έτσι, πραγματοποιώ 3 αναζητήσεις μέσω Random Search, μία με 100, με 1000 και 10000 πιθανούς συνδυασμούς. Τη συγκεκριμένη τεχνική την εφαρμόζω στα δεδομένα εισόδου, ανεξάρτητα από το λεξικό το οποίο αξιολογεί τα δεδομένα μου.

Έχοντας ολοκληρώσει την παρουσίαση των συνόλων που χρειάζονται για την αναζήτηση βέλτιστων παραμέτρων στους αλγορίθμους του classification, προχωράω στα σύνολα μεταβλητών που χρησιμοποιούνται από τους αλγορίθμους regression. Ο αλγόριθμος του linear regression δέχεται μόλις δύο παραμέτρους και αυτοί οι παράμετροι είναι Boolean, οπότε οι η εφαρμογή Random Search για εύρεση βέλτιστων παραμέτρων δεν έχει νόημα. Το σύνολο των παραμέτρων είναι:

"fit_intercept": [True, False],
"normalize": [True, False]

Ο Bayesian Ridge Regression έχει περισσότερους παραμέτρους, τους οποίους μπορώ να τροποποιήσω. Και σε αυτό τον αλγόριθμο οι παράμετροι που δέχεται πλην του αριθμού των μέγιστων επαναλήψεων είναι Boolean μεταβλητές. Οι παράμετροι φαίνονται παρακάτω:

"n_iter": sp_randint (55, 600),
"compute_score": [True, False],
"fit_intercept": [True, False],
"normalize": [True, False],
"copy_X": [True, False]

6.2 Προεπεξεργασία δεδομένων

Κάποιοι classifiers παρουσιάζουν διακύμανση στην απόδοση τους εξαιτίας της τυχαιότητας που έχουν στην δημιουργία τους. Η διακύμανση στην απόδοση αυτών των αλγορίθμων μπορεί να μειωθεί, εάν τα δεδομένα εισόδου προ-επεξεργαστούν και μεταφερθούν σε διαφορετική κλίμακα. Αυτή η προεπεξεργασία είναι συναρτήσεις που προσφέρονται από τη βιβλιοθήκη του scikit-learn και αλλάζουν τα δεδομένα σε κάποια πιο κατάλληλη μορφή για τα μοντέλα που δημιουργώ, στα οποία δοκιμάζω την απόρριψη συγκεκριμένων πεδίων που κρίνω ότι δεν προσφέρουν αρκετή πληροφορία στους classifiers για εξαγωγή συμπερασμάτων

6.2.1 Standardization

Η διαδικασία του Standardization στα δεδομένα εισόδου είναι σύνηθες προαπαιτούμενο για πολλούς classifiers. Ο λόγος είναι ότι πολλοί classifiers έχουν βέλτιστη απόδοση όταν τα δεδομένα εισόδου παρουσιάζουν Γκαουσιανή κατανομή με μέση τιμή μηδέν και κοινή διακύμανση.

Η βιβλιοθήκη του *scikit-learn* παρέχει 4 ειδών standardization μέσω του πακέτου *preprocessing*. Στην εργασία μου χρησιμοποίησα και τα 4 είδη standardization, τα οποία αναλύω παρακάτω και παραθέτω τα νέα αποτελέσματα που παρουσιάζουν οι αλγόριθμοι classification.

6.2.1.1 StandardScaler

Η συνάρτηση του StandardScaler αλλάζει τις τιμές των πεδίων αφαιρώντας τον μέσο και αλλάζοντας τη διακύμανση σε κοινές τιμές. Το κεντράρισμα και η κλιμάκωση των τιμών πραγματοποιούνται ανεξάρτητα για κάθε πεδίο, υπολογίζοντας τα στατιστικά στοιχεία για τις εγγραφές και κάθε σύνολο που εκπαιδεύεται. Στη συνέχεια η μέση τιμή και η τυπική απόκλιση αποθηκεύονται για να χρησιμοποιηθούν στην μετατροπή των δεδομένων.

6.2.1.2 MinMaxScaler

Η δεύτερη συνάρτηση Standardization που μελετάω είναι η MinMaxScaler, ένας τρόπος για να επιτευχθεί η αλλαγή στο εύρος τιμών των μεταβλητών. Συνήθως η αλλαγή στο εύρος τιμών είναι στο πεδίο [0,1]. Ο τύπος μέσω του οποίου υπολογίζονται τα νέα εύρη τιμών είναι:

X_std = (X - X.min(axis=0)) / (X.max(axis=0) - X.min(axis=0))
X_scaled = X_std * (max - min) + min

όπου min,max = εύρος τιμών των πεδίων. Η συγκεκριμένη συνάρτηση δίνει τη δυνατότητα για αλλαγή του εύρους τιμών που θα λάβουν τα δεδομένα μας μετά τη μετατροπή.

6.2.1.3 RobustScaler

Η τελευταία τεχνική Standardization που χρησιμοποιώ είναι το RobustScaler. Η συγκεκριμένη τεχνική αφαιρεί τον μέσο και μετατρέπει τα δεδομένα σε άλλη κλίμακα σύμφωνα με το quartile range (εύρος τεταρτημορίου), ενώ στις υπόλοιπες τεχνικές η μετατροπή γίνεται στο Inter-quartile Range (IQR - ενδοτεταρτημοριακό εύρος). Το IQR κυμαίνεται ανάμεσα στο πρώτο και στο τρίτο quartile.

Το κεντράρισμα και η κλιμάκωση των δεδομένων πραγματοποιούνται ανεξάρτητα σε κάθε εγγραφή υπολογίζοντας τη διακύμανση των εγγραφών στο δείγμα εκπαίδευσης. Ο μέσος και το ενδοτεταρτημοριακό εύρος για κάθε εγγραφή αποθηκεύονται και χρησιμοποιούνται όταν πραγματοποιείται η μετατροπή των δεδομένων

6.2.2 Normalization

Εφόσον ολοκλήρωσα την ανάλυση των μεθόδων του Standardization, προχωράω στο Normalization. Ως Normalization χαρακτηρίζεται η διαδικασία που μετατρέπει τα δεδομένα εισόδου έτσι ώστε να έχουν μία ενιαία μορφή. Το scikit-learn προσφέρει 3 διαφορετικούς κανόνες μετατροπής ‘l1’, ‘l2’ και ‘max’ για κάθε μη μηδενική εγγραφή.

6.2.3 Απόρριψη Πεδίων

Το επόμενο βήμα στην προεπεξεργασία των δεδομένων μου είναι η απόρριψη συγκεκριμένων πεδίων, τα οποία θεωρώ ότι δεν επηρεάζουν θετικά ή αρνητικά την πρόγνωση για το σκορ λεξιλογίου.

Τα πεδία που έχω στην ολοκληρωμένη βάση δεδομένων μου είναι 46. Ο αριθμός των πεδίων μου συγκριτικά με τον αριθμό των εγγραφών μου θεωρείται μεγάλος και προτείνεται ο κανόνας του 10%, δηλαδή τα πεδία να είναι τόσα όσο είναι το 10% των εγγραφών. Τα μόνα πεδία που θα είναι λάθος να αφαιρεθούν από τη βάση δεδομένων μου είναι οι αντιδράσεις, συνολικά 6 διαφορετικές, τα σχόλια, οι κοινοποιήσεις και τα 7 μέρη του λόγου τα οποία χρησιμοποιούνται κατά μέσο όρο περισσότερο από μία φορά ανά δημοσίευση.

6.3 Αποτελέσματα του λεξικού AFINN

Σε αυτό το κεφάλαιο παρουσιάζω τα αποτελέσματα των classifiers πάνω στα δεδομένα που έχω εξάγει, όταν αυτά αξιολογούνται από το λεξικό του AFINN. Τα δεδομένα τα χωρίζω σε δύο υποκεφάλαια, ανάλογα με το κοινωνικό δίκτυο από το οποίο προέρχονται.

6.3.1 Αποτελέσματα στα δεδομένα που εξαχθεί από το Facebook

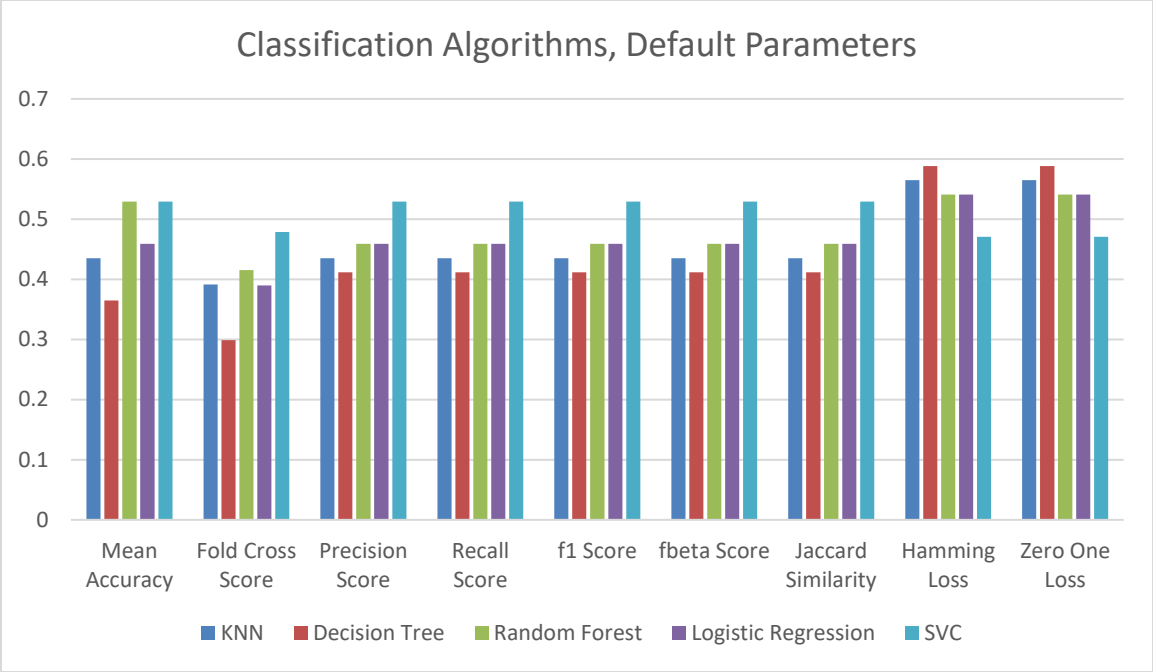
Σε κάθε λεξικό πρώτα παρουσιάζω την απόδοση του σε δεδομένα που έχουν εξαχθεί από το Facebook κι έπειτα σε δεδομένα που έχουν εξαχθεί από το Twitter. Στον πίνακα που ακολουθεί παρουσιάζω την απόδοση των classifiers, μέχρι το τέταρτο δεκαδικό ψηφίο, όταν εκτελούνται από τον αλγόριθμο με τις default παραμέτρους εισόδου.

	KNN	DT	RF	LR	SVC
Train Time(s)	0.0	0.0	0.0320	0.0780	0.0149
Score Time(s)	0.0	0.0160	0.0	0.0	0.0
Mean Accuracy	0.4352	0.3647	0.5294	0.4588	0.5294
Fold Cross Score	0.3914	0.2989	0.4154	0.3898	0.4784
Prec Score, micro	0.4352	0.4117	0.4588	0.4588	0.5294
Prec Score, macro	0.0782	0.1517	0.1828	0.0731	0.0588
Prec Score, weighted	0.2981	0.4348	0.4560	0.3155	0.2802
Recall Score, micro	0.4352	0.4117	0.4588	0.4588	0.5294
Recall Score, macro	0.1047	0.1304	0.1055	0.0923	0.1111
Recall Score, weighted	0.4352	0.4117	0.4588	0.4588	0.5294
f1 Score, micro	0.4352	0.4117	0.4588	0.4588	0.5294
f1 Score, macro	0.0885	0.1358	0.1027	0.0802	0.0769
f1 Score, weighted	0.3530	0.4163	0.3959	0.3720	0.3665
fbeta Score, micro	0.4352	0.4117	0.4588	0.4588	0.5294
fbeta Score, macro	0.0818	0.1433	0.1165	0.0755	0.0649
fbeta Score, weighted	0.3177	0.4247	0.3864	0.3355	0.3093
Jaccard Similarity	0.4352	0.4117	0.4588	0.4588	0.5294
Hamming Loss	0.5647	0.5882	0.5411	0.5411	0.4705
Zero One Loss	0.5647	0.5882	0.5411	0.5411	0.4705

Πίνακας 21: Σύγκριση classifiers, λεξικό AFINN, default παράμετροι εισόδου, δεδομένα από Facebook

Από τους classifiers που χρησιμοποιώ ξεχωρίζει ο SVC, ο οποίος εμφανίζει τα υψηλότερα σκορ σε όλες τις μετρικές. Αντίθετα ο Decision Tree, εμφανίζει τα χειρότερα, επίσης σε όλες τις μετρικές. Πρέπει να σημειωθεί ότι οι Decision Tree και Random Forest εμφανίζουν μία διακύμανση στα σκορ των μετρικών που χρησιμοποιώ που κυμαίνεται στο ανεκτό ±4%, ενώ οι KNN, Logistic Regression και SVC όχι.

Στον πίνακα που παρέθεσα δεν παρουσίασα τους classifiers των Linear SVC και MLP, επειδή παρουσιάζουν πολύ μεγάλη διακύμανση σε όλες τις μετρικές και η αξιολόγηση τους γίνεται αδύνατη.



Διάγραμμα 25: Σύγκριση classifiers, λεξικό AFINN, default παράμετροι εισόδου, δεδομένα από Facebook

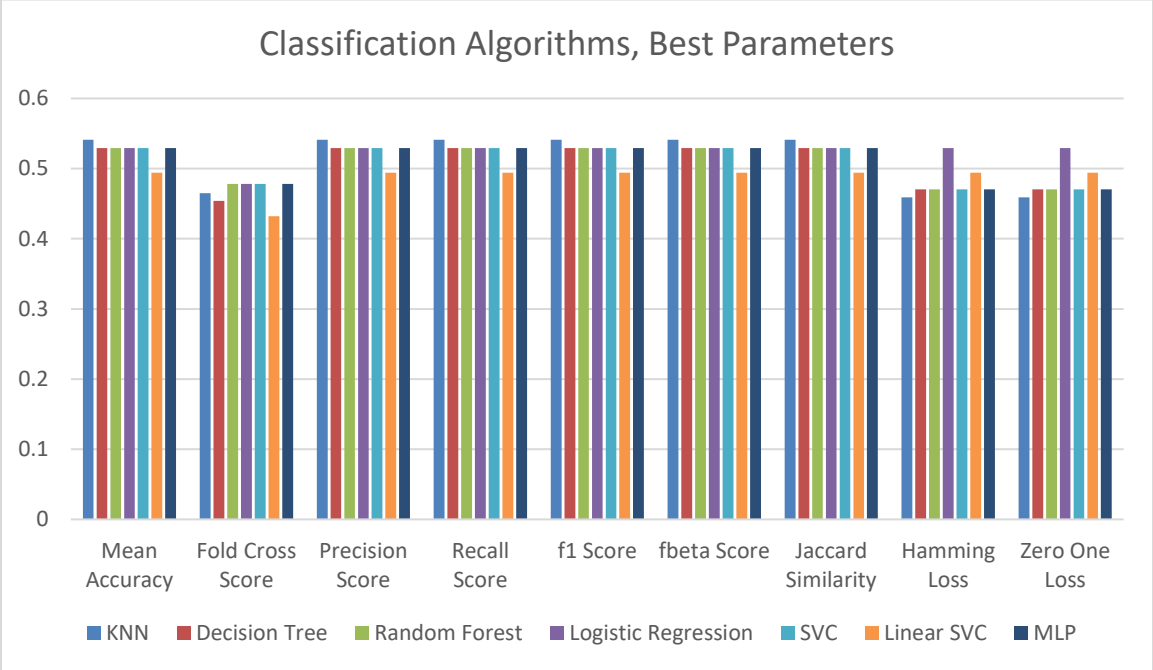
Από το διάγραμμα σύγκρισης classifiers φαίνεται και οπτικά η υπεροχή του SVC, ενώ παράλληλα διακρίνεται ότι ο Decision Tree παρουσιάζει τα χειρότερα αποτελέσματα. Τέλος, να σημειωθεί ότι οι χρόνοι εκτέλεσης και των 4 αλγορίθμων είναι αμελητέοι. Τα σκορ που παρουσιάζουν οι classifiers που εξετάζω σε οποιαδήποτε μετρική δεν μπορούν να χαρακτηριστούν επαρκή. Για αυτό το λόγο παραθέτω τα αποτελέσματα των classifiers με τη χρήση των βέλτιστων παραμέτρων και εφόσον τα δεδομένα εισόδου έχουν υποστεί προεπεξεργασία.

	KNN	DT	RF	LR	SVC	L. SVC	MLP
Train Time(s)	0.0	0.0	0.0310	0.125	0.0149	0.0850	0.9972
Score Time(s)	0.0	0.0020	0.0	0.0	0.0	0.0	0.0014
Mean Accuracy	0.5411	0.5294	0.5294	0.5294	0.5294	0.4941	0.5294
Fold Cross Score	0.4651	0.4541	0.4784	0.4784	0.4784	0.4323	0.4784
Prec Score, micro	0.5411	0.5294	0.5294	0.5294	0.5294	0.5058	0.5294
Prec Score, macro	0.1706	0.1145	0.0588	0.0588	0.0588	0.0537	0.0588
Prec Score, weighted	0.4483	0.3222	0.2802	0.2802	0.2802	0.2845	0.2802
Recall Score, micro	0.5411	0.5294	0.5294	0.5294	0.5294	0.5058	0.5294
Recall Score, macro	0.1190	0.1379	0.1111	0.5294	0.1111	0.0955	0.1111
Recall Score, weighted	0.5411	0.5294	0.5294	0.1111	0.5294	0.5058	0.5294
f1 Score, micro	0.5411	0.5294	0.5294	0.5294	0.5294	0.5058	0.5294
f1 Score, macro	0.0923	0.1162	0.0769	0.5294	0.0769	0.0688	0.0769
f1 Score, weighted	0.3913	0.3912	0.3665	0.0769	0.3665	0.3642	0.3665
fbeta Score, micro	0.5411	0.5294	0.5294	0.3665	0.5294	0.5058	0.5294
fbeta Score, macro	0.0964	0.1130	0.0649	0.5294	0.0649	0.0589	0.0649
fbeta Score, weighted	0.3583	0.3442	0.3093	0.0649	0.3093	0.3118	0.3093
Jaccard Similarity	0.5411	0.5294	0.5294	0.3093	0.5294	0.5058	0.5294

Hamming Loss	0.4588	0.4705	0.4705	0.5294	0.4705	0.4941	0.4705
Zero One Loss	0.4588	0.4705	0.4705	0.4705	0.4705	0.4941	0.4705

Πίνακας 22: Σύγκριση classifiers, λεξικό AFINN, βέλτιστοι παράμετροι εισόδου, δεδομένα από Facebook

Από τον πίνακα στον οποίο παραθέτω, γίνεται κατανοητό ότι όταν οι classifiers βελτιστοποιούνται, όλοι τους παρουσιάζουν κοντινά αποτελέσματα, πλην του Linear SVC, ο οποίος υπολείπεται σε όλες τις μετρικές. Δεν παραθέτω για τον κάθε classifier τις βέλτιστες παραμέτρους, ούτε τις τεχνικές προεπεξεργασίας με τις οποίες οι αλγόριθμοι εξάγουν τα καλύτερα αποτελέσματα. Έχω περιγράψει τη μεθοδολογία και έχω παρουσιάσει τα σημαντικά κομμάτια για τη συγγραφή του κώδικα, οπότε τα αποτελέσματα μου μπορούν να εξακριβωθούν ανά πάσα στιγμή.



Διάγραμμα 26: Σύγκριση classifiers, λεξικό AFINN, βέλτιστοι παράμετροι εισόδου, δεδομένα από Facebook

Στο διάγραμμα απεικονίζεται και οπτικά η απόδοση των classifiers, στο οποίο διακρίνεται η ελαφρώς καλύτερη απόδοση του KNN. Ο KNN είναι ο classifier ο οποίος όταν δέχθηκε αλλαγές στις παραμέτρους εισόδου και τα δεδομένα εισόδου δέχθηκαν προεπεξεργασία παρουσίασε τη μεγαλύτερη βελτίωση. Ακόμη να αναφέρω ότι οι classifiers δεν παρουσιάζουν μεγάλη διαφορά, μεγαλύτερη του ±3, με την απόρριψη πεδίων. Ως γενικό συμπέρασμα για το λεξικό του AFINN μπορεί να εξαχθεί ότι δεν είναι επαρκή για τα δεδομένα που έχω συλλέξει, χωρίς βέβαια η απόδοση του να είναι τόσο κακή ώστε να απορριφθεί ολοκληρωτικά, με κάποιες προσθέσεις στο λεξικό μπορεί να παρουσιάσει καλύτερα αποτελέσματα.

6.3.2 Αποτελέσματα στα δεδομένα που εξαχθεί από το Twitter

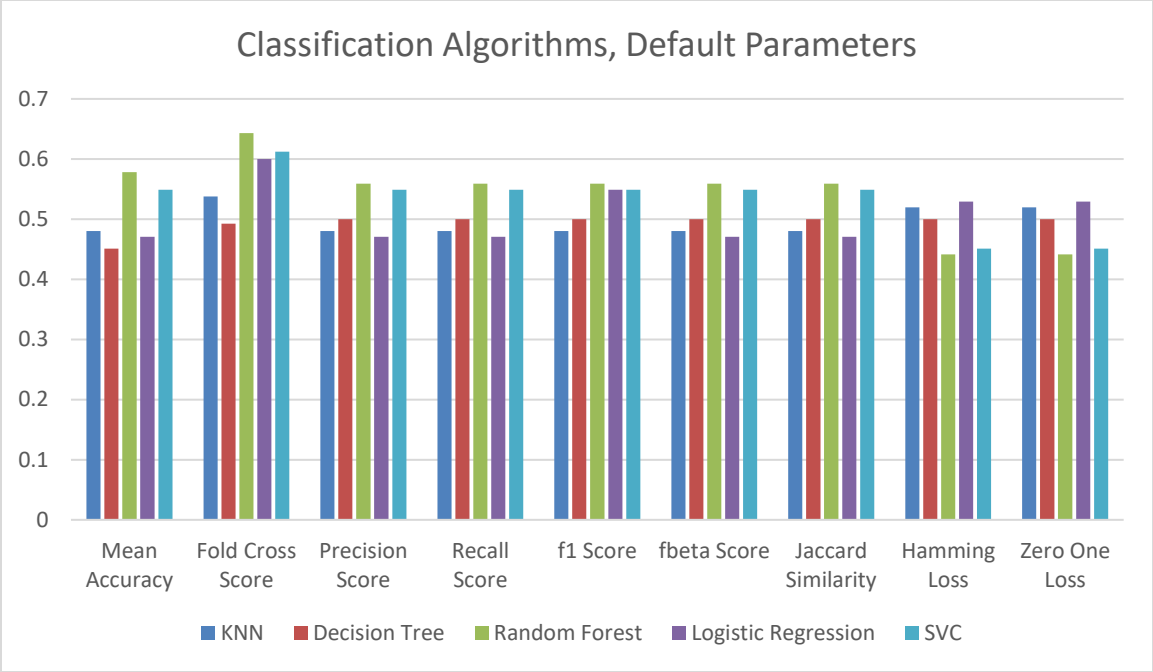
Έχοντας ολοκληρώσει την παράθεση αποτελεσμάτων στα δεδομένα του Facebook που έχουν αξιολογηθεί από το λεξικό του AFINN, προχωρά στα αποτελέσματα των classifiers όταν δέχονται δεδομένα από το Twitter.

	KNN	DT	RF	LR	SVC
Train Time(s)	0.0	0.0	0.0160	0.0150	0.0
Score Time(s)	0.0	0.0	0.0	0.0	0.0
Mean Accuracy	0.4803	0.4509	0.5784	0.4705	0.5490
Fold Cross Score	0.5914	0.4924	0.6432	0.6000	0.6123
Prec Score, micro	0.4803	0.5	0.5588	0.4705	0.5490
Prec Score, macro	0.0878	0.2470	0.4079	0.0894	0.0549
Prec Score, weighted	0.3288	0.5218	0.5256	0.3373	0.3014
Recall Score, micro	0.4803	0.5	0.5588	0.4705	0.5490
Recall Score, macro	0.0968	0.255	0.2124	0.1043	0.1000
Recall Score, weighted	0.4803	0.5	0.5588	0.4705	0.5490
f1 Score, micro	0.4803	0.5	0.5588	0.4705	0.5490
f1 Score, macro	0.0833	0.2366	0.2459	0.0933	0.0708
f1 Score, weighted	0.3807	0.4991	0.4771	0.3895	0.3891
fbeta Score, micro	0.4803	0.5	0.5588	0.4705	0.5490
fbeta Score, macro	0.0826	0.2384	0.3065	0.0901	0.0603
fbeta Score, weighted	0.3439	0.5074	0.4730	0.3555	0.3313
Jaccard Similarity	0.4803	0.5	0.5588	0.4705	0.5490
Hamming Loss	0.5196	0.5	0.4411	0.5294	0.4509
Zero One Loss	0.5196	0.5	0.4411	0.5294	0.4509

Πίνακας 23: Σύγκριση classifiers, λεξικό AFINN, default παράμετροι εισόδου, δεδομένα από Twitter

Τα αποτελέσματα που παρουσιάζουν οι classifiers όταν δέχονται δεδομένα που έχουν εξαχθεί από το Twitter είναι στην πλειοψηφία καλύτερα σε σχέση με αυτά που εξάγονται από το Facebook. Τη μεγαλύτερη αύξηση την πραγματοποίει ο KNN, ενώ ο SVC μαζί με το Random Forest παρουσιάζουν τα βέλτιστα αποτελέσματα. Οι Decision Tree και Random Forest παρουσιάζουν μία ανεκτή διακύμανση.

Στη συνέχεια παρουσιάζω την οπτική απεικόνιση για την απόδοση των classifiers με τις default παραμέτρους. Οι δύο classifiers που ξεχωρίζουν είναι ο Random Forest, ο SVC και ο Logistic Regression, λόγω της υψηλής τιμής στην μετρική του fold cross score. Πρέπει να σημειωθεί ότι στα δεδομένα του Twitter, οι classifiers παρουσιάζουν υψηλότερο fold cross score σε σχέση με τη συμπεριφορά τους όταν δέχονται δεδομένα από το Facebook. Η μετρική του fold cross score θεωρείται η πιο έγκυρη μετρική από αυτές που χρησιμοποιώ.



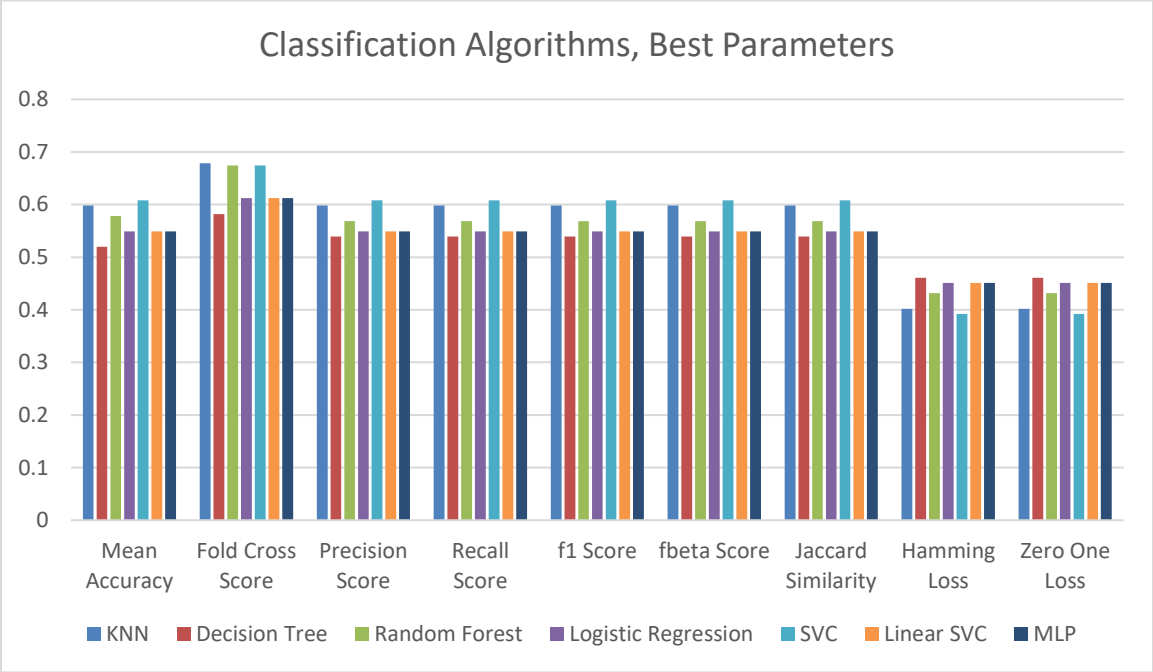
Διάγραμμα 27: Σύγκριση classifiers, λεξικό AFINN, default παράμετροι εισόδου, δεδομένα από Twitter

Στον πίνακα που ακολουθεί έχω παραθέσει τα αποτελέσματα των classifiers όταν εκτελούνται με τις βέλτιστες παραμέτρους εισόδου και τα δεδομένα εισόδου από το Twitter δέχονται προεπεξεργασία.

	KNN	DT	RF	LR	SVC	L. SVC	MLP
Train Time(s)	0.0	0.0	0.0150	0.0840	0.0999	0.0999	0.1019
Score Time(s)	0.0	0.0	0.0	0.0149	0.0	0.0	0.0013
Mean Accuracy	0.5980	0.5196	0.5784	0.5490	0.6078	0.5490	0.5490
Fold Cross Score	0.6786	0.5816	0.6740	0.6123	0.6740	0.6123	0.6123
Prec Score, micro	0.5980	0.5392	0.5686	0.5490	0.6078	0.5490	0.5490
Prec Score, macro	0.5078	0.0717	0.4062	0.0549	0.5583	0.0549	0.0549
Prec Score, weighted	0.6364	0.3181	0.5588	0.3014	0.6830	0.3014	0.3014
Recall Score, micro	0.5980	0.5392	0.5686	0.5490	0.6078	0.5490	0.5490
Recall Score, macro	0.2195	0.1464	0.1892	0.1000	0.2213	0.1000	0.1000
Recall Score, weighted	0.5980	0.5392	0.5686	0.5490	0.6078	0.5490	0.5490
f1 Score, micro	0.5980	0.5392	0.5686	0.5490	0.6078	0.5490	0.5490
f1 Score, macro	0.2546	0.0942	0.2084	0.0708	0.2572	0.0708	0.0708
f1 Score, weighted	0.4936	0.3996	0.4532	0.3891	0.4998	0.3891	0.3891
fbeta Score, micro	0.5980	0.5392	0.5686	0.5490	0.6078	0.5490	0.5490
fbeta Score, macro	0.3313	0.0791	0.2629	0.0603	0.3409	0.0603	0.0603
fbeta Score, weighted	0.4975	0.3464	0.4423	0.3313	0.5086	0.3313	0.3313
Jaccard Similarity	0.5980	0.5392	0.5686	0.5490	0.6078	0.5490	0.5490
Hamming Loss	0.4019	0.4607	0.4313	0.4509	0.3921	0.4509	0.4509
Zero One Loss	0.4019	0.4607	0.4313	0.4509	0.3921	0.4509	0.4509

Πίνακας 24: Σύγκριση classifiers, λεξικό AFINN, βέλτιστοι παράμετροι εισόδου, δεδομένα από Twitter

Από τον παραπάνω πίνακα ξεχωρίζουν τρεις classifiers, ο KNN, Random Forest και ο SVC, ενώ όλοι οι classifiers βελτιώνουν την απόδοση τους.



Διάγραμμα 28: Σύγκριση classifiers, λεξικό AFINN, βέλτιστοι παράμετροι εισόδου

Στην οπτική απεικόνιση της απόδοσης των classifiers ξεχωρίζουν οι τρεις αλγόριθμοι που ανέφερα νωρίτερα και το σκορ στην μετρική του fold cross score θα χαρακτηρίζονταν ικανοποιητικό.

6.4 Αποτελέσματα του λεξικού imdb

Έχοντας ολοκληρώσει την μελέτη των αποτελεσμάτων των classifiers όταν δέχονται δεδομένα τα οποία αξιολογούνται από το λεξικό του AFINN, προχωράω στην μελέτη των classifiers όταν αυτά δέχονται δεδομένα τα οποία έχουν αξιολογηθεί από το λεξικό του imdb. Ακόμη να υπενθυμίσω ότι οι classifiers παρουσίασαν καλύτερα αποτελέσματα στα δεδομένα που έχουν εξαχθεί από το Twitter.

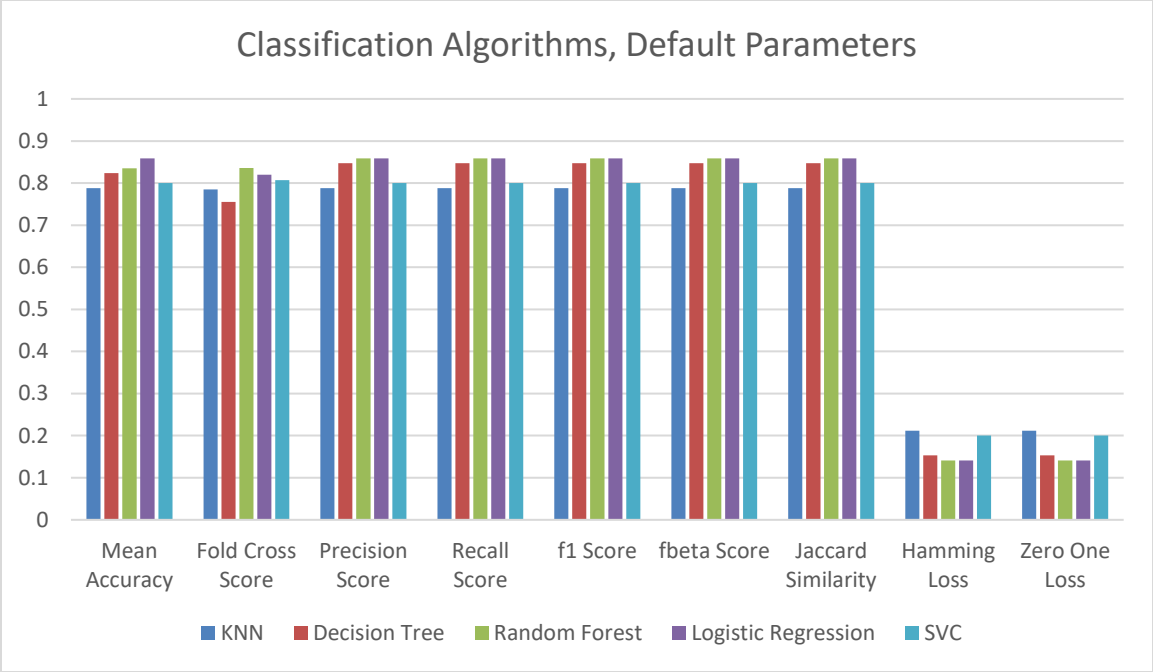
6.4.1 Αποτελέσματα στα δεδομένα που εξαχθεί από το Facebook

Στον πίνακα που ακολουθεί παρουσιάζω τα αποτελέσματα των αλγορίθμων όταν δέχονται δεδομένα τα οποία έχουν εξαχθεί από το Facebook και έχουν αξιολογηθεί από το λεξικό του imdb.

	KNN	DT	RF	LR	SVC
Train Time(s)	0.0013	0.0026	0.0930	0.0930	0.0369
Score Time(s)	0.0054	0.0007	0.0297	0.0008	0.0
Mean Accuracy	0.7882	0.8235	0.8352	0.8588	0.8
Fold Cross Score	0.7847	0.7553	0.8361	0.8198	0.8067
Prec Score, micro	0.7882	0.8470	0.8588	0.8588	0.8
Prec Score, macro	0.2276	0.3612	0.5700	0.5700	0.16
Prec Score, weighted	0.6831	0.8722	0.8564	0.8564	0.64
Recall Score, micro	0.7882	0.8470	0.8588	0.8588	0.8
Recall Score, macro	0.2141	0.2508	0.3199	0.3199	0.2
Recall Score, weighted	0.7882	0.8470	0.8588	0.8588	0.8
f1 Score, micro	0.7882	0.8470	0.8588	0.8588	0.8
f1 Score, macro	0.2067	0.2679	0.3647	0.3647	0.1777
f1 Score, weighted	0.7220	0.8438	0.8219	0.8219	0.7111
fbeta Score, micro	0.7882	0.8470	0.8588	0.8588	0.8
fbeta Score, macro	0.2121	0.2984	0.4402	0.4402	0.1666
fbeta Score, weighted	0.6934	0.8503	0.8242	0.8242	0.6666
Jaccard Similarity	0.7882	0.8470	0.8588	0.8588	0.8
Hamming Loss	0.2117	0.1529	0.1411	0.1411	0.2
Zero One Loss	0.2117	0.1529	0.1411	0.1411	0.2

Πίνακας 25: Σύγκριση classifiers, λεξικό imdb, default παράμετροι εισόδου, δεδομένα από Facebook

Τα αποτελέσματα με τη χρήση του λεξικού του imdb είναι εμφανώς καλύτερα σε σχέση με τα αποτελέσματα που είχα εξάγει με τη χρήση του λεξικού. Στους classifiers Decision Tree και Random Forest, παρατηρείται μία διακύμανση ±3% και κρίνοντας από τη μετρική του fold cross score η κατάταξη στην απόδοση των classifiers παραμένει ίδια, καλύτερο σκορ παρουσιάζει ο Random Forest και χειρότερο ο Decision Tree. Στο παρακάτω διάγραμμα φαίνεται η σύγκριση στην απόδοση του Random Forest με τη χρήση διαφορετικού λεξικού.



Διάγραμμα 29: Σύγκριση classifiers, λεξικό AFINN, default παράμετροι εισόδου, δεδομένα από Facebook

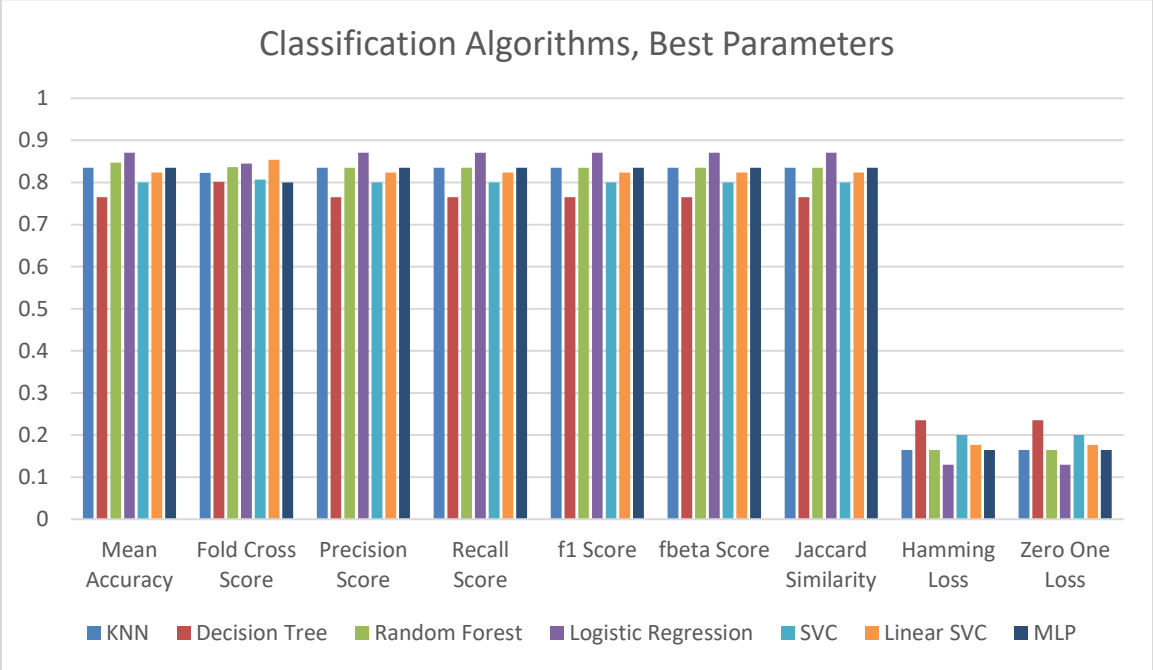
Στην οπτική απεικόνιση της απόδοσης των classifiers όταν δέχονται δεδομένα τα οποία αξιολογούνται από το λεξικό του imdb, διακρίνεται η υψηλή απόδοση όλων των classifiers. Random Forest, Logistic Regression και SVC είναι οι αλγόριθμοι οι οποίοι ξεχωρίζουν ελαφρώς από άποψης απόδοσης. Τα αποτελέσματα των classifiers όταν βελτιστοποιούνται διακρίνονται στον πίνακα που ακολουθεί.

	KNN	DT	RF	LR	SVC	L. SVC	MLP
Train Time(s)	0.0	0.0	0.031	0.4839	0.0369	0.0379	0.1085
Score Time(s)	0.0	0.0	0.0	0.0	0.0	0.0	0.0015
Mean Accuracy	0.8352	0.7647	0.8470	0.8705	0.8	0.8235	0.7058
Fold Cross Score	0.8227	0.8016	0.8324	0.8450	0.8067	0.8542	0.7566
Prec Score, micro	0.8352	0.7647	0.8352	0.8705	0.8	0.8235	0.8000
Prec Score, macro	0.3275	0.2509	0.3658	0.4524	0.16	0.2991	0.16
Prec Score, weighted	0.7641	0.7382	0.7810	0.8621	0.64	0.7418	0.64
Recall Score, micro	0.8352	0.7647	0.8352	0.8705	0.8	0.8235	0.8
Recall Score, macro	0.2770	0.3105	0.2600	0.3142	0.2	0.2399	0.2
Recall Score, weighted	0.8352	0.7647	0.8352	0.8705	0.8	0.8235	0.8
f1 Score, micro	0.8352	0.7647	0.8352	0.8705	0.8	0.8235	0.8
f1 Score, macro	0.2877	0.2718	0.2736	0.3544	0.1777	0.2428	0.1777
f1 Score, weighted	0.787	0.7462	0.7796	0.8515	0.7111	0.7615	0.7111
fbeta Score, micro	0.8352	0.7647	0.8352	0.8705	0.8	0.8235	0.8000
fbeta Score, macro	0.3060	0.2579	0.3080	0.4008	0.1666	0.2626	0.1666
fbeta Score, weighted	0.7691	0.7406	0.7670	0.8521	0.6666	0.7403	0.6666
Jaccard Similarity	0.8352	0.7647	0.8352	0.8705	0.8	0.8235	0.8
Hamming Loss	0.1647	0.2352	0.1647	0.1294	0.2	0.1764	0.2
Zero One Loss	0.1647	0.2352	0.1647	0.1294	0.2	0.1764	0.2

Πίνακας 26: Σύγκριση classifiers, λεξικό imdb, βέλτιστοι παράμετροι εισόδου, δεδομένα από Facebook

Όπως και στην περίπτωση που τα δεδομένα αξιολογήθηκαν από το λεξικό του AFINN, έτσι και στην περίπτωση που αξιολογήθηκαν από το λεξικό το imdb, οι classifiers των Linear SVC και MLP παρουσίασαν πολύ μεγάλη διακύμανση με τη χρήση των default παραμέτρων. Για αυτό το λόγο ανέφερα την απόδοση τους μόνο όταν σταθεροποιήθηκαν.

Από όλους τους classifiers το υψηλότερο σκορ στην μετρική του fold cross score επιτεύχθηκε από τον classifier του Logistic Regression. Σε γενικές γραμμές μπορεί να ειπωθεί ότι όλοι οι classifiers παρουσιάζουν σχετικά παρόμοια απόδοση και πολύ καλύτερη σε σχέση με την περίπτωση που τα δεδομένα εισόδου αξιολογούνται από το λεξικό του AFINN.



Διάγραμμα 30: Σύγκριση classifiers, λεξικό imdb, βέλτιστοι παράμετροι εισόδου, δεδομένα από Facebook

Τα αποτελέσματα των classifiers θεωρούνται παραπάνω ικανοποιητικά, καθώς πέντε στους έξι classifiers έχουν σκορ υψηλότερο από 0.8 στην μετρική του fold cross score. Σε προβλήματα επεξεργασίας φυσικής γλώσσα το 0.8 θεωρείται από τις υψηλότερες τιμές που μπορεί να πετύχει ένας classifier, λόγω των ιδιαιτεροτήτων που χαρακτηρίζουν την επεξεργασία φυσικής γλώσσας.

Το λεξικό του imdb είναι ιδανικό για το είδος των δεδομένων που έχω συλλέξει και άμα πραγματοποιηθούν κάποιες αλλαγές σε αυτό, ενδέχεται τα αποτελέσματα των classifiers να γίνουν εντυπωσιακά υψηλά.

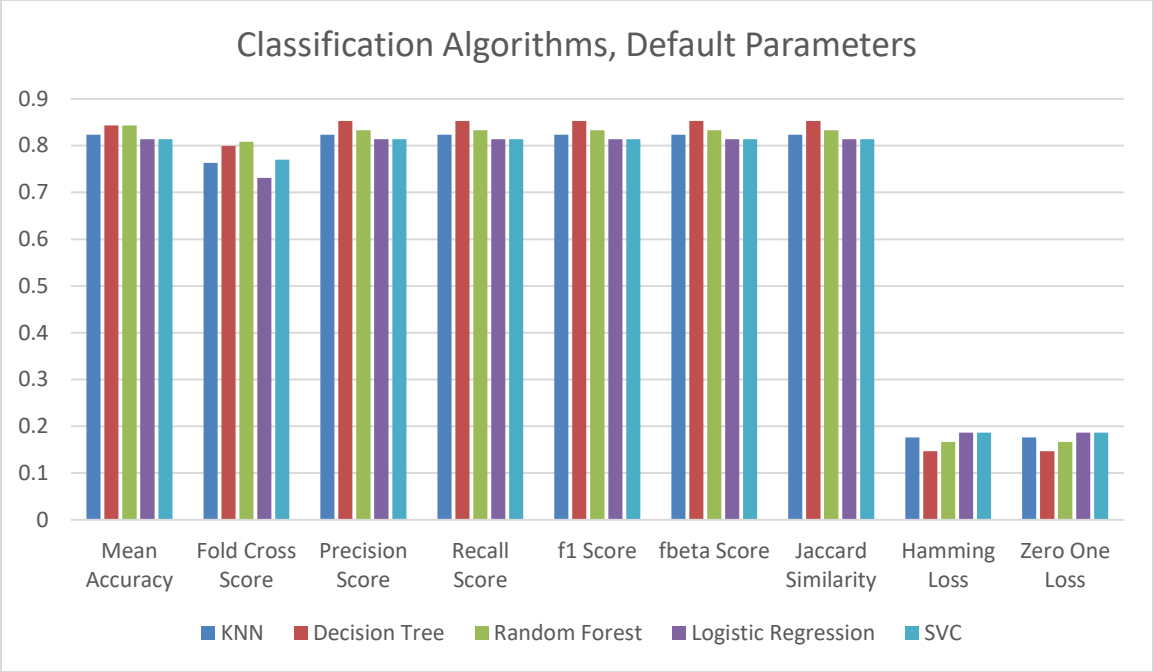
6.4.2 Αποτελέσματα στα δεδομένα που εξαχθεί από το Twitter

Έχοντας ολοκληρώσει την ανάλυση των αλγορίθμων στα δεδομένα του Facebook, κάνοντας χρήση του λεξικού από το imdb, προχωράω στην ανάλυση των δεδομένων από το Twitter.

	KNN	DT	RF	LR	SVC
Train Time(s)	0.001	0.0022	0.0521	0.0512	0.0309
Score Time(s)	0.0047	0.0006	0.0109	0.0009	0.0
Mean Accuracy	0.8235	0.8431	0.8431	0.8137	0.8137
Fold Cross Score	0.7633	0.7997	0.8081	0.731	0.7702
Prec Score, micro	0.8235	0.8529	0.8333	0.8137	0.8137
Prec Score, macro	0.2989	0.5959	0.6884	0.3877	0.3623
Prec Score, weighted	0.7312	0.8269	0.824	0.7919	0.7507
Recall Score, micro	0.8235	0.8529	0.8333	0.8137	0.8137
Recall Score, macro	0.2399	0.5552	0.4801	0.2021	0.22
Recall Score, weighted	0.8235	0.8529	0.8333	0.8137	0.8137
f1 Score, micro	0.8235	0.8529	0.8333	0.8137	0.8137
f1 Score, macro	0.2427	0.5518	0.5052	0.2148	0.2155
f1 Score, weighted	0.7585	0.8299	0.7931	0.7572	0.7382
fbeta Score, micro	0.8235	0.8529	0.8333	0.8137	0.8137
fbeta Score, macro	0.2624	0.5645	0.5564	0.2581	0.2401
fbeta Score, weighted	0.7341	0.8226	0.7875	0.7468	0.7132
Jaccard Similarity	0.8235	0.8529	0.8333	0.8137	0.8137
Hamming Loss	0.1764	0.147	0.1666	0.1862	0.1862
Zero One Loss	0.1764	0.147	0.1666	0.1862	0.1862

Πίνακας 27: Σύγκριση classifiers, λεξικό imdb, default παράμετροι εισόδου, δεδομένα από Twitter

Παρατηρώντας τα σκορ των μετρικών των αλγορίθμων με τη χρήση του λεξικού από το imdb στα δεδομένα του Twitter, τα αποτελέσματα είναι σαφώς ανώτερα αν συγκριθούν με αυτά που έχουν μετρηθεί με τη χρήση του λεξικού AFINN. Σε σχέση με τα δεδομένα που έχω εξάγει από το Facebook, μόνο ο Logistic Regression διαφοροποιείται σημαντικά καθώς παρατηρείται πτώση μέχρι και 7%, ενώ οι υπόλοιποι classifiers έχουν μικρές διαφοροποιήσεις.



Διάγραμμα 31: Σύγκριση classifiers, λεξικό imdb, default παράμετροι εισόδου, δεδομένα από Twitter

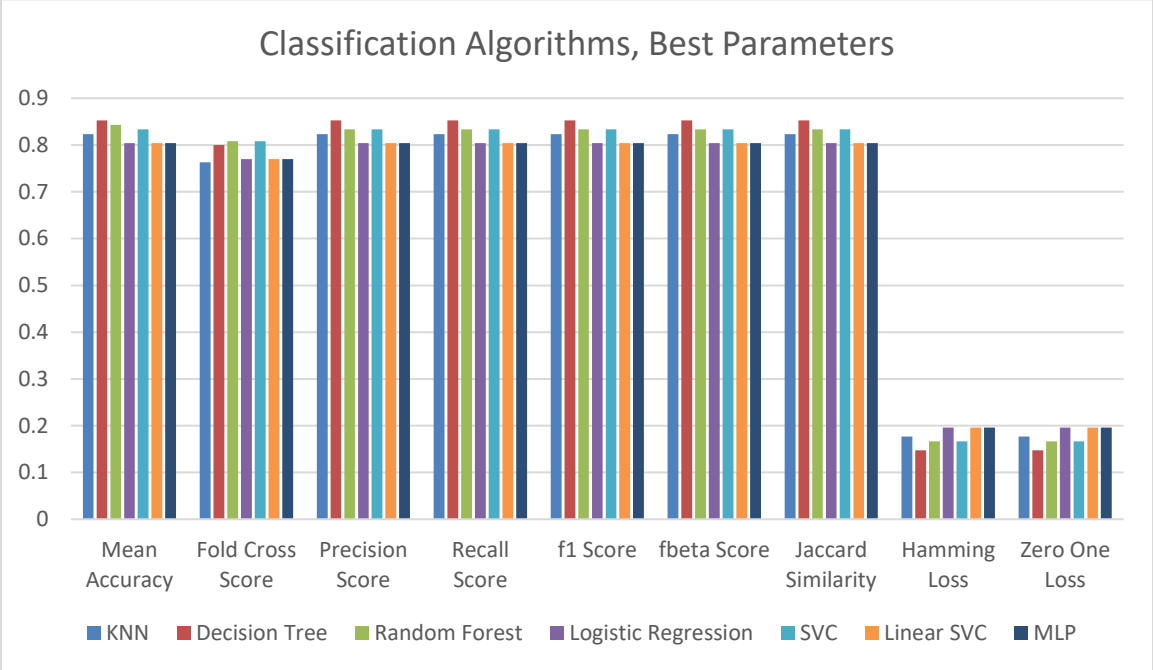
Από την οπτική απεικόνιση της απόδοσης των classifiers δεν διακρίνεται κάποιος από πλευράς απόδοσης, καθώς όλοι, πλην του Logistic Regression, παρουσιάζουν παρόμοια σκορ σε όλες τις μετρικές. Στον επόμενο πίνακα παρουσιάζω τους αλγορίθμους με τις βέλτιστες παραμέτρους εισόδου και με τα δεδομένα εισόδου να έχουν υποστεί προεπεξεργασία.

	KNN	DT	RF	LR	SVC	L. SVC	MLP
Train Time(s)	0.0010	0.0022	0.0521	0.0512	0.101	0.0	0.0498
Score Time(s)	0.0047	0.0006	0.0109	0.0009	0.0	0.0	0.0013
Mean Accuracy	0.8235	0.8431	0.8431	0.8137	0.8333	0.8039	0.8039
Fold Cross Score	0.7633	0.7997	0.8081	0.731	0.8083	0.7702	0.7702
Prec Score, micro	0.8235	0.8529	0.8333	0.8137	0.8333	0.8039	0.8039
Prec Score, macro	0.2989	0.5959	0.6884	0.3877	0.7656	0.1607	0.1607
Prec Score, weighted	0.7312	0.8269	0.8240	0.7919	0.8521	0.6462	0.6462
Recall Score, micro	0.8235	0.8529	0.8333	0.8137	0.8333	0.8039	0.8039
Recall Score, macro	0.2399	0.5552	0.4801	0.2021	0.445	0.2	0.2
Recall Score, weighted	0.8235	0.8529	0.8333	0.8137	0.8333	0.8039	0.8039
f1 Score, micro	0.8235	0.8529	0.8333	0.8137	0.8333	0.8039	0.8039
f1 Score, macro	0.2427	0.5518	0.5052	0.2148	0.462	0.1782	0.1782
f1 Score, weighted	0.7585	0.8299	0.7931	0.7572	0.7734	0.7165	0.7165
fbeta Score, micro	0.8235	0.8529	0.8333	0.8137	0.8333	0.8039	0.8039
fbeta Score, macro	0.2624	0.5645	0.5564	0.2581	0.5263	0.1673	0.1673
fbeta Score, weighted	0.7341	0.8226	0.7875	0.7468	0.7670	0.6726	0.6726
Jaccard Similarity	0.8235	0.8529	0.8333	0.8137	0.8333	0.8039	0.8039
Hamming Loss	0.1764	0.147	0.1666	0.1862	0.1666	0.196	0.196
Zero One Loss	0.1862	0.196	0.1960	0.196	0.1666	0.196	0.196

Πίνακας 28: Σύγκριση classifiers, λεξικό imdb, βέλτιστοι παράμετροι εισόδου, δεδομένα από Twitter

Οι classifiers Linear SVC και MLP παρουσιάζονται μόνο όταν τα δεδομένα εισόδου δέχονται προεπεξεργασία, γιατί ειδάλλως παρουσιάζουν μεγάλη διακύμανση. Η διακύμανση αυτών των δύο classifiers είναι χαρακτηριστική για τα δεδομένα που έχω συλλέξει, οπότε όταν δεν αναφέρω τα αποτελέσματα τους με τη χρήση των default παραμέτρων εισόδου, παρουσιάζουν υψηλή διακύμανση, η οποία κάνει την αδύνατη την αξιολόγηση τους.

Οι υπόλοιποι classifiers δεν παρουσίασαν διαφορές στην απόδοση είτε εκτελούνται με τις default παραμέτρους εισόδου είτε με τις παραμέτρους εισόδου που προτείνονται από το Grid Search.



Διάγραμμα 32: Σύγκριση classifiers, λεξικό imdb, βέλτιστοι παράμετροι εισόδου

Παρατηρώντας το διάγραμμα με τη σύγκριση της απόδοσης των 6 classifiers, δεν ξεχωρίσει κάποιος αλγόριθμος από τους υπόλοιπους. Οι αλγόριθμοι του Decision Tree και Random Forest εμφανίζουν να προηγούνται ελάχιστα σε απόδοση με τους υπόλοιπους, αλλά αν λάβουμε υπόψιν και τη διακύμανση των δύο συγκεκριμένων αλγορίθμων αυτή η διαφορά γίνεται ασήμαντη. Το σημείο που πρέπει να τονιστεί είναι ότι οι απλοί classifiers εμφανίζουν βέλτιστα αποτελέσματα, απλά με τη χρήση των default παραμέτρων.

6.5 Αποτελέσματα του λεξικού Amazon/TripAdvisor

Το τρίτο λεξικό του οποίου μετράω την αποτελεσματικότητα στην ανάγνωση δεδομένων προερχομένων από κοινωνικά δίκτυα είναι το λεξικό του Amazon/TripAdvisor. Θα ακολουθήσω την ίδια δομή στην παρουσίαση των αποτελεσμάτων που ακολούθησα και στα προηγούμενα λεξικά.

6.5.1 Αποτελέσματα στα δεδομένα που εξαχθεί από το Facebook

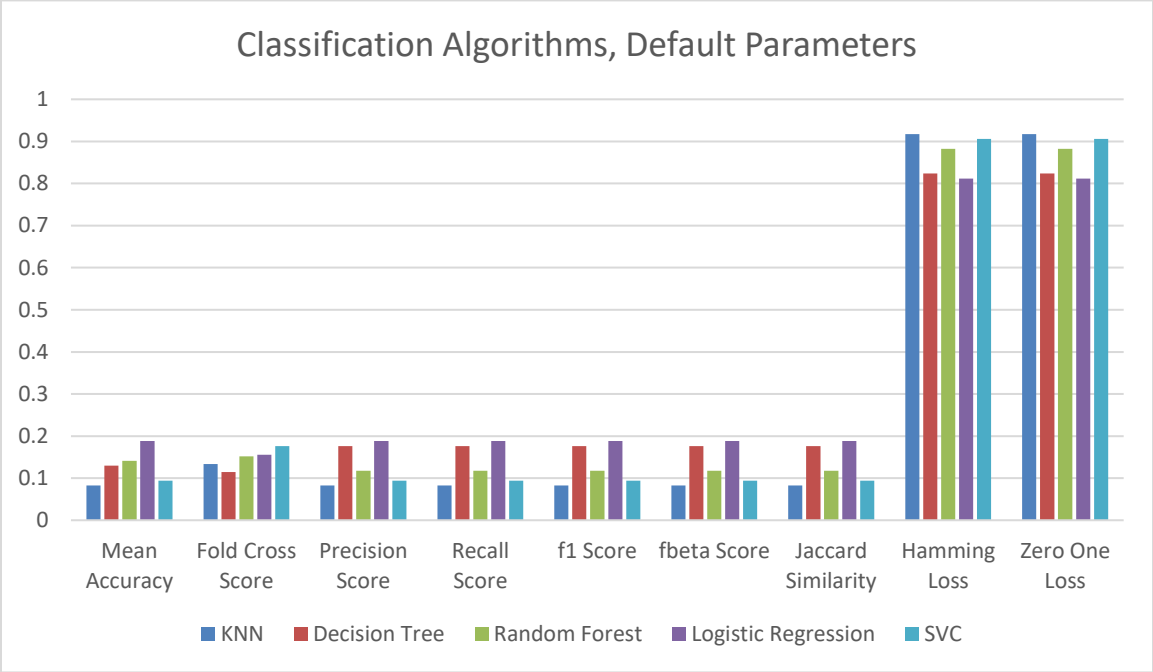
Πρώτα παρουσιάζω την απόδοση των classifiers όταν δέχονται δεδομένα από το Facebook και εκτελούνται με τη χρήση των default παραμέτρων εισόδου.

	KNN	DT	RF	LR	SVC
Train Time(s)	0.0	0.016	0.0399	0.231	0.019
Score Time(s)	0.0	0.0	0.002	0.0	0.0029
Mean Accuracy	0.0941	0.1529	0.1882	0.1058	0.0941
Fold Cross Score	0.1329	0.1381	0.2122	0.1405	0.1763
Prec Score, micro	0.0941	0.1529	0.1764	0.1058	0.0941
Prec Score, macro	0.0174	0.0726	0.1243	0.0528	0.0043
Prec Score, weighted	0.0381	0.2226	0.2683	0.1441	0.01
Recall Score, micro	0.0941	0.1529	0.1764	0.1058	0.0941
Recall Score, macro	0.0512	0.0653	0.0962	0.051	0.0404
Recall Score, weighted	0.0941	0.1529	0.1764	0.1058	0.0941
f1 Score, micro	0.0941	0.1529	0.1764	0.1058	0.0941
f1 Score, macro	0.0235	0.0554	0.0764	0.0438	0.0078
f1 Score, weighted	0.0461	0.146	0.1252	0.0942	0.0182
fbeta Score, micro	0.0941	0.1529	0.1764	0.1058	0.0941
fbeta Score, macro	0.0190	0.0597	0.0851	0.0442	0.0052
fbeta Score, weighted	0.0398	0.1702	0.1444	0.1033	0.0122
Jaccard Similarity	0.0941	0.1529	0.1764	0.1058	0.0941
Hamming Loss	0.9058	0.847	0.8235	0.8941	0.9058
Zero One Loss	0.9058	0.847	0.8235	0.8941	0.9058

Πίνακας 29: Σύγκριση classifiers, λεξικό Amazon/TripAdvisor, default παράμετροι εισόδου, δεδομένα από Facebook

Τα αποτελέσματα των classifiers είναι απογοητευτικά. Με βάση τη συμπεριφορά που είχαν οι classifiers όταν τα δεδομένα εισόδου αξιολογήθηκαν από το λεξικό του imdb, περίμενα το συγκεκριμένο λεξικό να παρουσιάσει παρόμοια απόδοση. Αλλά

αποδεικνύεται ότι δεν παίζει μεγάλο ρόλο η βασική ιδέα πίσω από την οποία δημιουργείται ένα λεξικό, αλλά τα οι εγγραφές του λεξικού.



Διάγραμμα 33: Σύγκριση classifiers, λεξικό Amazon/TripAdvisor, default παράμετροι εισόδου, δεδομένα από Facebook

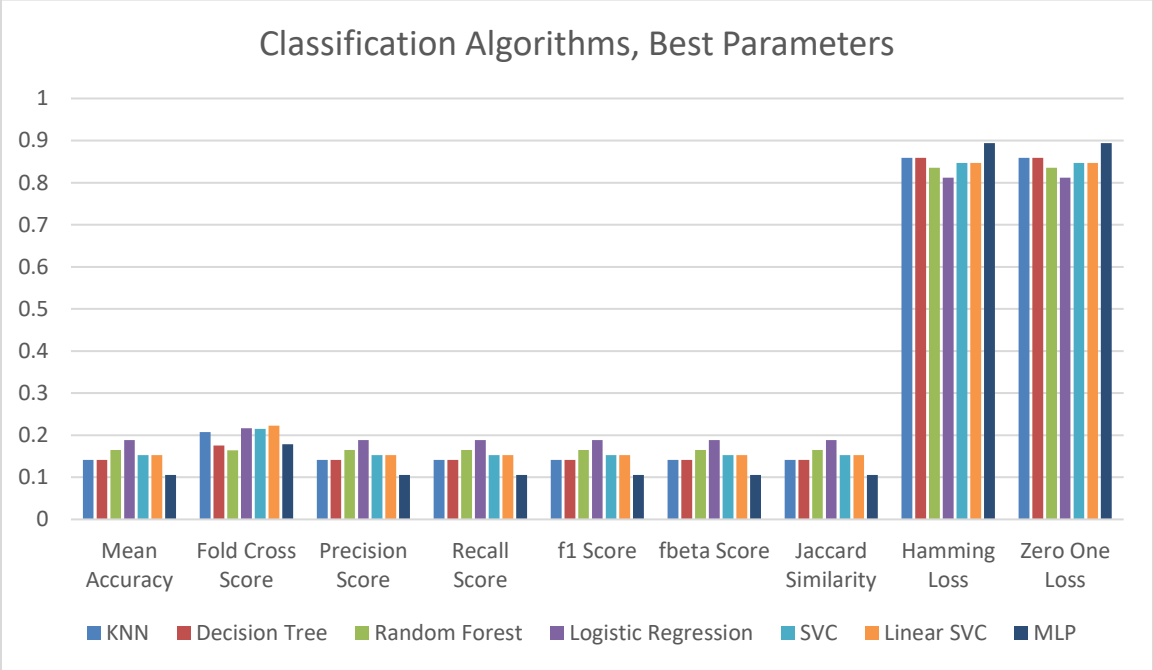
Στην οπτική απεικόνιση της απόδοσης των αλγορίθμων φαίνεται για μια ακόμη φορά η απογοητευτική απόδοση των classifiers. Δεν μπορώ να ξεχωρίζω κάποιον αλγόριθμο, γιατί κανένας αλγόριθμος δεν παρουσιάζει έστω ανεκτή απόδοση. Παρόλο αυτά εκτελώ Grid Search και προεπεξεργασία των δεδομένων εισόδου και παραθέτω τα αποτελέσματα.

	KNN	DT	RF	LR	SVC	L. SVC	MLP
Train Time(s)	0.0039	0.0	0.012	0.256	0.084	0.0080	1.6789
Score Time(s)	0.059	0.0	0.0009	0.0009	0.003	0.0009	0.002
Mean Accuracy	0.1411	0.1529	0.1764	0.1882	0.1529	0.1529	0.1176
Fold Cross Score	0.2077	0.1754	0.164	0.2167	0.2151	0.2223	0.1784
Prec Score, micro	0.1411	0.1411	0.1647	0.1882	0.1529	0.1529	0.1058
Prec Score, macro	0.0258	0.0166	0.0577	0.1044	0.1115	0.0337	0.0055
Prec Score, weighted	0.0404	0.04	0.0958	0.2896	0.2474	0.0682	0.0128
Recall Score, micro	0.1411	0.1411	0.1647	0.1882	0.1529	0.1529	0.1058
Recall Score, macro	0.0746	0.0652	0.0847	0.0965	0.0819	0.0779	0.0454
Recall Score, weighted	0.1411	0.1411	0.1647	0.1882	0.1529	0.1529	0.1058
f1 Score, micro	0.1411	0.1411	0.1647	0.1882	0.1529	0.1529	0.1058
f1 Score, macro	0.0349	0.0260	0.0577	0.0757	0.0617	0.0426	0.0098
f1 Score, weighted	0.0569	0.0609	0.0989	0.1608	0.0999	0.0811	0.0229
fbeta Score, micro	0.1411	0.1411	0.1647	0.1882	0.1529	0.1529	0.1058
fbeta Score, macro	0.0286	0.0194	0.0567	0.075	0.0724	0.0361	0.0067
fbeta Score, weighted	0.0453	0.0462	0.0948	0.1746	0.1237	0.1529	0.0156
Jaccard Similarity	0.1411	0.1411	0.1647	0.1882	0.1529	0.1529	0.1058
Hamming Loss	0.8588	0.8588	0.8352	0.8117	0.8470	0.847	0.8941

Zero One Loss	0.8588	0.8588	0.8352	0.8117	0.8470	0.847	0.8941

Πίνακας 30: Σύγκριση classifiers, λεξικό Amazon/TripAdvisor, βέλτιστοι παράμετροι εισόδου, δεδομένα από Facebook

Οι classifiers παρουσιάζουν μια μικρή βελτίωση όταν δέχονται τις προτεινόμενες παραμέτρους από το Grid Search, αλλά δεν είναι αρκετή ώστε να θεωρηθεί αποδεκτή η απόδοση τους.



Διάγραμμα 34: Σύγκριση classifiers, λεξικό Amazon/TripAdvisor, βέλτιστοι παράμετροι εισόδου, δεδομένα από Facebook

Από την οπτική απεικόνιση των classifiers φαίνεται για μια ακόμη φορά η κακή απόδοση τους. Κανένας αλγόριθμος δεν παρουσιάζει έστω αποδεκτό σκορ ούτε σε μία μετρική.

Η εξήγηση που δίνω για την τόσο κακή απόδοση του λεξικού, σε όλους τους classifiers, έγκειται στον τρόπο βαθμολογίας του λεξικού που έχω θέσει εγώ. Σε αντίθεση με το λεξικό του imdb, όπου οι λέξεις βαθμολογούνται στην κλίμακα [1,10] στο λεξικό του Amazon/TripAdvisor αξιολογούνται στην κλίμακα [1,5]. Η διαφορά στο εύρος τιμών σε συνδυασμό με τις μέσες τιμές που έχει το κάθε λεξικό, 3.3 το λεξικό του Amazon/TripAdvisor και 5.7 το λεξικό του imdb, δημιουργούν διαφορετική κατανομή στην αξιολόγηση των προτάσεων. Στο λεξικό του Amazon/TripAdvisor υπάρχει μεγάλη συγκέντρωση στις τιμές [0,5] εν μέρει εξαιτίας της βαθμολογίας των λέξεων που αποδίδεται από το λεξικό και εν μέρει επειδή υπάρχουν προτάσεις στις οποίες δεν αναγνωρίζεται ούτε μία λέξη. Αντίθετα στο λεξικό του imdb στις περιπτώσεις που δεν υπάρχει ούτε μία λέξη στην πρόταση προς ανάλυση η πρόταση βαθμολογείται με μηδέν και δεν συσσωρεύεται στο σύνολο των προτάσεων που αξιολογούνται από το λεξικό.

6.5.2 Αποτελέσματα στα δεδομένα που εξαχθεί από το Twitter

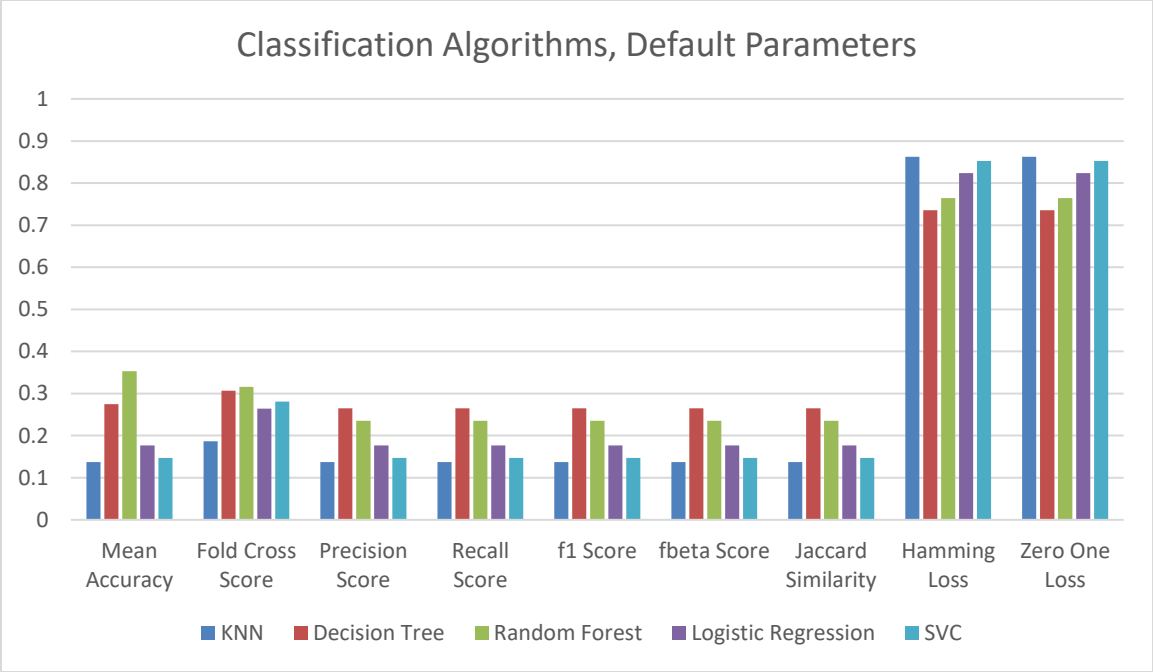
Προχωρώ στην μελέτη των classifiers όταν δέχονται δεδομένα από το Twitter τα οποία έχουν αξιολογηθεί από το λεξικό του Amazon/TripAdvisor. Από τη συμπεριφορά των classifiers στα προηγούμενα λεξικά περιμένω οι αλγόριθμοι να παρουσιάσουν ελαφρώς καλύτερη απόδοση.

	KNN	DT	RF	LR	SVC
Train Time(s)	0.0009	0.002	0.023	0.0439	0.022
Score Time(s)	0.002	0.0009	0.001	0.0	0.0049
Mean Accuracy	0.1372	0.2745	0.3529	0.1764	0.147
Fold Cross Score	0.1867	0.3065	0.3157	0.2641	0.2812
Prec Score, micro	0.1372	0.2647	0.2352	0.1764	0.147
Prec Score, macro	0.0732	0.1676	0.1949	0.0909	0.104
Prec Score, weighted	0.1385	0.3002	0.3072	0.1978	0.2256
Recall Score, micro	0.1372	0.2647	0.2352	0.1764	0.147
Recall Score, macro	0.0798	0.1441	0.1516	0.0898	0.0898
Recall Score, weighted	0.1372	0.2647	0.2352	0.1764	0.1470
f1 Score, micro	0.1372	0.2647	0.2352	0.1764	0.1470
f1 Score, macro	0.0646	0.1427	0.1532	0.0826	0.0491
f1 Score, weighted	0.1124	0.2636	0.2398	0.1727	0.0771
fbeta Score, micro	0.1372	0.2647	0.2352	0.1764	0.147
fbeta Score, macro	0.0671	0.152	0.1704	0.0859	0.0535
fbeta Score, weighted	0.1218	0.2784	0.2688	0.1846	0.0915
Jaccard Similarity	0.1372	0.2647	0.2352	0.1764	0.147
Hamming Loss	0.8627	0.7352	0.7647	0.8235	0.8529
Zero One Loss	0.8627	0.7352	0.7647	0.8235	0.8529

Πίνακας 31: Σύγκριση classifiers, λεξικό Amazon/TripAdvisor, default παράμετροι εισόδου, δεδομένα από Twitter

Οι classifiers παρουσιάζουν καλύτερα αποτελέσματα όταν δέχονται δεδομένα από το Twitter, με τους Decision Tree και Random Forest να ξεχωρίζουν. Συνεχίζουν όμως οι αλγόριθμοι να μην μπορούν να παρουσιάσουν έστω ανεκτή απόδοση.

Παραθέτω οπτική απεικόνιση της απόδοσης των classifiers όταν εκτελούνται με τις default παραμέτρους εισόδου και δέχονται δεδομένα τα οποία έχουν εξαχθεί από το Twitter.



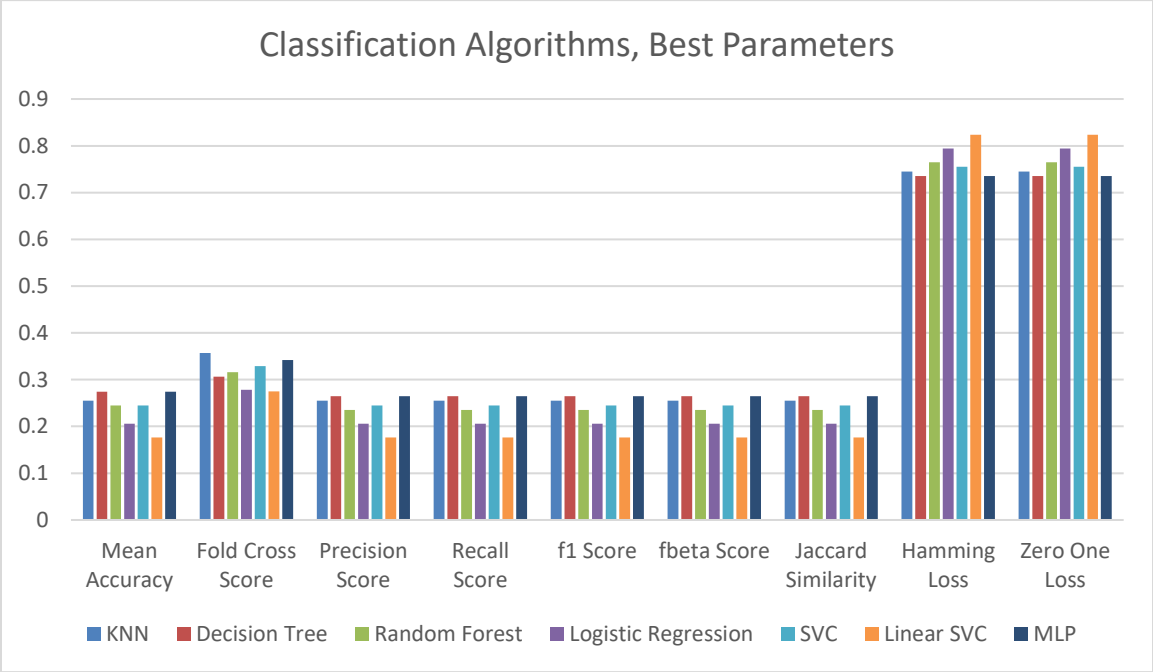
Διάγραμμα 35: Σύγκριση classifiers, λεξικό Amazon/TripAdvisor, default παράμετροι εισόδου, δεδομένα από Twitter

Στο διάγραμμα σύγκρισης φαίνεται ότι ο Random Forest ξεχωρίζει σε απόδοση, αλλά το σκορ του δεν είναι αρκετά υψηλό ώστε το λεξικό να θεωρηθεί αποδεκτό για την αξιολόγηση των δεδομένων που έχω συλλέξει. Στον πίνακα που ακολουθεί παραθέτω τα σκορ των classifiers όταν εκτελούνται με τις βέλτιστες παραμέτρους.

	KNN	DT	RF	LR	SVC	L. SVC	MLP
Train Time(s)	0.001	0.002	0.023	0.1319	0.083	0.016	2.1052
Score Time(s)	0.0039	0.0009	0.001	0.0	0.0019	0.0009	0.0032
Mean Accuracy	0.2549	0.2745	0.2450	0.2058	0.245	0.1764	0.2745
Fold Cross Score	0.3572	0.3065	0.3157	0.2786	0.3291	0.2752	0.3420
Prec Score, micro	0.2549	0.2647	0.2352	0.2058	0.245	0.1764	0.2647
Prec Score, macro	0.315	0.1676	0.1949	0.1606	0.4328	0.1097	0.1907
Prec Score, weighted	0.3564	0.3002	0.3072	0.1963	0.5246	0.1704	0.2754
Recall Score, micro	0.2549	0.2647	0.2352	0.2058	0.245	0.1764	0.2647
Recall Score, macro	0.2003	0.1441	0.1516	0.142	0.1911	0.1079	0.1905
Recall Score, weighted	0.2549	0.2647	0.2352	0.2058	0.245	0.1764	0.2647
f1 Score, micro	0.2549	0.2647	0.2352	0.2058	0.245	0.1764	0.2647
f1 Score, macro	0.2058	0.1427	0.1532	0.1211	0.1961	0.0975	0.1831
f1 Score, weighted	0.2396	0.2636	0.2398	0.1682	0.2195	0.1565	0.2579
fbeta Score, micro	0.2549	0.2647	0.2352	0.2058	0.245	0.1764	0.2647
fbeta Score, macro	0.2474	0.1520	0.1704	0.1313	0.2703	0.1019	0.1860
fbeta Score, weighted	0.2827	0.2784	0.2688	0.1738	0.3084	0.1612	0.2656
Jaccard Similarity	0.2549	0.2647	0.2352	0.2058	0.245	0.1764	0.2647
Hamming Loss	0.7450	0.7352	0.7647	0.7941	0.7549	0.8235	0.7352
Zero One Loss	0.7450	0.7352	0.7647	0.7941	0.7549	0.8235	0.7352

Πίνακας 32: Σύγκριση classifiers, λεξικό Amazon/TripAdvisor, βέλτιστοι παράμετροι εισόδου, δεδομένα από Twitter

Τα σκορ των classifiers έχουν έρθει πολύ κοντά μεταξύ τους πλέον και δεν υπάρχει κάποιος αλγόριθμος που να ξεχωρίζει από πλευράς απόδοσης. Στο διάγραμμα που ακολουθεί παραθέτω και μία οπτική απεικόνιση της απόδοσης των classifiers.



Διάγραμμα 36: Σύγκριση classifiers, λεξικό AFINN, βέλτιστοι παράμετροι εισόδου

Από το διάγραμμα απόδοσης των classifiers δεν ξεχωρίζει κάποιος classifiers. ο KNN παρουσιάζει την υψηλότερη τιμή στην μετρική του fold cross score, καθώς επίσης καλές τιμές σε όλες τις υπόλοιπες μετρικές και μηδενική διακύμανση. Στη συνέχεια θα πρότεινα τους αλγορίθμους του SVC και MLP.

Παρόλη την προσπάθεια που κατέβαλλα για βελτίωση της απόδοσης των classifiers, το λεξικό του Amazon/TripAdvisor δεν ξεπερνάει το 50%, οπότε δεν συνιστάται στα δεδομένα που έχω συλλέξει.

6.6 Αποτελέσματα του λεξικού Goodreads

Σε αυτό το υποκεφάλαιο το λεξικό που μελετώ αξιοποιεί τους πόρους από την ιστοσελίδα Goodreads. Το λεξικό έχει δημιουργηθεί με τον ίδιο τρόπο που δημιουργήθηκε το λεξικό του Amazon/TripAdvisor, οπότε περιμένω ομοιότητες στη συμπεριφορά των classifiers.

6.6.1 Αποτελέσματα στα δεδομένα που εξαχθεί από το Facebook

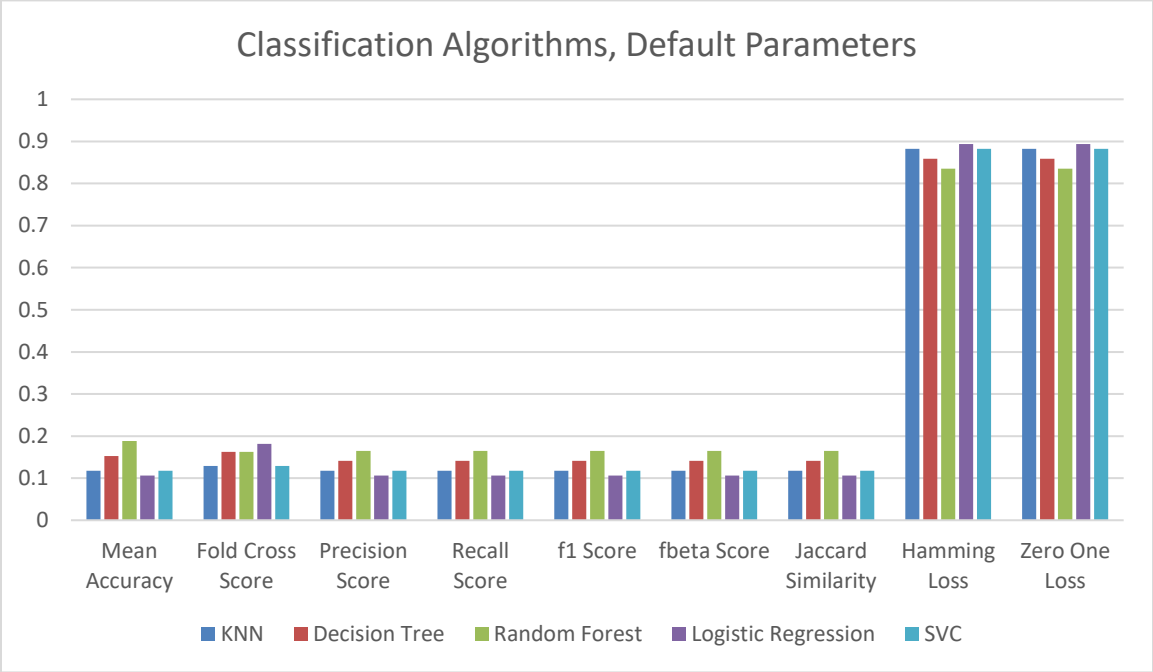
Προχωράω στην παρουσίαση των αποτελεσμάτων των classifiers όταν τα δεδομένα του Facebook έχουν αξιολογηθεί από το λεξικό του Goodreads.

	KNN	DT	RF	LR	SVC
Train Time(s)	0.0	0.0039	0.0049	0.138	0.0
Score Time(s)	0.0	0.001	0.015	0.0	0.0
Mean Accuracy	0.1176	0.1529	0.1882	0.1058	0.1176
Fold Cross Score	0.1292	0.1627	0.1628	0.1816	0.1292
Prec Score, micro	0.1176	0.1411	0.1647	0.1058	0.1176
Prec Score, macro	0.0372	0.0554	0.0641	0.0685	0.0372
Prec Score, weighted	0.0915	0.1383	0.1692	0.19	0.0915
Recall Score, micro	0.1176	0.1411	0.1647	0.1058	0.1176
Recall Score, macro	0.0527	0.0596	0.0668	0.0443	0.0527
Recall Score, weighted	0.1176	0.1411	0.1647	0.1058	0.1176
f1 Score, micro	0.1176	0.1411	0.1647	0.1058	0.1176
f1 Score, macro	0.0357	0.0565	0.0574	0.0348	0.0357
f1 Score, weighted	0.0832	0.1372	0.1464	0.086	0.0832
fbeta Score, micro	0.1176	0.1411	0.1647	0.1058	0.1176
fbeta Score, macro	0.0349	0.0556	0.0594	0.0379	0.0349
fbeta Score, weighted	0.0838	0.1373	0.1545	0.0982	0.0838
Jaccard Similarity	0.1176	0.1411	0.1647	0.1058	0.1176
Hamming Loss	0.8823	0.8588	0.8352	0.8941	0.8823
Zero One Loss	0.8823	0.8588	0.8352	0.8941	0.8823

Πίνακας 33: Σύγκριση classifiers, λεξικό Goodreads, default παράμετροι εισόδου, δεδομένα από Facebook

Τα σκορ που σημειώνουν οι classifiers είναι εξίσου χαμηλά με τα σκορ που σημείωσαν οι classifiers όταν δέχθηκαν ως είσοδο δεδομένα τα οποία είχαν αξιολογηθεί από το λεξικό

του Amazon/TripAdvisor. Ακόμη πρέπει να σημειώσω ότι ο Random Forest παρουσίασε μεγαλύτερη διακύμανση σε σχέση με το Decision Tree, ±5% έναντι ±2%.



Διάγραμμα 37: Σύγκριση classifiers, λεξικό AFINN, default παράμετροι εισόδου, δεδομένα από Facebook

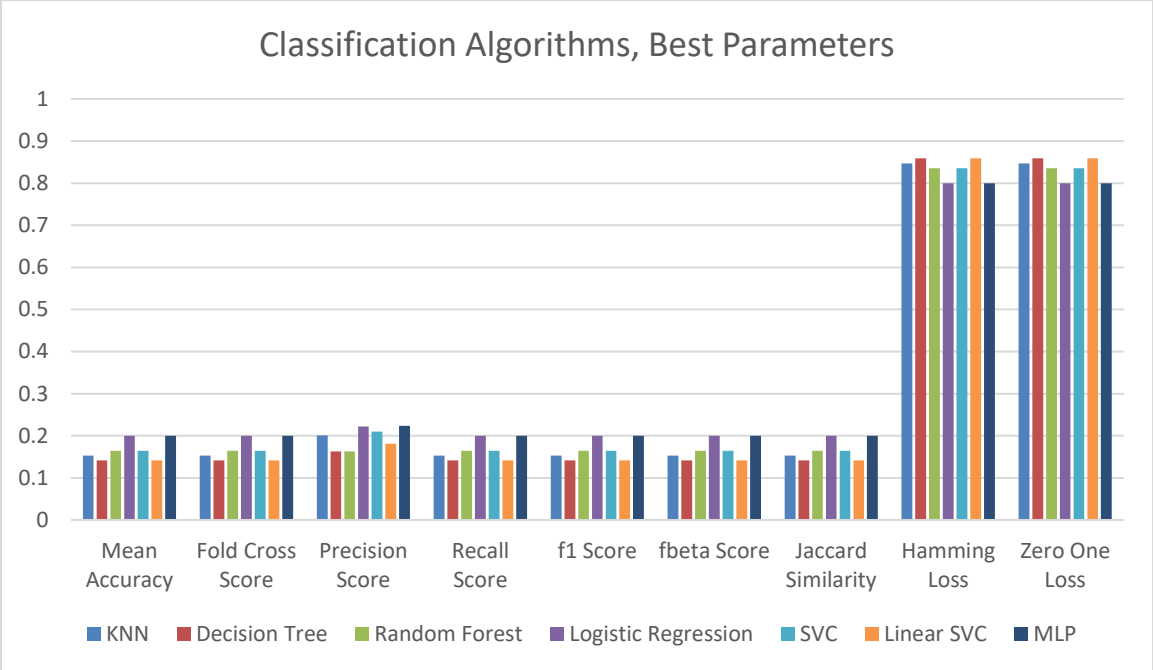
Από την οπτική απεικόνιση της απόδοσης των αλγορίθμων φαίνεται ότι ο Random Forest παρουσιάζει τα καλύτερα αποτελέσματα. Η αλήθεια είναι ότι η κατάταξη των αλγορίθμων δεν έχει μεγάλη σημασία καθώς το λεξικό του Goodreads φαίνεται ανεπαρκή, για αυτό προχωράω στα σκορ των classifiers όταν εκτελούνται με τις προτεινόμενες παραμέτρους από το Grid Search, που ίσως δείξουν κάποια βελτίωση.

	KNN	DT	RF	LR	SVC	L. SVC	MLP
Train Time(s)	0.0	0.0039	0.0049	0.084	0.0509	0.062	1.264
Score Time(s)	0.003	0.001	0.015	0.0	0.002	0.0	0.0017
Mean Accuracy	0.1529	0.1529	0.1882	0.2	0.1647	0.1411	0.2117
Fold Cross Score	0.2004	0.1627	0.1628	0.222	0.2096	0.1812	0.2237
Prec Score, micro	0.1529	0.1411	0.1647	0.2	0.1647	0.1411	0.2
Prec Score, macro	0.0857	0.0554	0.0641	0.0595	0.0554	0.0361	0.1078
Prec Score, weighted	0.1079	0.1383	0.1692	0.1384	0.1363	0.1004	0.162
Recall Score, micro	0.1529	0.1411	0.1647	0.2	0.1647	0.1411	0.2
Recall Score, macro	0.1067	0.0596	0.0668	0.091	0.0797	0.0558	0.1274
Recall Score, weighted	0.1529	0.1411	0.1647	0.2	0.1647	0.1411	0.2
f1 Score, micro	0.1529	0.1411	0.1647	0.2	0.1647	0.1411	0.2
f1 Score, macro	0.0897	0.0565	0.0574	0.0642	0.0465	0.0386	0.1117
f1 Score, weighted	0.1156	0.1372	0.1464	0.1443	0.1015	0.103	0.1673
fbeta Score, micro	0.1529	0.1411	0.1647	0.2	0.1647	0.1411	0.2
fbeta Score, macro	0.0867	0.0556	0.0594	0.0603	0.0463	0.0363	0.1085
fbeta Score, weighted	0.1095	0.1373	0.1545	0.1381	0.1076	0.0993	0.1619
Jaccard Similarity	0.1529	0.1411	0.1647	0.2	0.1647	0.1411	0.2

Hamming Loss	0.847	0.8588	0.8352	0.8	0.8352	0.8588	0.8
Zero One Loss	0.847	0.8588	0.8352	0.8	0.8352	0.8588	0.8

Πίνακας 34: Σύγκριση classifiers, λεξικό Goodreads, βέλτιστοι παράμετροι εισόδου, δεδομένα από Facebook

Οι classifiers παρουσιάζουν μια μικρή βελτίωση, πλην των Decision Tree και Random Forest στους οποίους δεν μπορεί να εντοπιστεί πιθανή αύξηση στα σκορ λόγω της διακύμανση που παρουσιάζουν. Η απόδοση των classifiers συνεχίζει να είναι πολύ χαμηλή και το λεξικό να μην θεωρείται κατάλληλο για τα δεδομένα που έχω συλλέξει από το Facebook.



Διάγραμμα 38: Σύγκριση classifiers, λεξικό Goodreads, βέλτιστοι παράμετροι εισόδου, δεδομένα από Facebook

Από όλους τους αλγορίθμους που έχω μελετήσει, έχοντας πραγματοποιήσει αναζήτηση βέλτιστων παραμέτρων και προεπεξεργασία δεδομένων εισόδου, ξεχωρίζει ελαφρώς ο Logistic Regression και ο MLP. Σε καμία περίπτωση όμως η απόδοση κανενός από τους δύο αλγορίθμους δεν χαρακτηρίζεται ως αρκετά καλή, ώστε το λεξικό του Goodreads να θεωρείται αποδεκτό.

Το λεξικό του Goodreads παρουσιάζει συμπεριφορά παρόμοια με αυτή του λεξικού Amazon/TripAdvisor.

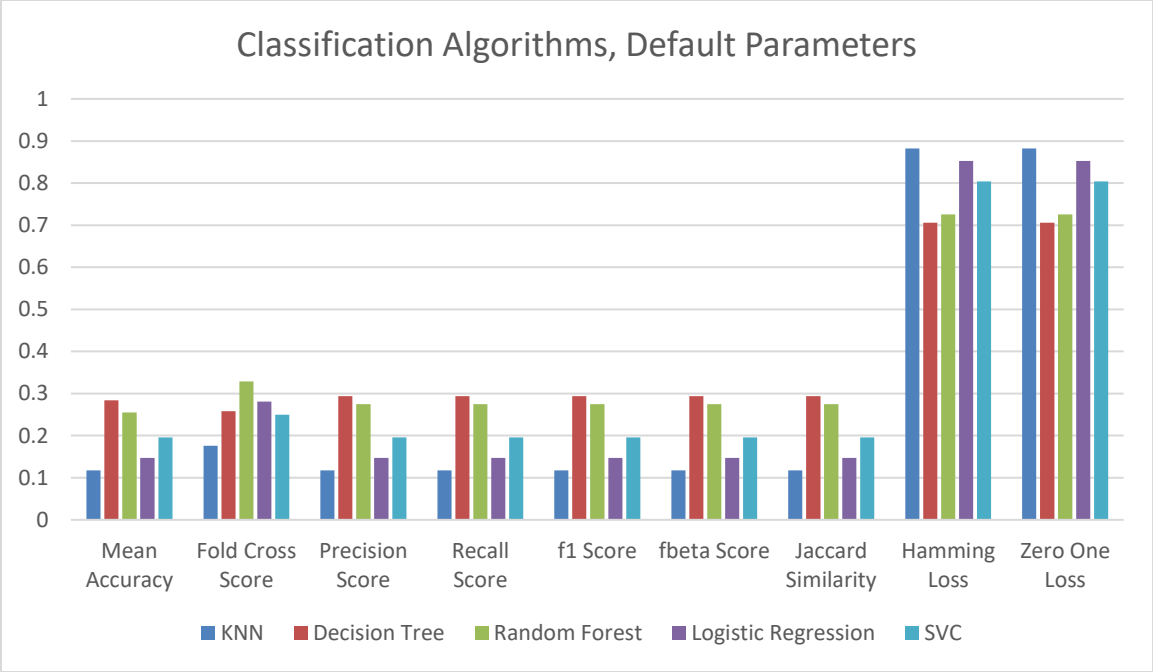
6.6.2 Αποτελέσματα στα δεδομένα που εξαχθεί από το Twitter

Συνεχίζοντας τη μελέτη του λεξικού Goodreads, εφαρμόζω τους classifiers στα δεδομένα που έχω συλλέξει από το Twitter και έχουν αξιολογηθεί από το συγκεκριμένο λεξικό.

	KNN	DT	RF	LR	SVC
Train Time(s)	0.0	0.003	0.0309	0.0409	0.021
Score Time(s)	0.0159	0.0	0.002	0.001	0.0029
Mean Accuracy	0.1372	0.3039	0.3333	0.2058	0.196
Fold Cross Score	0.1789	0.2732	0.3088	0.2915	0.2499
Prec Score, micro	0.1372	0.2941	0.3137	0.2058	0.196
Prec Score, macro	0.0565	0.1875	0.2113	0.1177	0.1934
Prec Score, weighted	0.1495	0.3213	0.3343	0.2321	0.382
Recall Score, micro	0.1372	0.2941	0.3137	0.2058	0.196
Recall Score, macro	0.0594	0.2272	0.2458	0.1099	0.1026
Recall Score, weighted	0.1372	0.2941	0.3137	0.2058	0.196
f1 Score, micro	0.1372	0.2941	0.3137	0.2058	0.196
f1 Score, macro	0.0467	0.1773	0.207	0.1041	0.0764
f1 Score, weighted	0.1169	0.2904	0.2968	0.1991	0.1568
fbeta Score, micro	0.1372	0.2941	0.3137	0.2058	0.196
fbeta Score, macro	0.0503	0.1797	0.2046	0.1091	0.1063
fbeta Score, weighted	0.1303	0.3042	0.312	0.2124	0.22
Jaccard Similarity	0.1372	0.2941	0.3137	0.2058	0.196
Hamming Loss	0.8627	0.7058	0.6862	0.7941	0.8039
Zero One Loss	0.8627	0.7058	0.6862	0.7941	0.8039

Πίνακας 35: Σύγκριση classifiers, λεξικό Goodreads, default παράμετροι εισόδου, δεδομένα από Twitter

Η βελτίωση των αλγορίθμων είναι εντυπωσιακή σε σχέση με τη συμπεριφορά τους όταν δέχονται δεδομένα του Facebook. Ο KNN παρουσιάζει τη μικρότερη βελτίωση κατά +3% και ο Decision Tree τη μεγαλύτερη κατά 15%. Πρέπει να σημειώσω ότι οι αλγόριθμοι του Decision Tree και Random Forest παρουσιάζουν όπως συνήθως μία διακύμανση, η οποία όμως είναι αποδεκτή, αφού είναι στο ±3%. Παρόλη την βελτίωση στα σκορ που παρουσιάζουν όλοι οι classifiers, το λεξικό του Goodreads κρίνεται ανεπαρκές.



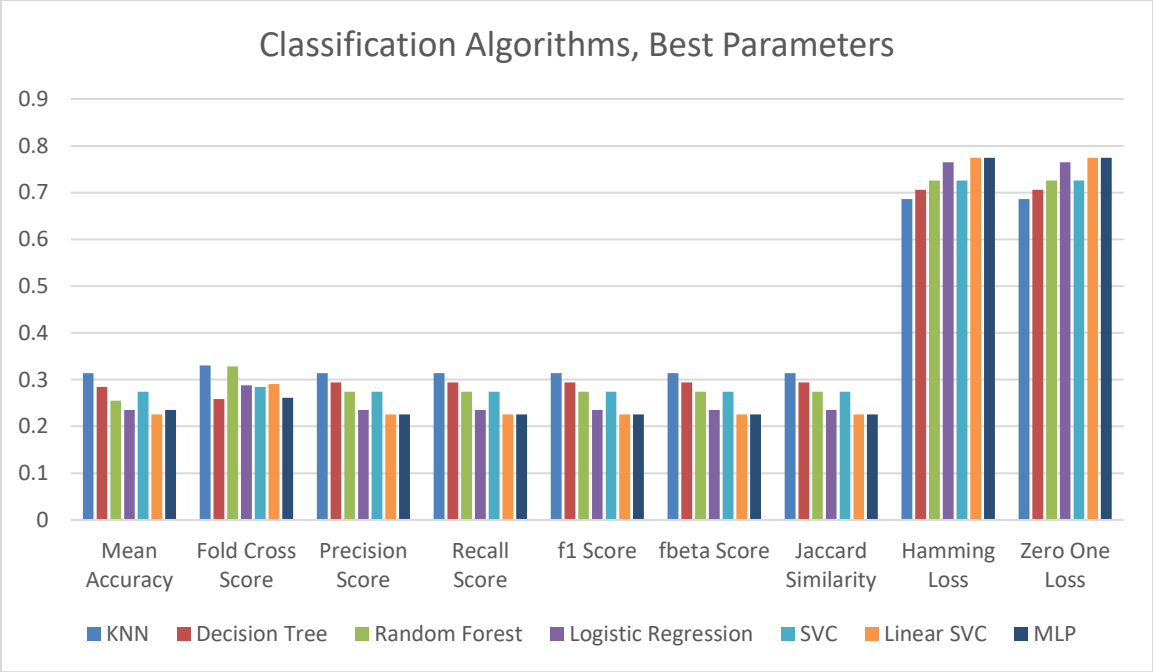
Διάγραμμα 39: Σύγκριση classifiers, λεξικό Goodreads, default παράμετροι εισόδου, δεδομένα από Twitter

Από την οπτική απεικόνιση των τεσσάρων αλγορίθμων ξεχωρίζουν οι Decision Tree και Random Forest. Βέβαια κανείς από τους αλγορίθμους δεν παρουσιάζει αρκετά υψηλό σκορ ώστε να θεωρηθεί το λεξικό του Goodreads αποδεκτό, για αυτό προχωράω στην εκτέλεση με βέλτιστες παραμέτρους εισόδου.

	KNN	DT	RF	LR	SVC	L. SVC	MLP
Train Time(s)	0.0	0.0039	0.0049	0.084	0.0509	0.062	1.264
Score Time(s)	0.003	0.001	0.015	0.0	0.002	0.0	0.0017
Mean Accuracy	0.1529	0.1529	0.1882	0.2	0.1647	0.1411	0.2117
Fold Cross Score	0.2004	0.1627	0.1628	0.222	0.2096	0.1812	0.2237
Prec Score, micro	0.1529	0.1411	0.1647	0.2	0.1647	0.1411	0.2
Prec Score, macro	0.0857	0.0554	0.0641	0.0595	0.0554	0.0361	0.1078
Prec Score, weighted	0.1079	0.1383	0.1692	0.1384	0.1363	0.1004	0.162
Recall Score, micro	0.1529	0.1411	0.1647	0.2	0.1647	0.1411	0.2
Recall Score, macro	0.1067	0.0596	0.0668	0.091	0.0797	0.0558	0.1274
Recall Score, weighted	0.1529	0.1411	0.1647	0.2	0.1647	0.1411	0.2
f1 Score, micro	0.1529	0.1411	0.1647	0.2	0.1647	0.1411	0.2
f1 Score, macro	0.0897	0.0565	0.0574	0.0642	0.0465	0.0386	0.1117
f1 Score, weighted	0.1156	0.1372	0.1464	0.1443	0.1015	0.103	0.1673
fbeta Score, micro	0.1529	0.1411	0.1647	0.2	0.1647	0.1411	0.2
fbeta Score, macro	0.0867	0.0556	0.0594	0.0603	0.0463	0.0363	0.1085
fbeta Score, weighted	0.1095	0.1373	0.1545	0.1381	0.1076	0.0993	0.1619
Jaccard Similarity	0.1529	0.1411	0.1647	0.2	0.1647	0.1411	0.2
Hamming Loss	0.847	0.8588	0.8352	0.8	0.8352	0.8588	0.8

Πίνακας 36: Σύγκριση classifiers, λεξικό Goodreads, βέλτιστοι παράμετροι εισόδου, δεδομένα από Twitter

Συνολικά από τους αλγορίθμους, μπορεί να θεωρηθεί ότι ο KNN παρουσιάζει τα καλύτερα αποτελέσματα σε σχέση με τους υπόλοιπους classifiers, αλλά σε καμία περίπτωση το λεξικό του Goodreads δεν είναι επαρκή για τα δεδομένα που μελετώ.



Διάγραμμα 40: Σύγκριση classifiers, λεξικό Goodreads, βέλτιστοι παράμετροι εισόδου

Το λεξικό του Goodreads δεν προτείνεται για τη φύση των δεδομένων που έχω συλλέξει, όπως και το λεξικό του Amazon/TripAdvisor, το οποίο μελέτησα νωρίτερα. Παρόλο που προσπάθησα να αυξήσω την απόδοση των classifiers, καμία μετρική δεν ξεπέρασε το 0.4.

Συνεχίζω με την μελέτη του OpenTable, αλλά πιστεύω ότι ούτε το λεξικό του OpenTable θα μπορέσει να παρουσιάσει αποδεκτά σκορ στις μετρικές που χρησιμοποιώ, εξαιτίας της κλίμακας πάνω στην οποία βαθμολογούνται οι εγγραφές τους σε συνδυασμό με τους μέσους όρους που εμφανίζουν.

6.7 Αποτελέσματα του λεξικού Opentable

Το τελευταίο λεξικό της οικογένειας των λεξικών που προέρχονται από ανάγνωση δεδομένων σε site που μελετάω είναι το λεξικό του Opentable. Η απόδοση που περιμένω να έχει το συγκεκριμένο λεξικό είναι παρόμοια με την απόδοση των λεξικών Amazon/TripAdvisor και Goodreads.

6.7.1 Αποτελέσματα στα δεδομένα που εξαχθεί από το Facebook

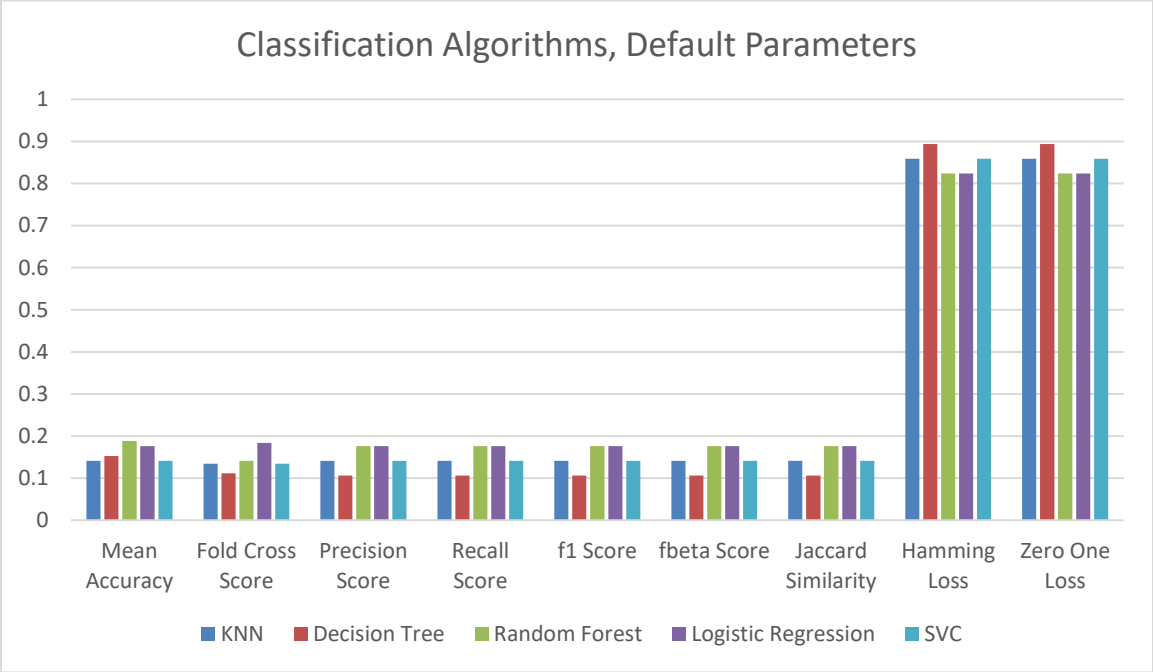
Τα αποτελέσματα του λεξικού όταν αξιολογεί δεδομένα τα οποία προέρχονται από το Facebook και οι classifiers εκτελούνται με τις default παραμέτρους εισόδου φαίνονται στον πίνακα που ακολουθεί.

	KNN	DT	RF	LR	SVC
Train Time(s)	0.0	0.0	0.016000	0.147	0.0
Score Time(s)	0.0	0.0	0.0	0.0	0.0
Mean Accuracy	0.1411	0.1529	0.1882	0.1764	0.1411
Fold Cross Score	0.1345	0.1113	0.1414	0.1835	0.1345
Prec Score, micro	0.1411	0.1058	0.1764	0.1764	0.1411
Prec Score, macro	0.0367	0.0601	0.0853	0.0858	0.0367
Prec Score, weighted	0.0797	0.1016	0.1497	0.1876	0.0797
Recall Score, micro	0.1411	0.1058	0.1764	0.1764	0.1411
Recall Score, macro	0.0688	0.0825	0.0973	0.0812	0.0688
Recall Score, weighted	0.1411	0.1058	0.1764	0.1764	0.1411
f1 Score, micro	0.1411	0.1058	0.1764	0.1764	0.1411
f1 Score, macro	0.0424	0.0673	0.0805	0.0785	0.0424
f1 Score, weighted	0.0907	0.1024	0.142	0.1714	0.0907
fbeta Score, micro	0.1411	0.1058	0.1764	0.1764	0.1411
fbeta Score, macro	0.0384	0.0624	0.082	0.0813	0.0384
fbeta Score, weighted	0.0829	0.1017	0.144	0.1778	0.0829
Jaccard Similarity	0.1411	0.1058	0.1764	0.1764	0.1411
Hamming Loss	0.8588	0.8941	0.8235	0.8235	0.8588
Zero One Loss	0.8588	0.8941	0.8235	0.8235	0.8588

Πίνακας 37: Σύγκριση classifiers, λεξικό Opentable, default παράμετροι εισόδου, δεδομένα από Facebook

Οι classifiers αποδίδουν χαμηλά σκορ, όπως άλλωστε περίμενα, χωρίς να ξεχωρίζει από πλευράς απόδοσης κάποιος αλγόριθμος. Ακολουθεί η οπτική απεικόνιση της

απόδοσης των classifiers όταν εκτελούνται με τις default παραμέτρους εισόδου και αξιολογούν δεδομένα που προέρχονται από το Facebook.



Διάγραμμα 41: Σύγκριση classifiers, λεξικό Opentable, default παράμετροι εισόδου, δεδομένα από Facebook

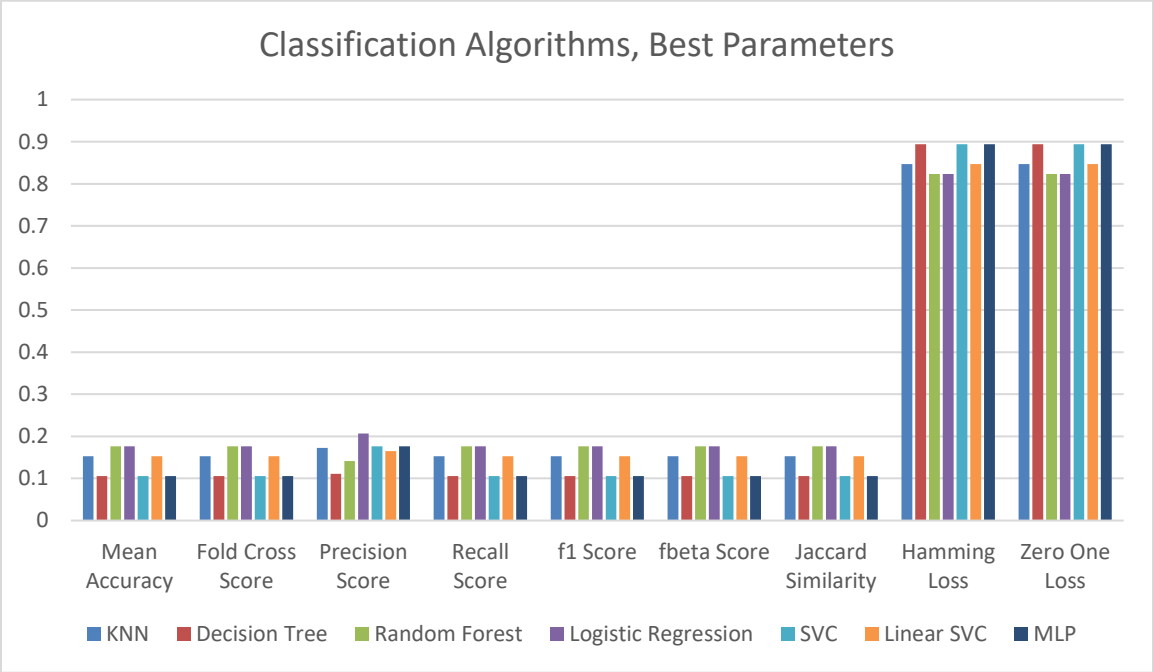
Από την παρατήρηση του διαγράμματος ξεχωρίζει ελαφρώς ο Logistic Regression, ο οποίος παρουσιάζει ελαφρώς καλύτερο σκορ στην μετρική του fold cross score. Το λεξικό του Opentable δεν έχει αποδεκτή απόδοση σε καμία μετρική. Η συμπεριφορά των classifiers όταν χρησιμοποιούνται με τις βέλτιστες παραμέτρους φαίνεται στον πίνακα που ακολουθεί.

	KNN	DT	RF	LR	SVC	L. SVC	MLP
Train Time(s)	0.0	0.0	0.016	0.013	0.0889	0.042	0.2956
Score Time(s)	0.015	0.0	0.0	0.0	0.003	0.0	0.0019
Mean Accuracy	0.1529	0.1529	0.1882	0.1764	0.1058	0.1529	0.1058
Fold Cross Score	0.1727	0.1113	0.1414	0.2067	0.1766	0.1652	0.1766
Prec Score, micro	0.1529	0.1058	0.1764	0.1764	0.1058	0.1529	0.1058
Prec Score, macro	0.0813	0.0601	0.0853	0.0714	0.0058	0.0704	0.0058
Prec Score, weighted	0.1008	0.1016	0.1497	0.1477	0.0112	0.1564	0.0112
Recall Score, micro	0.1529	0.1058	0.1764	0.1764	0.1058	0.1529	0.1058
Recall Score, macro	0.0856	0.0825	0.0973	0.0868	0.0555	0.0664	0.0555
Recall Score, weighted	0.1529	0.1058	0.1764	0.1764	0.1058	0.1529	0.1058
f1 Score, micro	0.1529	0.1058	0.1764	0.1764	0.1058	0.1529	0.1058
f1 Score, macro	0.0542	0.0673	0.0805	0.0593	0.0106	0.0561	0.0106
f1 Score, weighted	0.0844	0.1024	0.142	0.1219	0.0202	0.1275	0.0202
fbeta Score, micro	0.1529	0.1058	0.1764	0.1764	0.1058	0.1529	0.1058
fbeta Score, macro	0.0621	0.0624	0.082	0.0614	0.0071	0.0615	0.0071
fbeta Score, weighted	0.0864	0.1017	0.144	0.1271	0.0136	0.1382	0.0136
Jaccard Similarity	0.1529	0.1058	0.1764	0.1764	0.1058	0.1529	0.1058

Hamming Loss	0.8470	0.8941	0.8235	0.8235	0.8941	0.847	0.8941
Zero One Loss	0.8470	0.8941	0.8235	0.8235	0.8941	0.847	0.8941

Πίνακας 38: Σύγκριση classifiers, λεξικό Opentable, βέλτιστοι παράμετροι εισόδου, δεδομένα από Facebook

Οι classifiers παρουσιάζουν μια μικρή βελτίωση όταν δέχονται δεδομένα τα οποία έχουν υποστεί προεπεξεργασία και αυτοί εκτελούνται με τις βέλτιστες παραμέτρους, αλλά σε καμία περίπτωση η απόδοση του λεξικού δεν θεωρείται αποδεκτή.



Διάγραμμα 42: Σύγκριση classifiers, λεξικό Opentable, βέλτιστοι παράμετροι εισόδου, δεδομένα από Facebook

Από το διάγραμμα απόδοσης των classifiers φαίνεται ξεκάθαρα η χαμηλή απόδοση των classifies στο λεξικό του Opentable. Το λεξικό του OpenTable δεν προτείνεται να εφαρμοστεί στα δεδομένα που έχω συλλέξει. Στη συνέχεια μελετάω τη συμπεριφορά του λεξικού όταν δέχεται δεδομένα τα οποία έχουν εξαχθεί από το Twitter, αλλά ούτε σε αυτά τα δεδομένα είναι πιθανό να παρουσιάσει αποδεκτή απόδοση.

6.7.2 Αποτελέσματα στα δεδομένα που εξαχθεί από το Twitter

Έχοντας ολοκληρώσει την μελέτη των classifiers όταν τα δεδομένα του Facebook αξιολογούνται από το λεξικό του Opentable, προχωράω στη μελέτη της συμπεριφοράς των

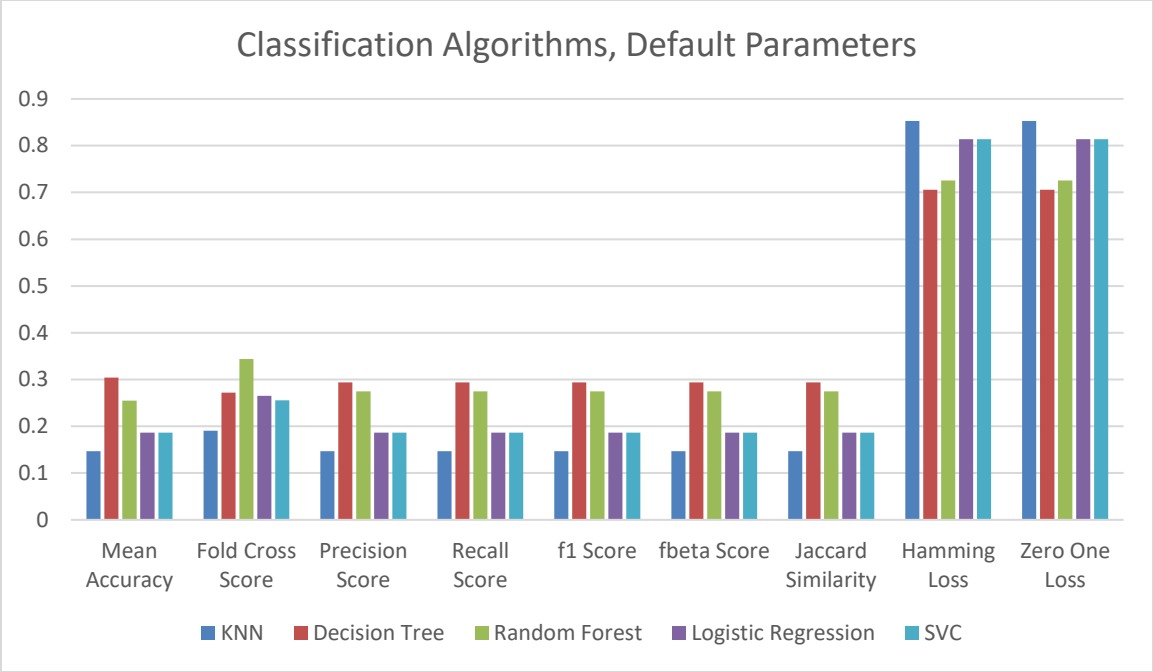
classifiers όταν δέχονται δεδομένα τα οποία από το Twitter. Τα σκορ που επιτυγχάνουν οι classifiers όταν εκτελούνται με τις default μετρικές φαίνονται στον πίνακα που ακολουθεί.

	KNN	DT	RF	LR	SVC
Train Time(s)	0.0	0.0	0.016	0.0309	0.0179
Score Time(s)	0.0	0.0	0.016	0.0	0.0
Mean Accuracy	0.147	0.3039	0.2549	0.1862	0.1862
Fold Cross Score	0.1907	0.2721	0.3437	0.2649	0.2554
Prec Score, micro	0.147	0.2941	0.2745	0.1862	0.1862
Prec Score, macro	0.1177	0.2363	0.2187	0.1445	0.1327
Prec Score, weighted	0.1673	0.3126	0.3099	0.2132	0.2392
Recall Score, micro	0.1470	0.2941	0.2745	0.1862	0.1862
Recall Score, macro	0.0851	0.2077	0.1612	0.1319	0.095
Recall Score, weighted	0.147	0.2941	0.2745	0.1862	0.1862
f1 Score, micro	0.147	0.2941	0.2745	0.1862	0.1862
f1 Score, macro	0.0803	0.2037	0.1642	0.13	0.0631
f1 Score, weighted	0.1276	0.289	0.2604	0.1818	0.1154
fbeta Score, micro	0.1470	0.2941	0.2745	0.1862	0.1862
fbeta Score, macro	0.0941	0.2166	0.1805	0.1356	0.0823
fbeta Score, weighted	0.1413	0.2973	0.2716	0.1937	0.1473
Jaccard Similarity	0.147	0.2941	0.2745	0.1862	0.1862
Hamming Loss	0.8529	0.7058	0.7254	0.8137	0.8137
Zero One Loss	0.8529	0.7058	0.7254	0.8137	0.8137

Πίνακας 39: Σύγκριση classifiers, λεξικό Opentable, default παράμετροι εισόδου, δεδομένα από Twitter

Οι classifiers παρουσιάζουν καλύτερα σκορ όταν λαμβάνουν είσοδο δεδομένα τα οποία έχουν εξαχθεί από το Twitter. Ο classifier που παρουσιάζει τα καλύτερα αποτελέσματα είναι ο Random Forest, αλλά ούτε κι αυτός έχει αποτελέσματα τα οποία είναι ανεκτά σε απόδοση.

Στη συνέχεια παρουσιάζω το διάγραμμα με τις αποδόσεις των classifiers, στο οποίο ξεχωρίζουν οι classifiers των Decision Tree και Random Forest, ενώ ο KNN παρουσιάζει τα χειρότερα αποτελέσματα.



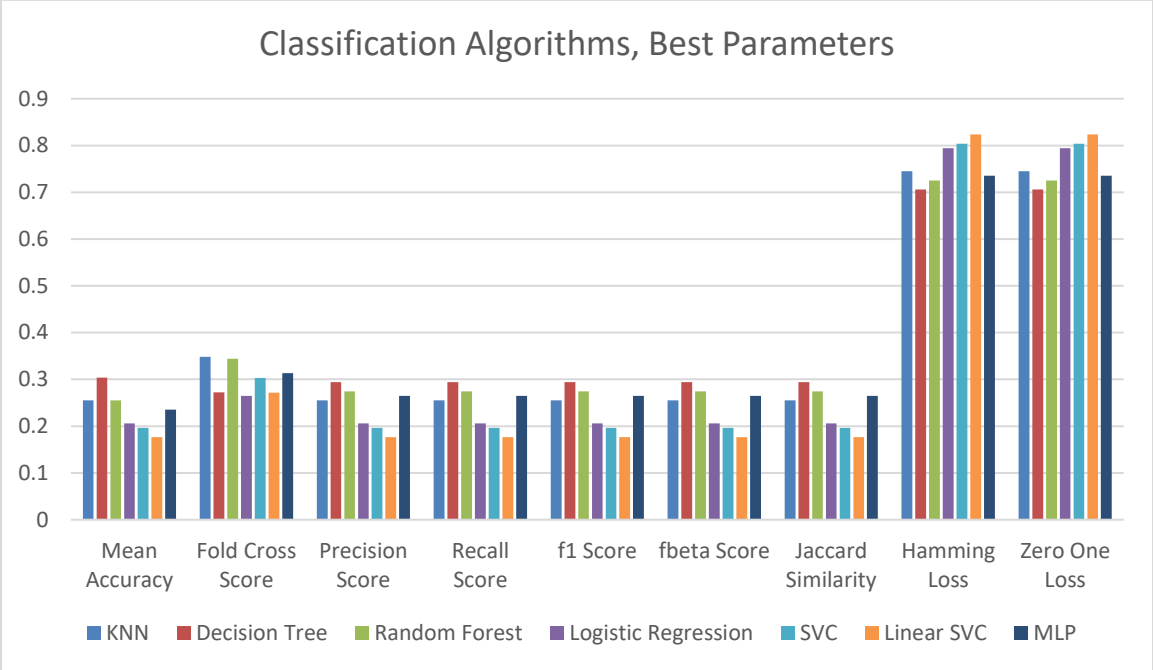
Διάγραμμα 43: Σύγκριση classifiers, λεξικό Opentable, default παράμετροι εισόδου, δεδομένα από Twitter

Κανένας από τους classifiers που χρησιμοποιώ δεν παρουσιάζει έστω ανεκτά απόδοση, άνω του 0.4, οπότε προχωράω στην μελέτη των classifiers όταν αυτοί εκτελούνται με τις βέλτιστες παραμέτρους και τα δεδομένα εισόδου δέχονται προεπεξεργασία.

	KNN	DT	RF	LR	SVC	L. SVC	MLP
Train Time(s)	0.015	0.0	0.016	0.1159	0.0789	0.1	0.8848
Score Time(s)	0.0	0.0	0.016	0.0	0.003	0.0	0.0044
Mean Accuracy	0.2549	0.3039	0.2549	0.2058	0.196	0.1764	0.2352
Fold Cross Score	0.3478	0.2721	0.3437	0.2648	0.3027	0.2717	0.3131
Prec Score, micro	0.2549	0.2941	0.2745	0.2058	0.196	0.1764	0.2647
Prec Score, macro	0.2397	0.2363	0.2187	0.1761	0.1345	0.1429	0.1581
Prec Score, weighted	0.3164	0.3126	0.3099	0.2452	0.2392	0.2084	0.2599
Recall Score, micro	0.2549	0.2941	0.2745	0.2058	0.196	0.1764	0.2647
Recall Score, macro	0.1707	0.2077	0.1612	0.144	0.1138	0.1084	0.1640
Recall Score, weighted	0.2549	0.2941	0.2745	0.2058	0.196	0.1764	0.2647
f1 Score, micro	0.2549	0.2941	0.2745	0.2058	0.196	0.1764	0.2647
f1 Score, macro	0.1735	0.2037	0.1642	0.1449	0.0989	0.0953	0.1555
f1 Score, weighted	0.2529	0.289	0.2604	0.1990	0.1663	0.1485	0.2558
fbeta Score, micro	0.2549	0.2941	0.2745	0.2058	0.196	0.1764	0.2647
fbeta Score, macro	0.1944	0.2166	0.1805	0.1575	0.1084	0.1082	0.1557
fbeta Score, weighted	0.2732	0.2973	0.2716	0.2169	0.1859	0.1656	0.2565
Jaccard Similarity	0.2549	0.2941	0.2745	0.2058	0.196	0.1764	0.2647
Hamming Loss	0.7450	0.7058	0.7254	0.7941	0.8039	0.8235	0.7352
Zero One Loss	0.7450	0.7058	0.7254	0.7941	0.8039	0.8235	0.7352

Πίνακας 40: Σύγκριση classifiers, λεξικό Opentable, βέλτιστοι παράμετροι εισόδου, δεδομένα από Twitter

Ο classifier του KNN παρουσιάζει την μεγαλύτερη βελτίωση και το υψηλότερο σκορ, αν ληφθεί υπόψιν η διακύμανση των υπόλοιπων classifiers. Συνεχίζουν όμως οι classifiers να μην παρουσιάζουν ικανοποιητικά σκορ οπότε απορρίπτω το λεξικό του Opentable για την αξιολόγηση των δεδομένων που έχω συλλέξει.



Διάγραμμα 44: Σύγκριση classifiers, λεξικό Opentable, βέλτιστοι παράμετροι εισόδου

Από την οπτική απεικόνιση των classifiers δεν ξεχωρίζει κάποιος αλγόριθμος από πλευράς απόδοσης, πόσο μάλλον δε αν υπολογίσουμε τη διακύμανση των Decision Tree και Random Forest, οι οποίοι ξεχωρίζουν ελαφρώς στις όλες τις μετρικές πλην του fold cross score.

Κανένα λεξικό από τα Amazon/TripAdvisor, Goodreads και Opentable δεν μπορεί να χρησιμοποιηθεί για να αξιολογήσει τα δεδομένα που έχω συλλέξει. Ο λόγος της χαμηλής απόδοσης των λεξικών είναι εν μέρει δική μου ευθύνη επειδή όπως αποδείχθηκε το εύρος τιμών ήταν μικρό και υπήρξε μεγάλη συσσώρευση τιμών στο εύρος τιμών [0,5], Έτσι οι classifiers αδυνατούσαν να πραγματοποιήσουν σωστή πρόβλεψη των επόμενων προτάσεων στη βάση δεδομένων που έχω συλλέξει.

6.8 Αποτελέσματα του λεξικού Opinion Observer

Έχοντας ολοκληρώσει την μελέτη των λεξικών των οποίων η αρχική ιδέα για τη δημιουργία τους ήταν η εξαγωγή δεδομένων μέσα από ιστοσελίδες με κριτικές, προχωράω στη μελέτη του λεξικό Opinion Observer. Το συγκεκριμένο λεξικό έχει δημιουργηθεί και αυτό μέσα από ανάγνωση κριτικών, αλλά σε αντίθεση με τα προηγούμενα λεξικά προσφέρει στις λέξεις μονάχα θετική ή αρνητική χροιά χωρίς βαρύτητα.

6.8.1 Αποτελέσματα στα δεδομένα που έχουν εξαχθεί από το Facebook

Τα αποτελέσματα των classifiers όταν δέχονται δεδομένα τα οποία έχουν εξαχθεί από το Facebook κι έχουν αξιολογηθεί από το λεξικό του Opinion Observer φαίνονται στον πίνακα παρακάτω.

	KNN	DT	RF	LR	SVC
Train Time(s)	0.0	0.0	0.0329	0.031	0.0219
Score Time(s)	0.0	0.0	0.0009	0.0	0.0
Mean Accuracy	0.3411	0.4	0.447	0.4705	0.5058
Fold Cross Score	0.4014	0.4199	0.4642	0.4732	0.4566
Prec Score, micro	0.3411	0.4352	0.4	0.4705	0.5058
Prec Score, macro	0.1353	0.2322	0.2224	0.1964	0.1023
Prec Score, weighted	0.2954	0.4675	0.3945	0.4321	0.2649
Recall Score, micro	0.3411	0.4352	0.4	0.4705	0.5058
Recall Score, macro	0.1484	0.2311	0.1957	0.1991	0.1954
Recall Score, weighted	0.3411	0.4352	0.4	0.4705	0.5058
f1 Score, micro	0.3411	0.4352	0.4	0.4705	0.5058
f1 Score, macro	0.14	0.230	0.1928	0.1962	0.1343
f1 Score, weighted	0.3142	0.449	0.3847	0.448	0.3477
fbeta Score, micro	0.3411	0.4352	0.4	0.4705	0.5058
fbeta Score, macro	0.1368	0.2309	0.2013	0.1959	0.1131
fbeta Score, weighted	0.302	0.4595	0.3838	0.4377	0.2928
Jaccard Similarity	0.3411	0.4352	0.4	0.4705	0.5058
Hamming Loss	0.6588	0.5647	0.5999	0.5294	0.4941
Zero One Loss	0.6588	0.5647	0.5999	0.5294	0.4941

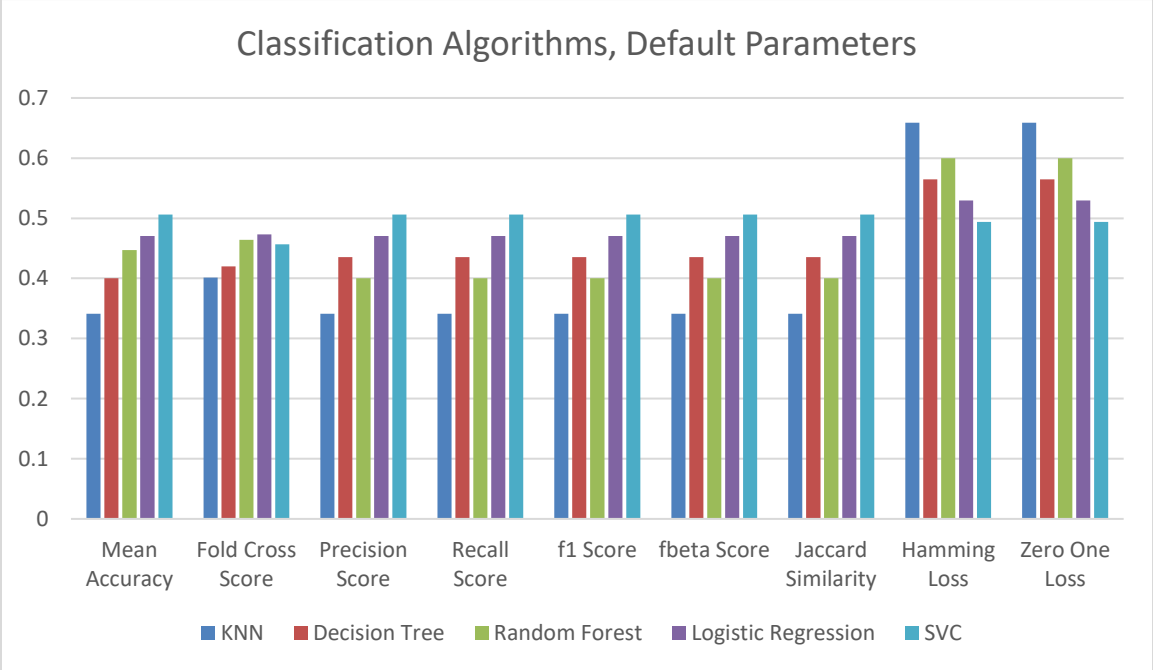
Πίνακας 41: Σύγκριση classifiers, λεξικό Opinion Observer, default παράμετροι εισόδου, δεδομένα από Facebook

Ο παραπάνω πίνακας έχει εξαχθεί θέτοντας τιμή στις εγγραφές ίση με ±1, ανάλογα με την κατηγορία στην οποία ανήκουν. Τα αποτελέσματα δεν είναι αρκετά καλά ώστε να θεωρήσουμε το λεξικό κατάλληλο για την αξιολόγηση των δεδομένων που έχω συλλέξει.

Στην προσπάθεια μου να βελτιώσω τα αποτελέσματα που έχουν εξαχθεί με τη χρήση του λεξικού που διαθέτει μόνο θετική και αρνητική χροιά, έδωσα μεγαλύτερο σκορ στις εγγραφές τόσο στις θετικές όσο και στις αρνητικές. Έδινα συνεχώς αυξανόμενες τιμές, προκειμένου να μελετήσω τη συμπεριφορά των classifiers. Ξεκίνησα σταδιακά από το ±1 και κατέληξα μέχρι το ±8. Υπήρξαν κάποιες μικρές αλλαγές στα σκορ των διαφορετικών μετρικών, αλλά δεν υπήρξε κάποιο μοτίβο, στο οποίο να υποδηλώνεται ότι όσο αυξάνεται η τιμή που θέτω στις εγγραφές τα αποτελέσματα των classifiers είναι καλύτερα.

Σχετικά με την απόδοση των classifiers όταν το λεξικό βαθμολογεί τις λέξεις με ±8 απλά αναφέρω τις τιμές του fold cross score στους διαφορετικούς classifiers. Ο KNN είχε σκορ ίσο με 0.3411, ο Decision Tree 0.447, ο Random Forest 0.4503, ο Logistic Regression 0.4732 και ο SVC 0.4564. Είτε δεν υπάρχουν διαφοροποιήσεις στο σκορ είτε δεν είναι σημαντικές.

Παρακάτω παραθέτω και το διάγραμμα με την οπτική απεικόνιση της απόδοσης των classifiers με τη χρήση του συγκεκριμένου λεξικού.



Διάγραμμα 45: Σύγκριση classifiers, λεξικό Opinion Observer, default παράμετροι εισόδου, δεδομένα από Facebook

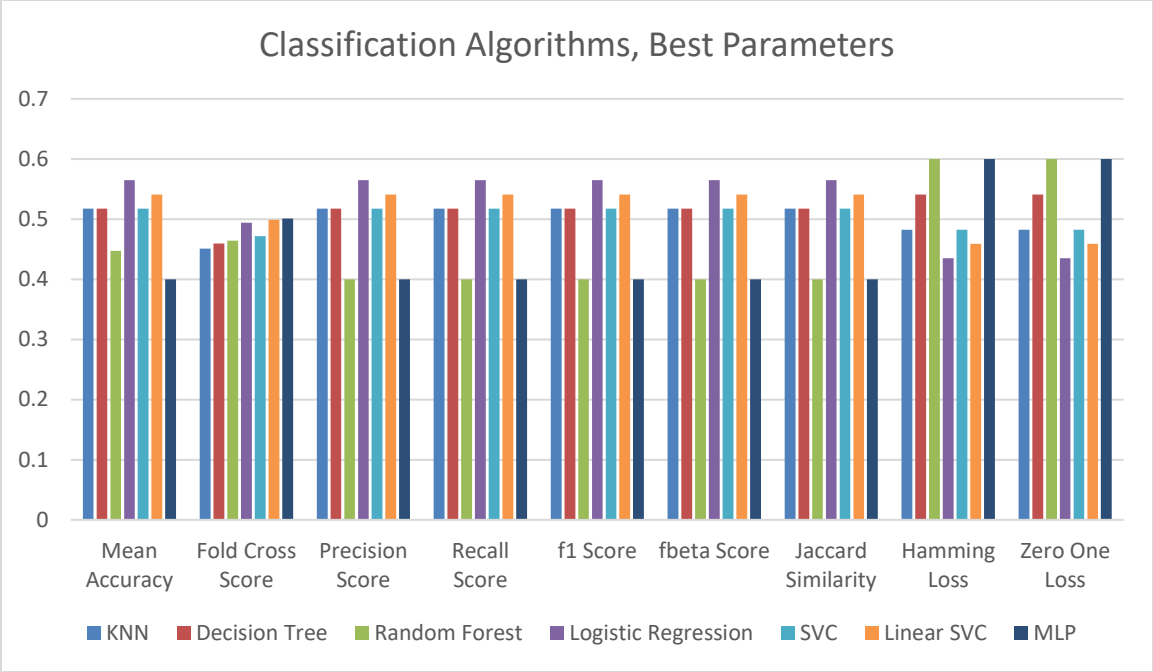
Από την παρατήρηση του παραπάνω διαγράμματος, ξεχωρίζουν οι αλγόριθμοι των Random Forest, Logistic Regression και SVC. Οι classifiers των Random Forest και SVC είναι οι δύο classifiers οι οποίοι παρουσιάζουν από τα καλύτερα αποτελέσματα σε όλα τα λεξικά όταν εκτελούνται οι classifiers με τις default παραμέτρους και στη συνέχεια δέχονται ελάχιστη βελτίωση. Αντίθετα ο classifier του KNN παρουσιάζει άσχημα αποτελέσματα με τη χρήση των default παραμέτρων, αλλά διαθέτει μεγάλο περιθώριο βελτίωσης.

Στον πίνακα που ακολουθεί παραθέτω τα αποτελέσματα των classifiers όταν εκτελούνται με τις προτεινόμενες παραμέτρους από το Grid Search και εφόσον τα δεδομένα εισόδου έχουν δεχθεί προεπεξεργασία.

	KNN	DT	RF	LR	SVC	L. SVC	MLP
Train Time(s)	0.001	0.0	0.0329	0.0309	0.0689	0.0	2.0499
Score Time(s)	0.0009	0.0	0.0009	0.0	0.0	0.0	0.002
Mean Accuracy	0.5176	0.5176	0.447	0.5647	0.5176	0.5411	0.447
Fold Cross Score	0.451	0.4594	0.4642	0.4947	0.4717	0.4987	0.501
Prec Score, micro	0.5176	0.4588	0.4	0.5647	0.5176	0.5411	0.4
Prec Score, macro	0.283	0.1602	0.2224	0.3955	0.1035	0.1968	0.2367
Prec Score, weighted	0.4584	0.3442	0.3945	0.5547	0.2679	0.4153	0.3769
Recall Score, micro	0.5176	0.4588	0.4	0.5647	0.5176	0.5411	0.4
Recall Score, macro	0.2813	0.2021	0.1957	0.2552	0.2	0.2422	0.1874
Recall Score, weighted	0.5176	0.4588	0.4	0.5647	0.5176	0.5411	0.4
f1 Score, micro	0.5176	0.4588	0.4	0.5647	0.5176	0.5411	0.4
f1 Score, macro	0.2775	0.1767	0.1928	0.2387	0.1364	0.2147	0.1802
f1 Score, weighted	0.4801	0.3901	0.3847	0.4839	0.3531	0.466	0.3624
fbeta Score, micro	0.5176	0.4588	0.4	0.5647	0.5176	0.5411	0.4
fbeta Score, macro	0.2796	0.166	0.2013	0.2612	0.1145	0.2031	0.1923
fbeta Score, weighted	0.4653	0.3605	0.3838	0.4732	0.2965	0.4334	0.3549
Jaccard Similarity	0.5176	0.4588	0.4	0.5647	0.5176	0.5411	0.4
Hamming Loss	0.4823	0.5411	0.5999	0.4352	0.4823	0.4588	0.5999
Zero One Loss	0.4823	0.5411	0.5999	0.4352	0.4823	0.4588	0.5999

Πίνακας 42: Σύγκριση classifiers, λεξικό Opinion Observer, βέλτιστοι παράμετροι εισόδου, δεδομένα από Facebook

Η απόδοση όλων των classifiers είναι αρκετά κοντινή, ειδικά αν υπολογίσουμε και τη διακύμανση των Decision Tree και Random Forest. Ακόμη η βελτίωση που παρουσιάζει ο KNN είναι θεαματική.



Διάγραμμα 46: Σύγκριση classifiers, λεξικό Opinion Observer, βέλτιστοι παράμετροι εισόδου, δεδομένα από Facebook

Το λεξικό του Opinion Observer παρουσιάζει παρόμοια απόδοση με αυτό του AFINN. Η απόδοση τους δεν είναι εντυπωσιακή, αλλά ούτε κι απογοητευτική και με κάποιες βελτιώσεις και προσθέσεις, όπως πρόσθεση βαρύτητας για κάθε λέξη, το λεξικό μπορεί να γίνει κατάλληλο για την αξιολόγηση των δεδομένων που έχω συλλέξει.

6.8.2 Αποτελέσματα στα δεδομένα που εξαχθεί από το Twitter

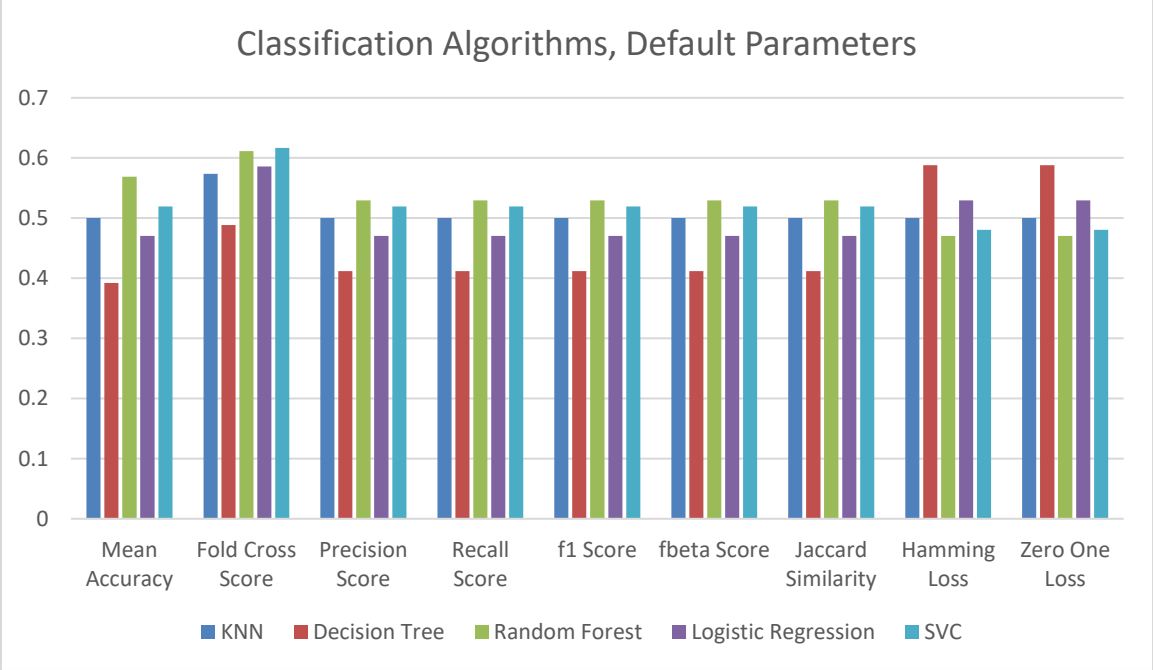
Το λεξικό του Opinion Observer περιμένω να παρουσιάσει ελαφρώς καλύτερα αποτελέσματα όταν καλείται να επεξεργαστεί δεδομένα τα οποία έχουν εξαχθεί από το Twitter. Τα αποτελέσματα των classifiers τα παραθέτω στον πίνακα που ακολουθεί.

	KNN	**DT**	**RF**	**LR**	**SVC**
Train Time(s)	0.0	0.0	0.0149	0.0	0.015000
Score Time(s)	0.0	0.0	0.0	0.0	0.0
Mean Accuracy	0.5	0.3921	0.5686	0.4705	0.5196
Fold Cross Score	0.5737	0.4887	0.6116	0.5856	0.6169
Prec Score, micro	0.5	0.4117	0.5294	0.4705	0.5196
Prec Score, macro	0.1971	0.2142	0.3343	0.1428	0.1430
Prec Score, weighted	0.3819	0.4060	0.4974	0.38	0.3761

Recall Score, micro	0.5	0.4117	0.5294	0.4705	0.5196
Recall Score, macro	0.196	0.2482	0.2707	0.1604	0.1660
Recall Score, weighted	0.5	0.4117	0.5294	0.4705	0.5196
f1 Score, micro	0.5	0.4117	0.5294	0.4705	0.5196
f1 Score, macro	0.1789	0.2233	0.2646	0.144	0.1233
f1 Score, weighted	0.4069	0.4024	0.4603	0.4062	0.3776
fbeta Score, micro	0.5	0.4117	0.5294	0.4705	0.5196
fbeta Score, macro	0.1832	0.2161	0.2886	0.1411	0.1161
fbeta Score, weighted	0.381	0.4027	0.4601	0.3858	0.3408
Jaccard Similarity	0.5	0.4117	0.5294	0.4705	0.5196
Hamming Loss	0.5	0.5882	0.4705	0.5294	0.4803
Zero One Loss	0.5	0.5882	0.4705	0.5294	0.4803

Πίνακας 43: Σύγκριση classifiers, λεξικό Opinion Observer, default παράμετροι εισόδου, δεδομένα από Twitter

Οι classifiers με τη χρήση του λεξικού Opinion Observer όταν δέχονται ως είσοδο τα δεδομένα που έχουν εξαχθεί από το Twitter, παρουσιάζουν πολύ καλύτερα αποτελέσματα συγκριτικά με την απόδοση τους έχοντας ως δεδομένα εισόδου δεδομένα από το Facebook. Η απόδοση των αλγορίθμων με τη χρήση του λεξικού Opinion Observer είναι παρόμοια με αυτή που παρουσιάζουν όταν χρησιμοποιείται το λεξικό AFINN, τόσο στα δεδομένα του Facebook, όσο και στα δεδομένα του Twitter. Το διάγραμμα με την οπτική απεικόνιση της απόδοσης των classifiers φαίνεται παρακάτω.



Διάγραμμα 47: Σύγκριση classifiers, λεξικό Opinion Observer, default παράμετροι εισόδου, δεδομένα από Twitter

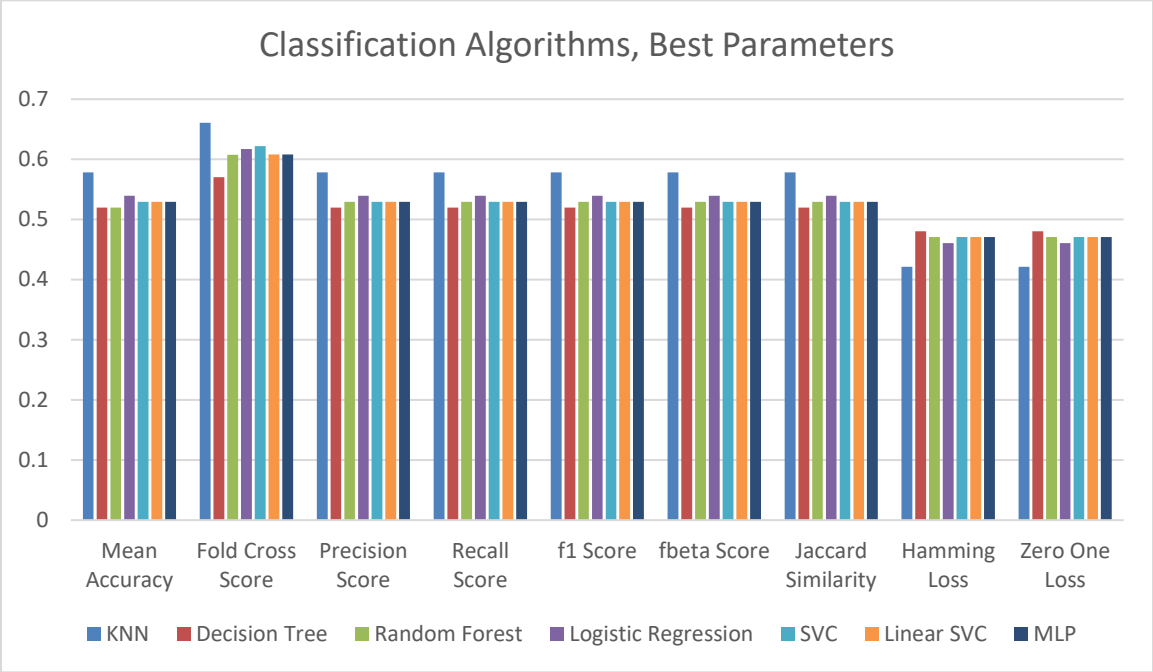
Ο Random Forest και ο SVC παρουσιάζουν τα καλύτερα αποτελέσματα, αλλά πρέπει να συνυπολογίσουμε και τη διακύμανση που παρουσιάζει ο Random Forest. Στον πίνακα που ακολουθεί παραθέτω τα βέλτιστα αποτελέσματα των classifiers.

	KNN	DT	RF	LR	SVC	L. SVC	MLP
Train Time(s)	0.0	0.0150	0.0160	0.0149	0.0509	0.0469	0.0285
Score Time(s)	0.0	0.0	0.0	0.0	0.001	0.0150	0.0009
Mean Accuracy	0.5784	0.5196	0.5196	0.5392	0.5392	0.5294	0.5294
Fold Cross Score	0.6609	0.5703	0.6075	0.6169	0.6346	0.6078	0.6078
Prec Score, micro	0.5784	0.5196	0.5294	0.5392	0.5392	0.5294	0.5294
Prec Score, macro	0.5927	0.2121	0.1636	0.2557	0.2557	0.0882	0.0882
Prec Score, weighted	0.7457	0.4391	0.4153	0.5771	0.5771	0.2802	0.2802
Recall Score, micro	0.5784	0.5196	0.5294	0.5392	0.5392	0.5294	0.5294
Recall Score, macro	0.2414	0.1855	0.1765	0.1722	0.1722	0.1666	0.1666
Recall Score, weighted	0.5784	0.5196	0.5294	0.5392	0.5392	0.5294	0.5294
f1 Score, micro	0.5784	0.5196	0.5294	0.5392	0.5392	0.5294	0.5294
f1 Score, macro	0.2468	0.1690	0.1475	0.1268	0.1268	0.1153	0.1153
f1 Score, weighted	0.4649	0.4296	0.4204	0.3878	0.3878	0.3665	0.3665
fbeta Score, micro	0.5784	0.5196	0.5294	0.5392	0.5392	0.5294	0.5294
fbeta Score, macro	0.3259	0.1814	0.1483	0.1227	0.1227	0.0974	0.0974
fbeta Score, weighted	0.4847	0.4173	0.3998	0.3553	0.3553	0.3093	0.3093
Jaccard Similarity	0.5784	0.5196	0.5294	0.5392	0.5392	0.5294	0.5294
Hamming Loss	0.4215	0.4803	0.4705	0.4607	0.4607	0.4705	0.4705

Πίνακας 44: Σύγκριση classifiers, λεξικό Opinion Observer, βέλτιστοι παράμετροι εισόδου, δεδομένα από Twitter

Με τα νέα σκορ που επιτυγχάνουν οι classifiers, ο αλγόριθμος του KNN ξεχωρίζει από τους υπόλοιπους, αφού παρουσιάζει υψηλότερο σκορ σε όλες τις μετρικές. Ο KNN σε ακόμη ένα λεξικό παρουσίασε εντυπωσιακή βελτίωση όταν εκτελέστηκε με τις προτεινόμενες παραμέτρους και αφού τα δεδομένα εισόδου δέχθηκαν προεπεξεργασία.

Παρακάτω φαίνεται και οπτικά η απόδοση των αλγορίθμων, με τη χρήση βέλτιστων παραμέτρων και έχοντας τα δεδομένα εισόδου υποστούν προεπεξεργασία



Διάγραμμα 48: Σύγκριση classifiers, λεξικό Opinion Observer, βέλτιστοι παράμετροι εισόδου

Το λεξικό του Opinion Observer παρουσιάζει ελαφρώς χειρότερα αποτελέσματα συγκριτικά με το λεξικό του AFINN και το λεξικό του imdb και σαφώς καλύτερα από τα λεξικά των Amazon/TripAdvisor, Goodreads και Opentable.

6.9 Αποτελέσματα του λεξικού SentiWordNet

Σε αυτό το υποκεφάλαιο μελετάω τη συμπεριφορά των classifiers όταν τα δεδομένα που δέχονται έχουν αξιολογηθεί από το λεξικό του SentiWordNet. Υπενθυμίζω ότι για το συγκεκριμένο λεξικό δημιούργησα δύο διαφορετικές εκδοχές, στην πρώτη κράτησα την πρώτη τιμή στην οποία συναντάται η λέξη και στη δεύτερη το μέγιστη τιμή που λαμβάνει μία λέξη.

6.9.1 Αποτελέσματα στα δεδομένα που εξαχθεί από το Facebook

Τα αποτελέσματα των classifiers όταν δέχονται δεδομένα που έχουν εξαχθεί από το Facebook και έχουν αξιολογηθεί από τη πρώτη έκδοση του λεξικού, αυτή που θέτει σκορ στις διπλοεγγραφές την πρώτη τιμή που συναντάει, παρουσιάζονται στον πίνακα που ακολουθεί.

	KNN	DT	RF	LR	SVC
Train Time(s)	0.0	0.0	0.015	0.0319	0.156
Score Time(s)	0.0	0.0	0.0159	0.015	0.0
Mean Accuracy	0.4215	0.4019	0.4901	0.3823	0.447
Fold Cross Score	0.4409	0.3588	0.5271	0.494	0.4602
Prec Score, micro	0.4215	0.4509	0.5196	0.3823	0.447
Prec Score, macro	0.0565	0.1083	0.1899	0.0488	0.0343
Prec Score, weighted	0.2959	0.4389	0.4610	0.2818	0.1998
Recall Score, micro	0.4215	0.4509	0.5196	0.3823	0.447
Recall Score, macro	0.0725	0.1275	0.1448	0.0659	0.0769
Recall Score, weighted	0.4215	0.4509	0.5196	0.3823	0.447
f1 Score, micro	0.4215	0.45090	0.5196	0.3823	0.447
f1 Score, macro	0.0623	0.1149	0.1448	0.0561	0.0475
f1 Score, weighted	0.344	0.4406	0.4555	0.3245	0.2762
fbeta Score, micro	0.4215	0.4509	0.5196	0.3823	0.447
fbeta Score, macro	0.0584	0.1104	0.159	0.0515	0.0386
fbeta Score, weighted	0.3126	0.4388	0.4425	0.2975	0.2247
Jaccard Similarity	0.4215	0.4509	0.5196	0.3823	0.447
Hamming Loss	0.5784	0.5490	0.4803	0.6176	0.5529
Zero One Loss	0.5784	0.5490	0.4803	0.6176	0.5529

Πίνακας 45: Σύγκριση classifiers, λεξικό SentiWordNet, πρώτη έκδοση, default παράμετροι εισόδου, δεδομένα από Facebook

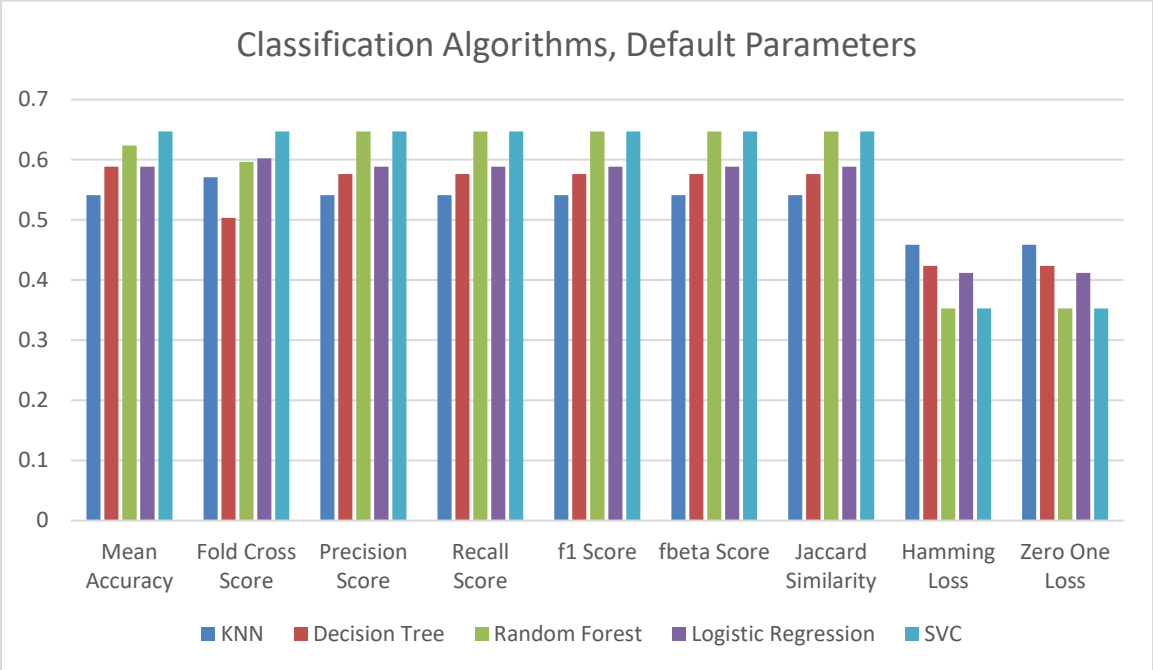
Με την πρώτη έκδοση του λεξικού κανένας classifiers δεν καταφέρνει να ξεπεράσει σε καμία μετρική το 0.5, οπότε εξετάζω και την δεύτερη εκδοχή του λεξικό παρακάτω.

	KNN	DT	RF	LR	SVC
Train Time(s)	0.0	0.0	0.015000	0.031000	0.062
Score Time(s)	0.0	0.0	0.0	0.0	0.0
Mean Accuracy	0.541176	0.588235	0.623529	0.588235	0.647
Fold Cross Score	0.570695	0.503145	0.596444	0.602218	0.647
Prec Score, micro	0.541176	0.576470	0.647058	0.588235	0.647
Prec Score, macro	0.105022	0.185185	0.168055	0.131193	0.1078
Prec Score, weighted	0.407735	0.523921	0.472058	0.446104	0.4186
Recall Score, micro	0.541176	0.576470	0.647058	0.588235	0.647
Recall Score, macro	0.139393	0.207407	0.182154	0.162373	0.1309
Recall Score, weighted	0.541176	0.576470	0.647058	0.588235	0.5084
f1 Score, micro	0.541176	0.576470	0.647058	0.588235	0.647
f1 Score, macro	0.119791	0.195437	0.161111	0.143281	0.1309

f1 Score, weighted	0.465073	0.548735	0.535294	0.505681	0.5084
fbeta Score, micro	0.541176	0.576470	0.647058	0.588235	0.647
fbeta Score, macro	0.110470	0.189108	0.159682	0.135273	0.116
fbeta Score, weighted	0.428886	0.533532	0.491092	0.467693	0.4504
Jaccard Similarity	0.541176	0.576470	0.647058	0.588235	0.647
Hamming Loss	0.458823	0.423529	0.352941	0.411764	0.3529
Zero One Loss	0.458823	0.423529	0.352941	0.411764	0.3529

Πίνακας 46: Σύγκριση classifiers, λεξικό SentiWordNet, δεύτερη έκδοση, default παράμετροι εισόδου, δεδομένα από Facebook

Αντίστοιχα χρησιμοποιώντας τη δεύτερη εκδοχή του λεξικού, στην οποία ελαχιστοποιούνται οι μηδενικές τιμές τα αποτελέσματα είναι καλύτερα σε όλες τις μετρικές και η βελτίωση αγγίζει το 20% στον classifier του Logistic Regression.



Διάγραμμα 49: Σύγκριση classifiers, λεξικό SentiWordNet, default παράμετροι εισόδου, δεδομένα από Facebook

Γίνεται σαφές ότι είναι προτιμότερο να υπάρχουν τιμές στο σκορ που λαμβάνει κάθε λέξη, είτε αυτές οι τιμές είναι θετικές είτε αρνητικές. Η παραπάνω παρατήρηση γίνεται ακόμα πιο σημαντική αν συνυπολογίσουμε το γεγονός ότι οι διπλοεγγραφές που λαμβάνουν μία μηδενική τιμή, δύσκολα στη δεύτερη εγγραφή θα έχουν μεγάλο σκορ, αλλά και το μεγαλύτερο δυνατό σκορ είναι ίσο με τη μονάδα, που δεν είναι πολύ μεγάλο, αν συγκριθεί με τις τιμές που δίνει το λεξικό του imdb, το καλύτερο λεξικό που έχω εξετάσει μέχρι τώρα. Εκτός της περίπτωσης των μηδενικών τιμών πρέπει να συνυπολογίσω και τις περιπτώσεις όπου αμφισήμαντες λέξεις, που κάποιες φορές λαμβάνουν θετικές τιμές και κάποιες άλλες αρνητικές, ενδέχεται στην αξιολόγηση μίας πρότασης να λάβουν όλες οι λέξεις θετική ή αρνητική χροιά, με αποτέλεσμα το συνολικό σκορ της λέξης να λάβει μεγάλη τιμή.

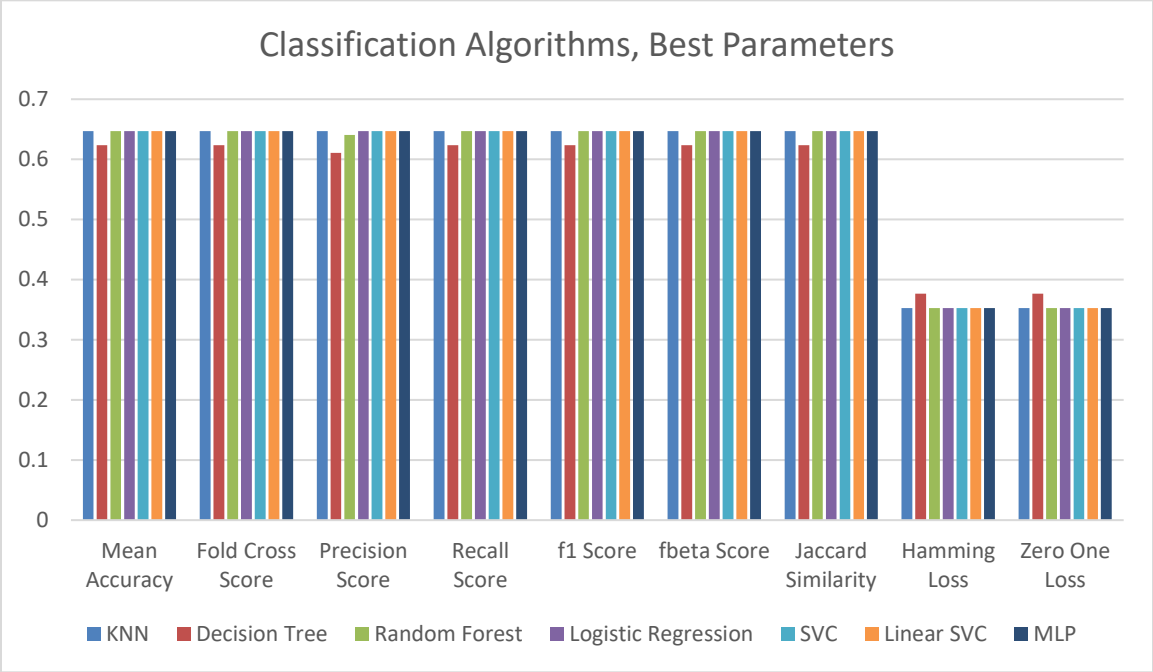
Για λόγους σύμβασης από εδώ και στο εξής όταν θα αναφέρομαι σε αυτό το κεφάλαιο στο ‘λεξικό’ ή στα επόμενα κεφάλαια στο ‘λεξικό του SentiWordNet’ θα αναφέρομαι στη δεύτερη εναλλακτική του λεξικού.

Τα αποτελέσματα των classifiers όταν αυτοί εκτελούνται με τις βέλτιστες παραμέτρους φαίνονται στον πίνακα που ακολουθεί.

	KNN	DT	RF	LR	SVC	L. SVC	MLP
Train Time(s)	0.001	0.0019	0.0409	0.0409	0.013	0.0	0.7126
Score Time(s)	0.0019	0.0009	0.002	0.0	0.002	0.0	0.0013
Mean Accuracy	0.647	0.6352	0.647	0.647	0.647	0.647	0.647
Fold Cross Score	0.647	0.6109	0.6407	0.647	0.647	0.647	0.647
Prec Score, micro	0.647	0.6235	0.647	0.647	0.647	0.647	0.647
Prec Score, macro	0.1078	0.1064	0.1078	0.1078	0.1078	0.1078	0.1078
Prec Score, weighted	0.4186	0.4131	0.4186	0.4186	0.4186	0.4186	0.4186
Recall Score, micro	0.647	0.6235	0.647	0.647	0.647	0.647	0.647
Recall Score, macro	0.1666	0.1606	0.1666	0.1666	0.1666	0.1666	0.1666
Recall Score, weighted	0.647	0.6235	0.647	0.647	0.647	0.647	0.647
f1 Score, micro	0.647	0.6235	0.647	0.647	0.647	0.647	0.647
f1 Score, macro	0.13	0.128	0.1309	0.1309	0.1309	0.1309	0.1309
f1 Score, weighted	0.5084	0.4970	0.5084	0.5084	0.5084	0.5084	0.5084
fbeta Score, micro	0.6470	0.6235	0.647	0.647	0.647	0.6470	0.647
fbeta Score, macro	0.1160	0.1141	0.116	0.1160	0.116	0.1160	0.116
fbeta Score, weighted	0.4504	0.4430	0.4504	0.4504	0.4504	0.4504	0.4504
Jaccard Similarity	0.6470	0.6235	0.647	0.647	0.6470	0.647	0.647
Hamming Loss	0.3529	0.3764	0.3529	0.3529	0.3529	0.3529	0.3529
Zero One Loss	0.3529	0.3764	0.3529	0.3529	0.3529	0.3529	0.3529

Πίνακας 47: Σύγκριση classifiers, λεξικό SentiWordNet, βέλτιστοι παράμετροι εισόδου, δεδομένα από Facebook

Οι classifiers παρουσιάζουν όλοι εξαιρετικά κοντινά αποτελέσματα. Ο μόνος classifier που έχει ένα μειονέκτημα είναι ο Decision Tree, ο οποίος παρουσιάζει και διακύμανση σε όλες τις μετρικές. Μία διακύμανση η οποία δεν περιορίστηκε ούτε με τη χρήση διαφορετικών παράμετρών ούτε με την προεπεξεργασία δεδομένων.



Διάγραμμα 50: Σύγκριση classifiers, λεξικό SentiWordNet, βέλτιστοι παράμετροι εισόδου, δεδομένα από Facebook

Σε γενικές γραμμές το λεξικό του SentiWordNet παρουσιάζει καλή απόδοση, το μόνο λεξικό που παρουσιάζει καλύτερη απόδοση είναι το λεξικό του imdb. Ακόμη από πλευράς απόδοσης των classifiers το λεξικό παρουσιάζει ομοιότητες με το λεξικό του AFINN.

6.9.2 Αποτελέσματα στα δεδομένα που εξαχθεί από το Twitter

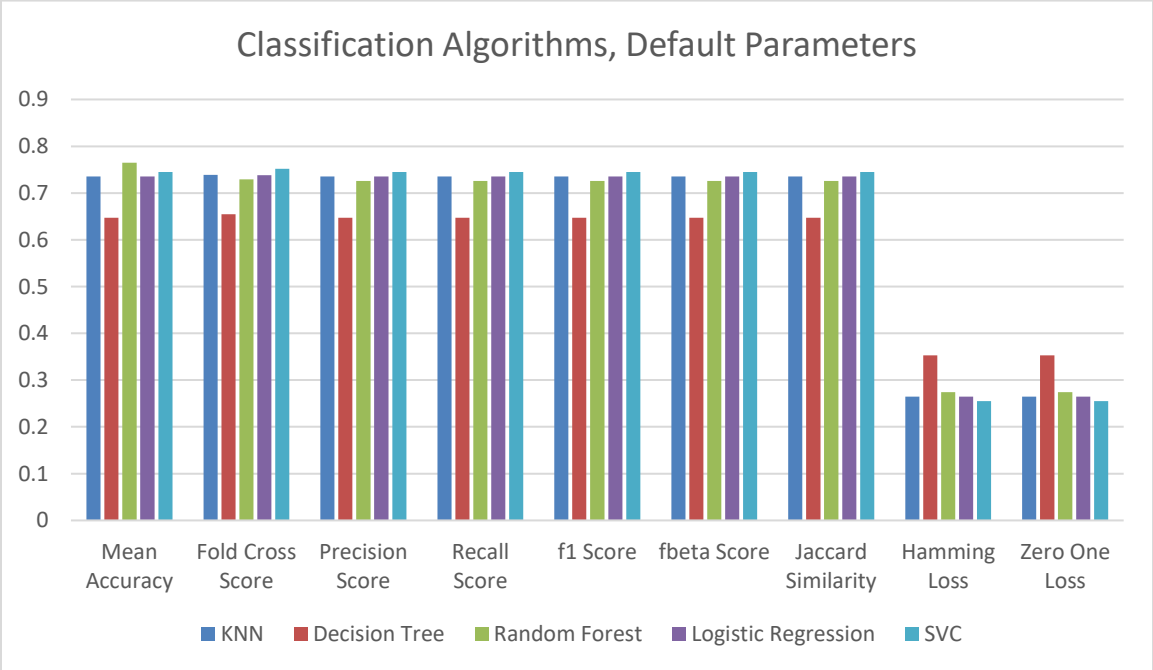
Έχοντας ολοκληρώσει την μελέτη του λεξικού σε δεδομένα που έχουν εξαχθεί από το Facebook, προχωράω στα δεδομένα του Twitter. Περιμένω τα αποτελέσματα των classifiers να είναι ελαφρώς καλύτερα, όταν τα δεδομένα προέρχονται από το Twitter.

	KNN	**DT**	**RF**	**LR**	**SVC**
Train Time(s)	0.0	0.0	0.015	0.015	0.016
Score Time(s)	0.0	0.0	0.016	0.0	0.0
Mean Accuracy	0.7352	0.647	0.7647	0.7352	0.745
Fold Cross Score	0.7388	0.6546	0.7291	0.7381	0.7517
Prec Score, micro	0.7352	0.647	0.7254	0.7352	0.745
Prec Score, macro	0.125	0.2478	0.3472	0.1836	0.1241
Prec Score, weighted	0.5588	0.6572	0.6013	0.6183	0.5551

Recall Score, micro	0.7352	0.647	0.7254	0.7352	0.745
Recall Score, macro	0.1644	0.3101	0.265	0.1838	0.1666
Recall Score, weighted	0.7352	0.647	0.7254	0.7352	0.745
f1 Score, micro	0.7352	0.647	0.7254	0.7352	0.745
f1 Score, macro	0.1420	0.2606	0.2839	0.1756	0.1423
f1 Score, weighted	0.635	0.6490	0.6506	0.6636	0.6362
fbeta Score, micro	0.7352	0.6470	0.7254	0.7352	0.745
fbeta Score, macro	0.1313	0.2511	0.3131	0.1772	0.1308
fbeta Score, weighted	0.5869	0.6535	0.6175	0.6325	0.5849
Jaccard Similarity	0.7352	0.647	0.7254	0.7352	0.745
Hamming Loss	0.2647	0.3529	0.2745	0.2647	0.2549
Zero One Loss	0.2647	0.3529	0.2745	0.2647	0.2549

Πίνακας 48: Σύγκριση classifiers, λεξικό SentiWordNet, default παράμετροι εισόδου, δεδομένα από Twitter

Οι classifiers όταν δέχονται δεδομένα που έχουν εξαχθεί από το Twitter παρουσιάζουν καλύτερα αποτελέσματα συγκριτικά με τα δεδομένα που έχουν εξαχθεί από το Facebook. Η βελτίωση ήταν αναμενόμενη, καθώς είχε παρατηρηθεί επίσης όταν χρησιμοποιήθηκαν τα λεξικά του AFINN και του Opinion Observer. Όλοι οι classifiers, πλην του Decision Tree, παρουσιάζουν παρόμοια αποτελέσματα και δεν ξεχωρίζει κάποιος ούτε σε απόδοση ούτε σε χρόνο εκτέλεσης. Στο παρακάτω διάγραμμα φαίνεται και οπτικά η απόδοση των classifiers.



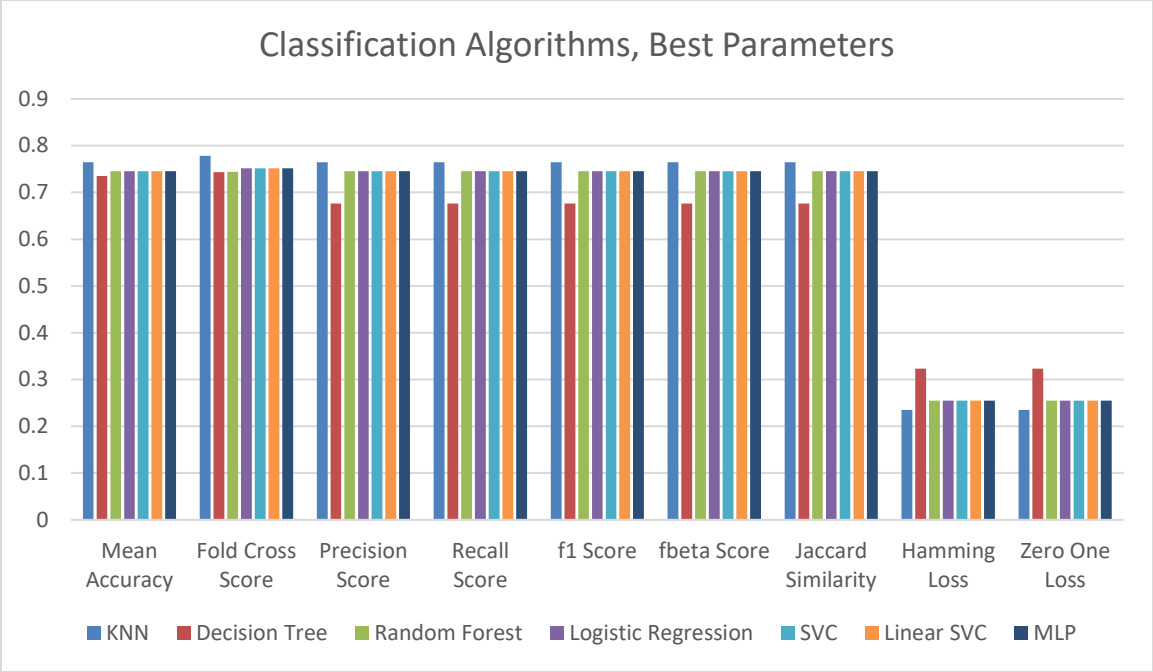
Διάγραμμα 51: Σύγκριση classifiers, λεξικό SentiWordNet, default παράμετροι εισόδου, δεδομένα από Twitter

Τα αποτελέσματα στις μετρικές που χρησιμοποιώ για όλους τους classifiers με τη χρήση των βέλτιστων παραμέτρων και εφόσον τα δεδομένα εισόδου έχουν υποστεί προεπεξεργασία φαίνονται στον παρακάτω πίνακα

	KNN	DT	RF	LR	SVC	L. SVC	MLP
Train Time(s)	0.0	0.0	0.015000	0.030999	0.0	0.0309	0.5277
Score Time(s)	0.0	0.0	0.0	0.0	0.0	0.0	0.0012
Mean Accuracy	0.7647	0.7352	0.745	0.745	0.7647	0.7450	0.7450
Fold Cross Score	0.7784	0.7436	0.7439	0.7517	0.7784	0.7517	0.7517
Prec Score, micro	0.7647	0.6764	0.745	0.745	0.7647	0.745	0.7450
Prec Score, macro	0.4599	0.1396	0.1241	0.1241	0.4599	0.1241	0.1241
Prec Score, weighted	0.6545	0.5692	0.5551	0.5551	0.6545	0.5551	0.5551
Recall Score, micro	0.7647	0.6764	0.745	0.745	0.7647	0.745	0.7450
Recall Score, macro	0.2738	0.161	0.1666	0.1666	0.2738	0.1666	0.1666
Recall Score, weighted	0.7647	0.6764	0.745	0.745	0.7647	0.745	0.745
f1 Score, micro	0.7647	0.6764	0.745	0.745	0.7647	0.745	0.745
f1 Score, macro	0.2967	0.1490	0.1423	0.1423	0.2967	0.1423	0.1423
f1 Score, weighted	0.6737	0.6177	0.6362	0.6362	0.6737	0.6362	0.6362
fbeta Score, micro	0.7647	0.6764	0.745	0.745	0.7647	0.745	0.745
fbeta Score, macro	0.3476	0.1431	0.1308	0.1308	0.3476	0.1308	0.1308
fbeta Score, weighted	0.6423	0.5875	0.5849	0.5849	0.6423	0.5849	0.5849
Jaccard Similarity	0.7647	0.6764	0.745	0.745	0.7647	0.745	0.7450
Hamming Loss	0.2352	0.3235	0.2549	0.2549	0.2352	0.2549	0.2549

Πίνακας 49: Σύγκριση classifiers, λεξικό SentiWordNet, βέλτιστοι παράμετροι εισόδου, δεδομένα από Twitter

Οι classifiers όταν εκτελούνται με τις βέλτιστες παραμέτρους παρουσιάζουν ελαφρώς καλύτερα αποτελέσματα ή στην περίπτωση του Decision Tree αισθητά καλύτερα. Γραφικά οι αποδόσεις των classifers φαίνονται στο παρακάτω διάγραμμα.



Διάγραμμα 52: Σύγκριση classifiers, λεξικό SentiWordNet, βέλτιστοι παράμετροι εισόδου

Στο διάγραμμα γίνεται καλύτερα αντιληπτή η μικρή διαφορά που έχουν στην απόδοση οι classifiers. Οι συγκεκριμένες μετρήσεις ενισχύουν την παρατήρηση ότι εφόσον οι classifiers δεχθούν τη μέγιστη δυνατή τροποποίηση παρουσιάζουν πολύ κοντινά σκορ και είναι δύσκολο κάποιος να διαφοροποιηθεί σημαντικά.

6.10 Αποτελέσματα του λεξικού Subjectivity

Σε αυτό το υποκεφάλαιο μελετώ τη συμπεριφορά των classifiers όταν εξάγεται σκορ από το λεξικό του Subjectivity. Υπενθυμίζω ότι οι λέξεις που υπάρχουν στο λεξικό βαθμολογούνται στο εύρος [-2,2] με ακέραια τιμή, ανάλογα με την πολικότητα και την βαρύτητα τους.

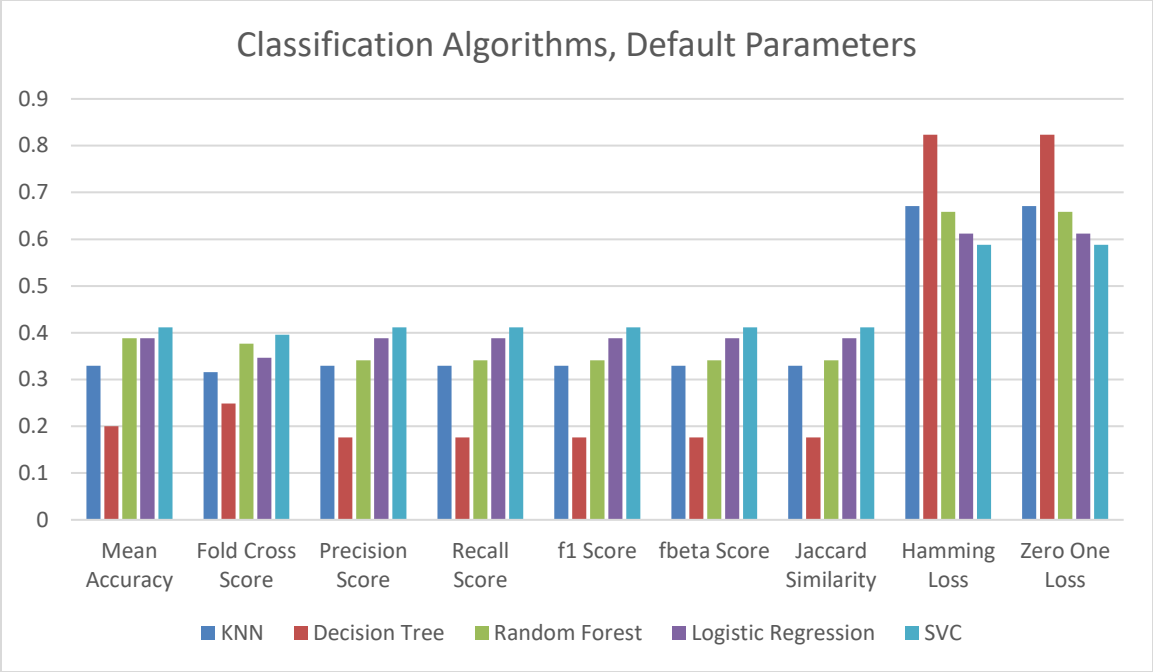
6.10.1 Αποτελέσματα στα δεδομένα που εξαχθεί από το Facebook

Στον πίνακα που ακολουθεί παραθέτω τα αποτελέσματα των classifiers όταν εκτελούνται με τις default παραμέτρους και δέχονται δεδομένα από το Facebook που έχουν αξιολογηθεί από το λεξικό του Subjectivity.

	KNN	DT	RF	LR	SVC
Train Time(s)	0.0	0.0	0.0159	0.1	0.0309
Score Time(s)	0.0	0.0	0.0	0.0	0.0
Mean Accuracy	0.3294	0.2	0.3882	0.3882	0.4117
Fold Cross Score	0.3159	0.2487	0.3763	0.3465	0.3957
Prec Score, micro	0.3294	0.1764	0.3411	0.3882	0.4117
Prec Score, macro	0.0544	0.0433	0.129	0.0543	0.0316
Prec Score, weighted	0.2204	0.2014	0.3083	0.2838	0.1695
Recall Score, micro	0.3294	0.1764	0.3411	0.3882	0.4117
Recall Score, macro	0.0694	0.0385	0.0797	0.0674	0.0769
Recall Score, weighted	0.3294	0.1764	0.3411	0.3882	0.4117
f1 Score, micro	0.3294	0.1764	0.3411	0.3882	0.4117
f1 Score, macro	0.056	0.0407	0.0791	0.0585	0.0448
f1 Score, weighted	0.2495	0.188	0.2853	0.3221	0.2401
fbeta Score, micro	0.3294	0.1764	0.3411	0.3882	0.4117
fbeta Score, macro	0.0532	0.0422	0.0927	0.0556	0.0358
fbeta Score, weighted	0.2256	0.1958	0.2801	0.2966	0.1921
Jaccard Similarity	0.3294	0.1764	0.3411	0.3882	0.4117
Hamming Loss	0.6705	0.8235	0.6588	0.6117	0.5882
Zero One Loss	0.6705	0.8235	0.6588	0.6117	0.5882

Πίνακας 50: Σύγκριση classifiers, λεξικό Subjectivity, default παράμετροι εισόδου, δεδομένα από Facebook

Οι αποδόσεις των classifiers δεν είναι ικανοποιητικές, καθώς καμία μετρική δεν ξεπερνάει το 0.4. Τα λεξικά των AFINN, Opinion Observer και imdb παρουσιάζουν αισθητά ανώτερα αποτελέσματα. Παρακάτω παραθέτω και το διάγραμμα με την οπτική απεικόνιση της απόδοσης των classifiers με τη χρήση του συγκεκριμένου λεξικού.



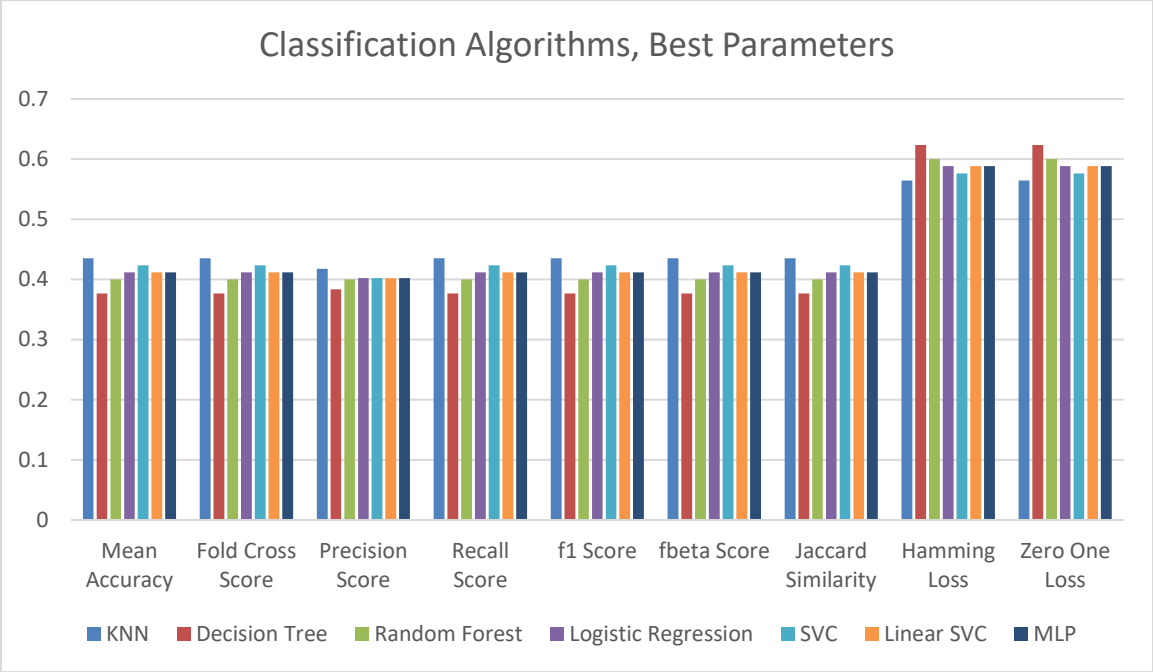
Διάγραμμα 53: Σύγκριση classifiers, λεξικό Subjectivity, default παράμετροι εισόδου, δεδομένα από Facebook

Τα λεξικά που παρουσιάζουν χαμηλά σκορ όταν χρησιμοποιούνται από τους classifiers με τις default παραμέτρους τείνουν να βελτιώνονται αισθητά όταν οι classifiers εκτελούνται με τις βέλτιστες παραμέτρους. Παρακάτω παρουσιάζω τα αποτελέσματα.

	KNN	DT	RF	LR	SVC	L. SVC	MLP
Train Time(s)	0.0	0.0	0.0	0.141	0.078	0.015	1.068
Score Time(s)	0.0	0.0	0.0160	0.0	0.0	0.0	0.0015
Mean Accuracy	0.4352	0.4117	0.4	0.4117	0.4235	0.4117	0.4117
Fold Cross Score	0.4176	0.3837	0.4001	0.4019	0.4019	0.4019	0.4019
Prec Score, micro	0.4352	0.3764	0.4	0.4117	0.4235	0.4117	0.4117
Prec Score, macro	0.1098	0.0315	0.0318	0.0316	0.1089	0.0316	0.0316
Prec Score, weighted	0.3058	0.0315	0.1707	0.1695	0.3598	0.1695	0.1695
Recall Score, micro	0.4352	0.3764	0.4	0.4117	0.4235	0.4117	0.4117
Recall Score, macro	0.0997	0.0703	0.0747	0.0316	0.0817	0.0769	0.0769
Recall Score, weighted	0.4352	0.3764	0.4	0.1695	0.4235	0.4117	0.4117
f1 Score, micro	0.4352	0.3764	0.4	0.4117	0.4235	0.4117	0.4117
f1 Score, macro	0.0855	0.0435	0.0447	0.0448	0.0542	0.0448	0.0448
f1 Score, weighted	0.3116	0.2332	0.2393	0.2401	0.2643	0.2401	0.2401
fbeta Score, micro	0.4352	0.3764	0.4	0.4117	0.4235	0.4117	0.4117
fbeta Score, macro	0.0933	0.0354	0.036	0.0358	0.0555	0.0358	0.0358
fbeta Score, weighted	0.2938	0.1898	0.1928	0.1921	0.2412	0.1921	0.1921
Jaccard Similarity	0.4352	0.3764	0.4	0.4117	0.4235	0.4117	0.4117
Hamming Loss	0.5647	0.6235	0.5999	0.5882	0.5764	0.5882	0.5882
Zero One Loss	0.5647	0.6235	0.5999	0.5882	0.5764	0.5882	0.5882

Πίνακας 51: Σύγκριση classifiers, λεξικό Subjectivity, βέλτιστοι παράμετροι εισόδου, δεδομένα από Facebook

Οι classifiers βελτιώνονται μέχρι και 0.1, ο KNN στην μετρική του fold cross score, αλλά η απόδοση τους συνεχίζει να μην είναι ικανοποιητική. Η οπτική απεικόνιση της απόδοσης των classifiers φαίνεται στο διάγραμμα που ακολουθεί.



Διάγραμμα 54: Σύγκριση classifiers, λεξικό Subjectivity, βέλτιστοι παράμετροι εισόδου, δεδομένα από Facebook

Από όλους τους classifiers που έχω χρησιμοποιήσει, και βελτιστοποιήσει με τη χρήση κατάλληλων παραμέτρων και επεξεργασίας των δεδομένων, ο αλγόριθμος του KNN παρουσιάζει ελαφρώς καλύτερα αποτελέσματα από τους ανταγωνιστές τους. Η δεύτερη επιλογή αλγορίθμου θα ήταν ο SVC, εξαιτίας της καλής τους απόδοσης με τη χρήση των default παραμέτρων. Βεβαία σε ακόμη ένα λεξικό διακρίνεται η μικρή σημασία στην επιλογή του classifier για την αξιολόγηση λεξικών, αφού όλοι οι classifiers παρουσιάζουν πολύ κοντινά αποτελέσματα.

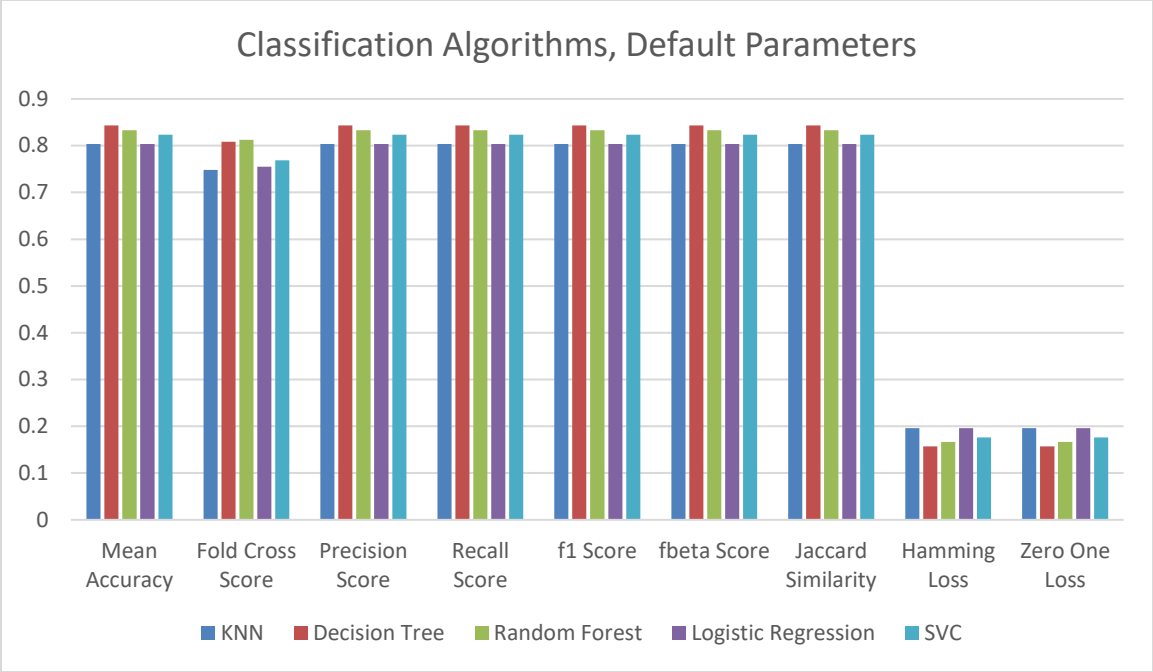
6.10.2 Αποτελέσματα στα δεδομένα που εξαχθεί από το Twitter

Έχοντας ολοκληρώσει την μελέτη της συμπεριφοράς των classifiers όταν δέχονται ως είσοδο δεδομένα από το Facebook, αξιολογώντας τα με βάση το λεξικό του Subjectivity, προχωράω στη μελέτη συμπεριφοράς των ίδιων αλγορίθμων όταν δέχονται δεδομένα που προέρχονται από το Twitter.

	KNN	DT	RF	LR	SVC
Train Time(s)	0.0	0.002	0.0179	0.0139	0.016
Score Time(s)	0.0	0.0009	0.0009	0.001	0.0
Mean Accuracy	0.8039	0.8431	0.8333	0.8039	0.8235
Fold Cross Score	0.7479	0.808	0.8121	0.7551	0.7689
Prec Score, micro	0.8039	0.8431	0.8333	0.8039	0.8235
Prec Score, macro	0.252	0.2929	0.29	0.2645	0.3639
Prec Score, weighted	0.7399	0.8269	0.7939	0.7448	0.8258
Recall Score, micro	0.8039	0.8431	0.8333	0.8039	0.8235
Recall Score, macro	0.2279	0.3123	0.2725	0.2279	0.2235
Recall Score, weighted	0.8039	0.8431	0.8333	0.8039	0.8235
f1 Score, micro	0.8039	0.8431	0.8333	0.8039	0.8235
f1 Score, macro	0.2285	0.3018	0.2779	0.2297	0.2223
f1 Score, weighted	0.7592	0.8343	0.8095	0.757	0.7595
fbeta Score, micro	0.8039	0.8431	0.8333	0.8039	0.8235
fbeta Score, macro	0.2376	0.2963	0.2841	0.2426	0.2501
fbeta Score, weighted	0.7428	0.8297	0.799	0.7424	0.7505
Jaccard Similarity	0.8039	0.8431	0.8333	0.8039	0.8235
Hamming Loss	0.196	0.1568	0.1666	0.196	0.1764
Zero One Loss	0.196	0.1568	0.1666	0.196	0.1764

Πίνακας 52: Σύγκριση classifiers, λεξικό Subjectivity, default παράμετροι εισόδου, δεδομένα από Twitter

Η απόδοση των classifiers όταν δέχονται δεδομένα που προέρχονται από το Twitter είναι πολύ ανώτερη σε σχέση με τα δεδομένα που προέρχονται από το Facebook. Η βελτίωση της απόδοσης ήταν αναμενομένη, αλλά όχι σε τόσο μεγάλο βαθμό. Υπάρχει διπλασιασμός της απόδοσης όλων των classifiers. Οι classifiers όταν χρησιμοποιούν το λεξικό του Subjectivity ξεπερνούν σε απόδοση οποιοδήποτε άλλο λεξικό, όταν έχουν ως είσοδο τα δεδομένα του Twitter. Για να γίνει καλύτερα κατανοητή η εκτόξευση των σκορ των classifiers όταν δέχονται δεδομένα από το Twitter παραθέτω το παρακάτω διάγραμμα.



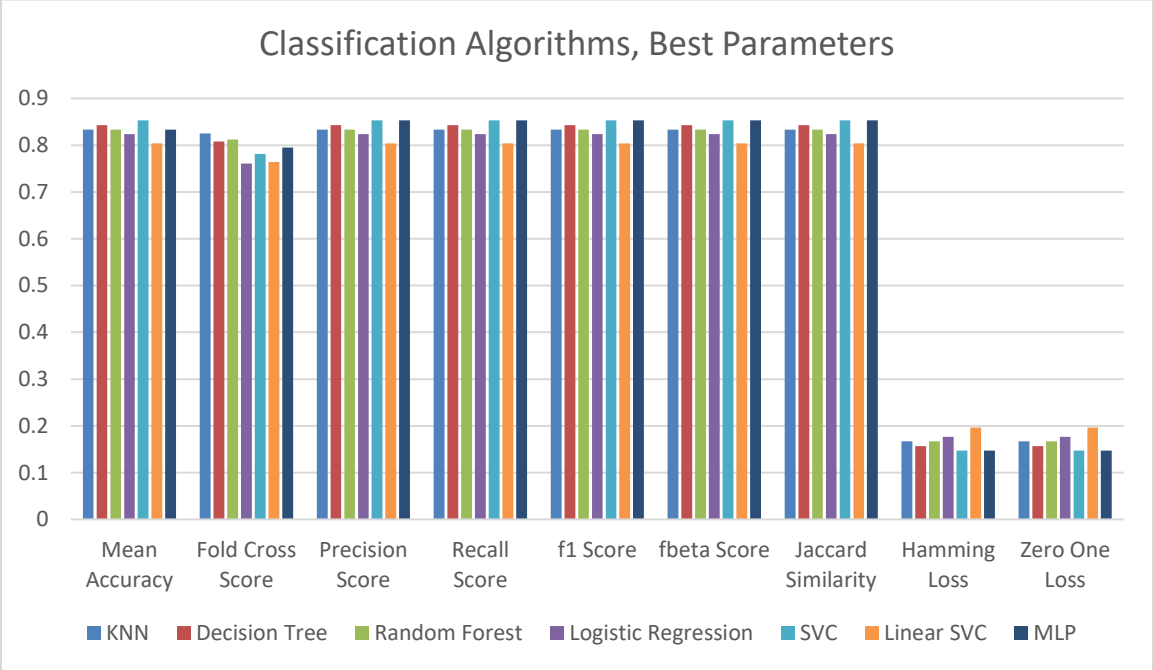
Διάγραμμα 55: Σύγκριση classifiers, λεξικό Subjectivity, default παράμετροι εισόδου, δεδομένα από Twitter

Οι classifiers παρουσιάζουν εντυπωσιακή αύξηση στα σκορ συγκριτικά με την απόδοση του όταν δέχονται δεδομένα από το Facebook. Επίσης δεν υπάρχει κάποιος classifier να ξεχωρίζει από πλευράς απόδοσης. Όταν οι classifies εκτελούνται με τις βέλτιστες παραμέτρους εμφανίζουν ελάχιστα καλύτερα αποτελέσματα.

	KNN	DT	RF	LR	SVC	L. SVC	MLP
Train Time(s)	0.003	0.002	0.0179	0.0549	0.03	0.1289	0.9101
Score Time(s)	0.0269	0.0009	0.0009	0.0	0.0019	0.0	0.0013
Mean Accuracy	0.8333	0.8431	0.8333	0.8235	0.8529	0.8039	0.8333
Fold Cross Score	0.825	0.808	0.8121	0.7608	0.7815	0.7641	0.795
Prec Score, micro	0.8333	0.8431	0.8333	0.8235	0.8529	0.8039	0.8529
Prec Score, macro	0.302	0.2929	0.29	0.281	0.3275	0.1607	0.3137
Prec Score, weighted	0.7894	0.8269	0.7939	0.7754	0.8211	0.6462	0.8199
Recall Score, micro	0.8333	0.8431	0.8333	0.8235	0.8529	0.8039	0.8529
Recall Score, macro	0.2446	0.3123	0.2725	0.2515	0.2774	0.2	0.2961
Recall Score, weighted	0.8333	0.8431	0.8333	0.8235	0.8529	0.8039	0.8529
f1 Score, micro	0.8333	0.8431	0.8333	0.8235	0.8529	0.8039	0.8529
f1 Score, macro	0.2515	0.3018	0.2779	0.2574	0.2905	0.1782	0.3024
f1 Score, weighted	0.7896	0.8343	0.8095	0.7899	0.8247	0.7165	0.8334
fbeta Score, micro	0.8333	0.8431	0.8333	0.8235	0.8529	0.8039	0.8529
fbeta Score, macro	0.2713	0.2963	0.2841	0.2683	0.3082	0.1673	0.3085
fbeta Score, weighted	0.7799	0.8297	0.7990	0.778	0.8183	0.6726	0.8245
Jaccard Similarity	0.8333	0.8431	0.8333	0.8235	0.8529	0.8039	0.8529
Hamming Loss	0.1666	0.1568	0.1666	0.1764	0.147	0.196	0.147

Πίνακας 53: Σύγκριση classifiers, λεξικό Subjectivity, βέλτιστοι παράμετροι εισόδου, δεδομένα από Twitter

Από το παραπάνω πίνακα φαίνεται ότι όλοι οι classifiers είναι κοντά σε απόδοση, εφόσον έχω πραγματοποιήσει τη βελτιστοποίηση τους. Δεν υπάρχει κάποιος αλγόριθμος ο οποίος να υπερτερεί σε όλες τις μετρικές που χρησιμοποιώ, κάνοντας δύσκολη την κατάταξη τους από πλευράς απόδοσης. Στο διάγραμμα που ακολουθεί φαίνεται η απόδοση των αλγορίθμων οπτικά.



Διάγραμμα 56: Σύγκριση classifiers, λεξικό Subjectivity, βέλτιστοι παράμετροι εισόδου

Από την γραφική απεικόνιση των classifiers συνεχίζει να είναι δύσκολη η εύρεση του καλύτερου classifier. Ο SVC υπερτερεί σε όλες τις μετρικές πλην του fold cross score, στο οποίο εμφανίζει σχετικά χαμηλό σκορ, ενώ ο KNN υπερτερεί στο fold cross score και παρουσιάζει καλά αποτελέσματα σε όλες τις άλλες μετρικές. Αλλά σε ακόμη ένα λεξικό οι αποδόσεις των classifiers είναι ιδιαίτερα κοντινές.

6.11 Αποτελέσματα του λεξικού inquirer

Το τελευταίο λεξικό που μελετάω είναι το inquirer, ένα λεξικό που διαθέτει 8,698 μη μηδενικές εγγραφές και 156,585 συνολικές εγγραφές, με μεγάλη συγκέντρωση στο εύρος [-2.5,2.5].

6.11.1 Αποτελέσματα στα δεδομένα που εξαχθεί από το Facebook

Το τελευταίο λεξικό που μελετάω είναι το inquirer, ένα λεξικό που διαθέτει 8,698 μη μηδενικές εγγραφές και 156,585 συνολικές εγγραφές, με μεγάλη συγκέντρωση στο εύρος [-2.5,2.5].

Πριν τη διαδικασία εφαρμογής των classifiers στα δεδομένα του Facebook τα οποία έχουν αξιολογηθεί από το λεξικό του inquirer για την εξαγωγή των σκορ των classifiers, πρέπει να αναφέρω κάποια στοιχεία παραπάνω για τον τρόπο βαθμολόγησης του λεξικού. Οι περισσότερες εγγραφές της βάσης δεδομένων, το 82.33% των συνολικών, βαθμολογούνται με μηδενική τιμή. Αυτό το φαινόμενο οφείλεται εν μέρει και στην απαραίτητη στρογγυλοποίηση που πραγματοποιείται στις βαθμολογίες της κάθε εγγραφής. Ακόμη ένα 7.66% βαθμολογείται με τιμή ίση με -2 και συνολικά υπάρχουν 5 δυνατές τιμές, συμπεριλαμβανομένων των τιμών 0 και -2. Οπότε το 90% των συνολικών εγγραφών ανήκουν σε μία από τις δύο πιθανές κατηγορίες.

Ο συγκεκριμένος τρόπος βαθμολόγησης των εγγραφών σχεδόν αλλάζει το πρόβλημα του classification σε διαφορετικές κλάσεις σε πρόβλημα δυαδικού classification, με τις δύο πιθανές τιμές να είναι 0 και -2.

Από τα λεξικά που έχω δοκιμάσει δεν έχει υπάρξει άλλο λεξικό το οποίο να έχει τόσα λίγα πιθανά σκορ στην αξιολόγηση προτάσεων και τέτοια συσσώρευση σε δύο τιμές. Τα λεξικά του SentiWordNet και του Opinion Observer παρουσιάζουν τα αμέσως μικρότερα σύνολα δυνατών τιμών για τις προτάσεις προς αξιολόγηση, σύνολο εφτά (7) τιμών. Το λεξικό του Opinion Observer παρουσιάζει συσσώρευση 73.33% συσσώρευση στις τιμές 0 και 8, αλλά και πολύ μεγάλη διακύμανση τιμών, ενώ το λεξικό του SentiWordNet παρουσιάζει συγκέντρωση 0.66% στις τιμές 0 και 1. Το λεξικό του imdb δείχνει τη μεγαλύτερη συσσώρευση σε μία τιμή, την μηδενική (0), αλλά οι προτάσεις προς αξιολόγηση λαμβάνουν δέκα (10) διαφορετικές τιμές.

Γίνεται κατανοητό ότι το λεξικό του inquirer έχει διαφορετικό τρόπο αξιολόγησης των προτάσεων, σε σχέση με τα υπόλοιπα λεξικά. Λόγω του ότι είναι πολύ κοντά σε δυαδικό classification περιμένω ανώτερα αποτελέσματα σε σχέση με τα υπόλοιπα λεξικά.

Στον πίνακα που ακολουθεί παραθέτω τα σκορ των classifiers όταν δέχονται δεδομένα τα οποία αξιολογούνται από το λεξικό του inquirer.

	KNN	**DT**	**RF**	**LR**	**SVC**
Train Time(s)	0.0009	0.002	0.019	0.036	0.0939
Score Time(s)	0.002	0.0	0.0009	0.0	0.0
Mean Accuracy	0.8823	0.9294	0.9058	0.9294	0.8941
Fold Cross Score	0.8478	0.8236	0.9133	0.8779	0.8745
Prec Score, micro	0.8823	0.8941	0.8941	0.9294	0.8941
Prec Score, macro	0.3089	0.4272	0.3810	0.4817	0.2235
Prec Score, weighted	0.8343	0.9191	0.8756	0.911	0.7994
Recall Score, micro	0.8823	0.8941	0.8941	0.9294	0.8941
Recall Score, macro	0.2791	0.4445	0.3472	0.3571	0.25
Recall Score, weighted	0.8823	0.8941	0.8941	0.9294	0.8941
f1 Score, micro	0.8823	0.8941	0.8941	0.9294	0.8941
f1 Score, macro	0.2841	0.4348	0.3604	0.3905	0.236
f1 Score, weighted	0.8539	0.9059	0.8833	0.9095	0.8441
fbeta Score, micro	0.8823	0.8941	0.8941	0.9294	0.8941
fbeta Score, macro	0.2947	0.43	0.3716	0.4325	0.2283
fbeta Score, weighted	0.8405	0.9137	0.8782	0.906	0.8167
Jaccard Similarity	0.8823	0.8941	0.8941	0.9294	0.8941
Hamming Loss	0.1176	0.1058	0.1058	0.0705	0.1058
Zero One Loss	0.1176	0.1058	0.1058	0.0705	0.1058

Πίνακας 54: Σύγκριση classifiers, λεξικό inquirer, πρώτη έκδοση, default παράμετροι εισόδου, δεδομένα από Facebook

Το λεξικό του inquirer παρουσιάζει εντυπωσιακά σκορ σε όλες τις μετρικές, ανεξάρτητα από τον classifier που χρησιμοποιείται. Τα σκορ είναι ιδιαίτερα υψηλά, όχι μόνο συγκριτικά με τα υπόλοιπα λεξικά που έχω χρησιμοποιήσει, αλλά και με τη διεθνή βιβλιογραφία. Στο διάγραμμα που ακολουθεί φαίνεται η οπτική απεικόνιση της απόδοσης των classifiers όταν δέχονται δεδομένα που έχουν εξαχθεί από το Facebook.

Τα ιδιαίτερα υψηλά σκορ των classifiers αποδίδονται στη σχεδόν δυαδική αντιμετώπιση των εγγραφών. Κανένα λεξικό από αυτά που έχω χρησιμοποιήσει δεν έχει τόσο υψηλή συγκέντρωση σε μόλις 2 κατηγορίες. Για την πλήρη κατανόηση του λεξικού πολλαπλασιάσω τα σκορ που δίνονται στις εγγραφές τις βάσεις δεδομένων μου με το 10 κι έπειτα προχωράω στην στρογγυλοποίηση και μετά στην δοκιμή των classifiers. Με αυτή την τεχνική μειώνω τις εγγραφές που βαθμολογούνται με 0 στο 74.6% και το αμέσως επόμενο σκορ με τη μεγαλύτερη συχνότητα είναι η τιμή του -16 και συγκεντρώνει το 7.6%. Οι εγγραφές της βάσης δεδομένων αξιολογούνται με 15 διαφορετικές βαθμολογίες. Δηλαδή το 82.2% των εγγραφών συγκεντρώνεται σε δύο τιμές αντί του 90% που ίσχυε στην προηγούμενη κατάσταση.

Με αυτή την αλλαγή το λεξικό θυμίζει αρκετά το λεξικό του imdb, κι επομένως περιμένω και την ανάλογη απόδοση των classifiers. Τα αποτελέσματα με αυτή την έκδοση του λεξικού φαίνονται στον πίνακα που ακολουθεί.

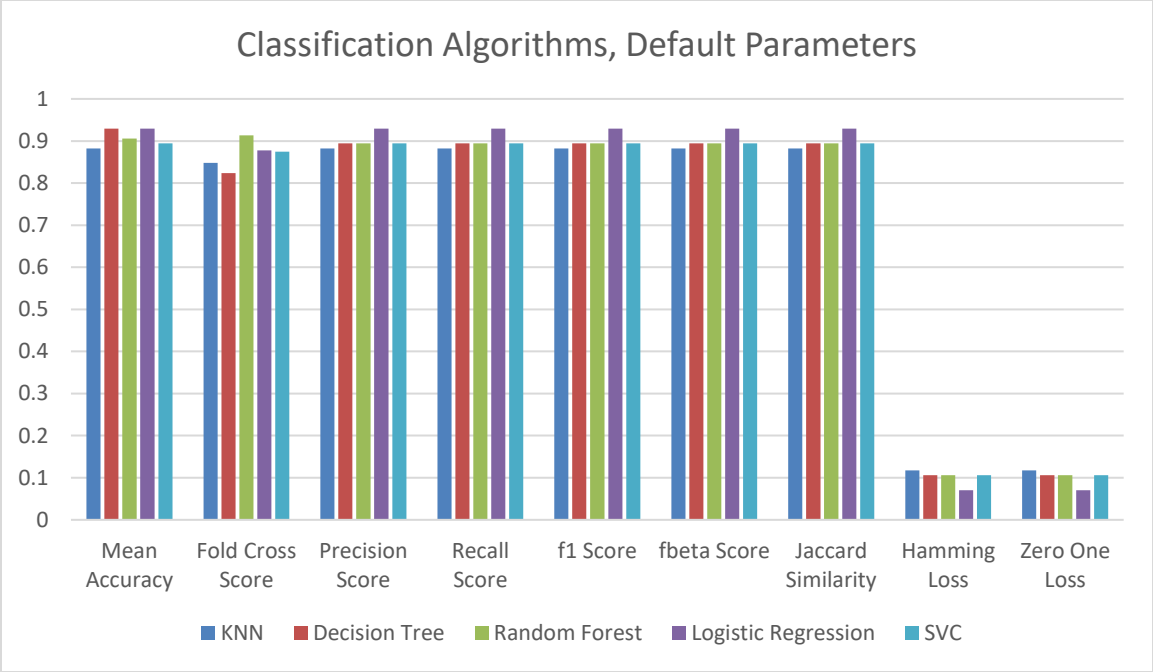
	KNN	DT	RF	LR	SVC
Train Time(s)	0.0	0.002	0.018	0.058	0.203
Score Time(s)	0.002	0.0	0.0029	0.0009	0.0
Mean Accuracy	0.7882	0.8352	0.8352	0.8352	0.8
Fold Cross Score	0.7936	0.7848	0.8392	0.8097	0.8218
Prec Score, micro	0.7882	0.847	0.8352	0.8352	0.8
Prec Score, macro	0.1422	0.33	0.3234	0.2296	0.1
Prec Score, weighted	0.6713	0.8093	0.767	0.7523	0.64
Recall Score, micro	0.7882	0.847	0.8352	0.8352	0.8
Recall Score, macro	0.1391	0.2682	0.2183	0.1945	0.1111
Recall Score, weighted	0.7882	0.847	0.8352	0.8352	0.7111
f1 Score, micro	0.7882	0.847	0.8352	0.8352	0.8
f1 Score, macro	0.135	0.2777	0.2438	0.204	0.1111
f1 Score, weighted	0.7204	0.8211	0.7868	0.7842	0.7111
fbeta Score, micro	0.7882	0.847	0.8352	0.8352	0.8
fbeta Score, macro	0.137	0.2982	0.2787	0.2166	0.1041
fbeta Score, weighted	0.6883	0.8107	0.7696	0.7623	0.6666
Jaccard Similarity	0.7882	0.847	0.8352	0.8352	0.8
Hamming Loss	0.2117	0.1529	0.1647	0.1647	0.1999
Zero One Loss	0.2117	0.1529	0.1647	0.1647	0.1999

Πίνακας 55: Σύγκριση classifiers, λεξικό inquirer, δεύτερη έκδοση, default παράμετροι εισόδου, δεδομένα από Facebook

Με την τεχνική του πολλαπλασιασμού των σκορ που δίνονται στις εγγραφές της βάσης, το λεξικό παρουσιάζει χειρότερα αποτελέσματα σε σχέση με την πρώτη έκδοση του λεξικού, αλλά συνεχίζει να παρουσιάζει πολύ καλά αποτελέσματα συγκρίσιμα με τα αποτελέσματα που έχουν εξαχθεί με τη χρησιμοποίηση του λεξικού του imdb.

Για τη συνέχιση αυτής της εργασίας θεωρώ ότι δεν είναι απαραίτητος ο πολλαπλασιασμός των σκορ των εγγραφών με κάποια τιμή, καθώς και η βαθμολόγηση των εγγραφών με τη μηδενική είναι μία ικανότητα την οποία πρέπει να έχουν οι classifiers. Το λεξικό του inquirer βρίσκει στο 94.6% των εγγραφών της βάσης έστω μία λέξη για να αξιολογήσει με οποιοδήποτε σκορ, συμπεριλαμβανομένου του 0. Δηλαδή το λεξικό δεν αναγνωρίζει καμία λέξη στις προτάσεις προς επεξεργασία, μόλις σε ποσοστό 5.4%.

Στο διάγραμμα που ακολουθεί παρουσιάζω τα σκορ των classifiers όταν δέχονται δεδομένα από το Facebook που έχουν αξιολογηθεί από το λεξικό του inquirer.



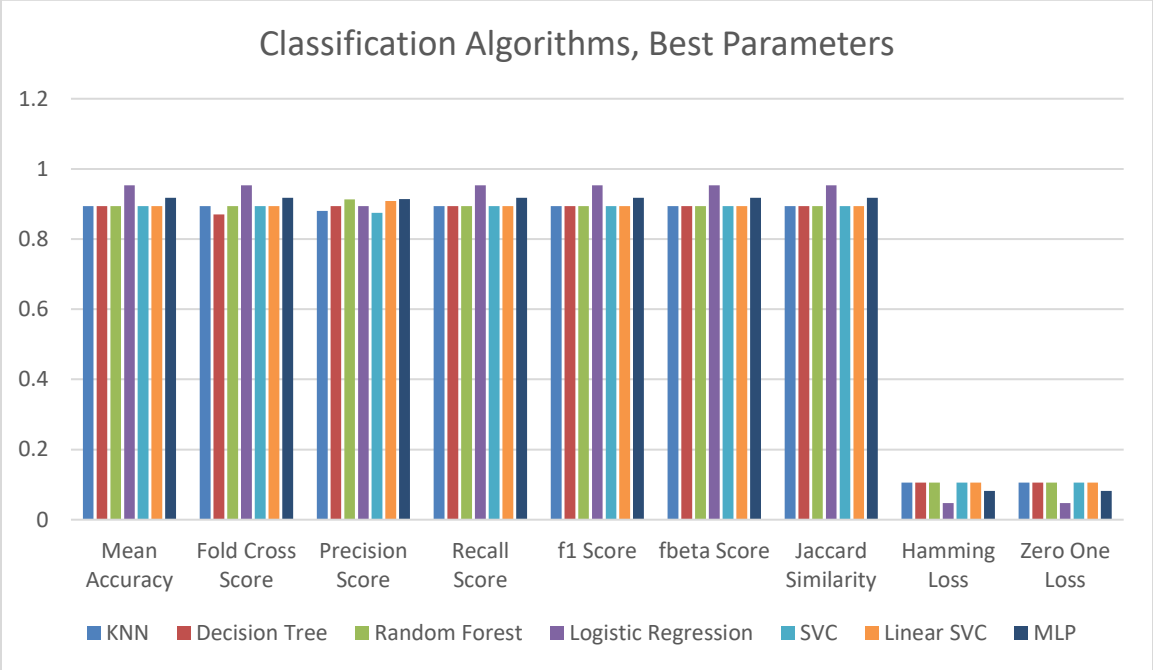
Διάγραμμα 57: Σύγκριση classifiers, λεξικό inquirer, default παράμετροι εισόδου, δεδομένα από Facebook

Στη συνέχεια πραγματοποιώ αναζήτηση βέλτιστων παραμέτρων για τους classifiers και εφόσον τα δεδομένα εισόδου δεχθούν προεπεξεργασία και παρουσιάζω τα αποτελέσματα.

	KNN	DT	RF	LR	SVC	L. SVC	MLP
Train Time(s)	0.001	0.0009	0.019	0.473	0.0209	0.0079	0.7946
Score Time(s)	0.003	0.001	0.0009	0.0	0.003	0.0	0.0015
Mean Accuracy	0.8941	0.8823	0.9058	0.9529	0.8941	0.8941	0.9294
Fold Cross Score	0.8801	0.8702	0.9133	0.8935	0.8745	0.9089	0.9141
Prec Score, micro	0.8941	0.8941	0.8941	0.9529	0.8941	0.8941	0.9176
Prec Score, macro	0.2235	0.2871	0.381	0.7375	0.2235	0.2235	0.4814
Prec Score, weighted	0.7994	0.8779	0.8756	0.9435	0.7994	0.7994	0.9102
Recall Score, micro	0.8941	0.8941	0.8941	0.9529	0.8941	0.8941	0.9176
Recall Score, macro	0.25	0.2778	0.3472	0.6428	0.25	0.25	0.3538
Recall Score, weighted	0.8941	0.8941	0.8941	0.9529	0.8941	0.8941	0.9176
f1 Score, micro	0.8941	0.8941	0.8941	0.9529	0.8941	0.8941	0.9176
f1 Score, macro	0.236	0.2819	0.3604	0.6754	0.236	0.236	0.3888
f1 Score, weighted	0.8441	0.8856	0.8833	0.9428	0.8441	0.8441	0.9036
fbeta Score, micro	0.8941	0.8941	0.8941	0.9529	0.8941	0.8941	0.9176
fbeta Score, macro	0.2283	0.2849	0.3716	0.7072	0.2283	0.2283	0.4317
fbeta Score, weighted	0.8167	0.8809	0.8782	0.9413	0.8167	0.8167	0.9032
Jaccard Similarity	0.8941	0.8941	0.8941	0.9529	0.8941	0.8941	0.9176
Hamming Loss	0.1058	0.1058	0.1058	0.047	0.1058	0.1058	0.0823
Zero One Loss	0.1058	0.1058	0.1058	0.047	0.1058	0.1058	0.0823

Πίνακας 56: Σύγκριση classifiers, λεξικό inquirer, βέλτιστοι παράμετροι εισόδου, δεδομένα από Twitter

Όταν βελτιστοποιούνται οι classifiers με τη χρήση των προτεινόμενων παραμέτρων εισόδου παρουσιάζουν κατά κανόνα καλύτερα αποτελέσματα, έτσι γίνεται και στην περίπτωση του λεξικού inquirer. Η οπτική απεικόνιση των classifiers φαίνεται στο διάγραμμα που ακολουθεί.



Διάγραμμα 58: Σύγκριση classifiers, λεξικό inquirer, βέλτιστοι παράμετροι εισόδου, δεδομένα από Facebook

Από του classifiers ξεχωρίζει ο Logistic Regression ο οποίος εμφανίζει πολύ υψηλά σκορ, 95% σε όλες τις μετρικές πλην του fold cross score, στην οποία εμφανίζει, το επίσης υψηλό, 89%. Τέλος, για ένα ακόμη λεξικό εξάγεται το συμπέρασμα ότι αν βελτιστοποιήσουμε τους classifiers, οι classifiers μεταξύ τους δεν παρουσιάζουν πολύ μεγάλη διαφορά στην απόδοση.

6.11.2 Αποτελέσματα στα δεδομένα που εξαχθεί από το Twitter

Έχοντας ολοκληρώσει την αξιολόγηση των classifiers όταν δέχονται δεδομένα από το Facebook, προχωράω στην μελέτη της συμπεριφοράς των classifiers όταν δέχονται δεδομένα από το Twitter. Προτού παρουσιάσω τα αποτελέσματα των classifiers πρέπει να πραγματοποιήσω μία μικρή εισαγωγή για τη γενικότερη συμπεριφορά του λεξικού στα δεδομένα που έχω συλλέξει από το Twitter. Η μέση τιμή του σκορ που δίνεται στα tweets

που έχω συλλέξει είναι -0.164930 με μικρή διακύμανση, μόλις 0.363005. Δίνονται 16 διαφορετικές τιμές στα tweets όταν αξιολογούνται από το inquirer, αλλά όταν πραγματοποιείται η στρογγυλοποίηση οι τιμές γίνονται συνολικά 5 [0, 1, -2, 5, -1].

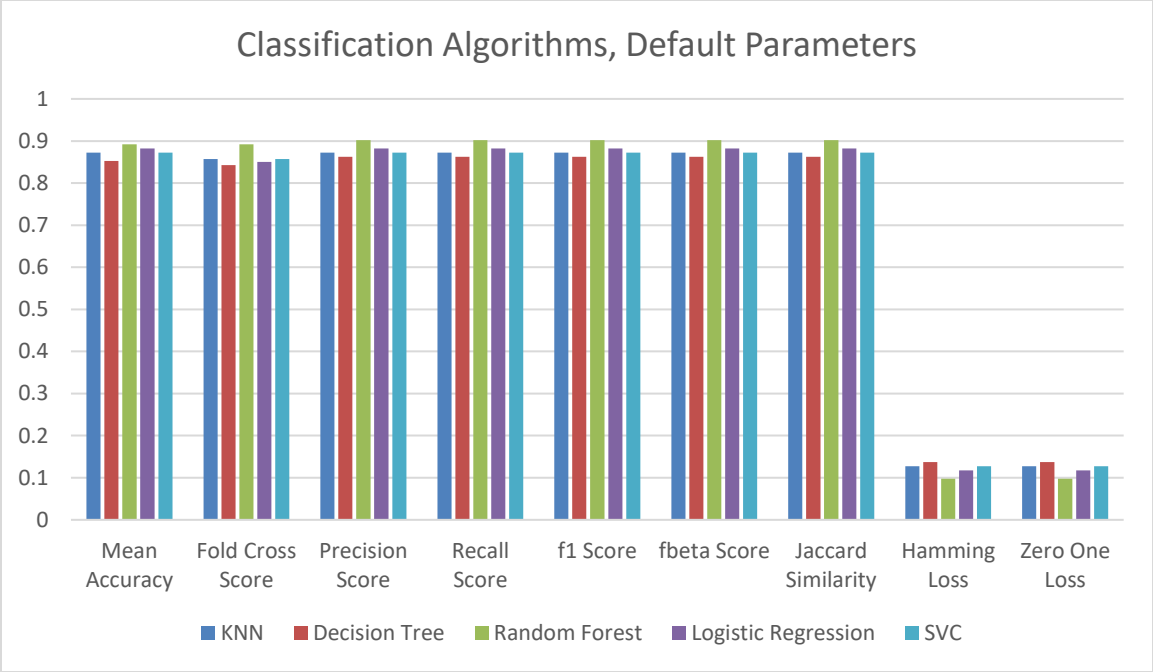
Έχοντας ως σύνολο αυτές τις 5 τιμές, το 86.9% των tweets βαθμολογούνται με μηδενική τιμή και το 8.9% με σκορ -2. Δηλαδή των 95.8 των συνολικών εγγραφών βαθμολογούνται με δύο πιθανές τιμές [0,-2].

Επειδή πολύ μεγάλο ποσοστό των tweets βαθμολογούνται με μηδέν, εξετάζω αν το μηδενικό σκορ δημιουργείται από βαθμολόγηση των επιμέρους λέξεων με μηδέν ή από μη ύπαρξη των λέξεων που χρησιμοποιούνται στα tweets στο λεξικό. Από τα tweets που έχω συλλέξει μόνο το 4.15% περιέχει λέξεις από τις οποίες ούτε μία δεν βρίσκεται στο λεξικό του inquirer.

	KNN	**DT**	**RF**	**LR**	**SVC**
Train Time(s)	0.0	0.0	0.016	0.0149	0.0
Score Time(s)	0.0	0.0	0.0	0.0	0.0
Mean Accuracy	0.8725	0.8529	0.8921	0.8823	0.8725
Fold Cross Score	0.8568	0.8428	0.8924	0.8501	0.8568
Prec Score, micro	0.8725	0.8627	0.9019	0.8823	0.8725
Prec Score, macro	0.2768	0.4369	0.6065	0.2227	0.2768
Prec Score, weighted	0.8231	0.8916	0.8901	0.7949	0.8231
Recall Score, micro	0.8725	0.8627	0.9019	0.8823	0.8725
Recall Score, macro	0.2774	0.5201	0.4322	0.2472	0.2774
Recall Score, weighted	0.8725	0.8627	0.9019	0.8823	0.8725
f1 Score, micro	0.8725	0.8627	0.9019	0.8823	0.8725
f1 Score, macro	0.2757	0.4634	0.4769	0.2343	0.2757
f1 Score, weighted	0.8466	0.8721	0.8905	0.8363	0.8466
fbeta Score, micro	0.8725	0.86275	0.9019	0.8823	0.8725
fbeta Score, macro	0.2759	0.4453	0.5330	0.2272	0.2759
fbeta Score, weighted	0.8322	0.883	0.8876	0.811	0.8322
Jaccard Similarity	0.8725	0.8627	0.9019	0.8823	0.8725
Hamming Loss	0.1274	0.1372	0.098	0.1176	0.1274
Zero One Loss	0.1274	0.1372	0.098	0.1176	0.1274

Πίνακας 57: Σύγκριση classifiers, λεξικό inquirer, default παράμετροι εισόδου, δεδομένα από Twitter

Τα αποτελέσματα είναι όμοια με αυτά που εξάχθηκαν όταν χρησιμοποιήθηκε ως δεδομένα δημοσιεύσεις από το Facebook. Η οπτική απεικόνιση της απόδοσης των classifiers απεικονίζεται στο διάγραμμα που ακολουθεί.



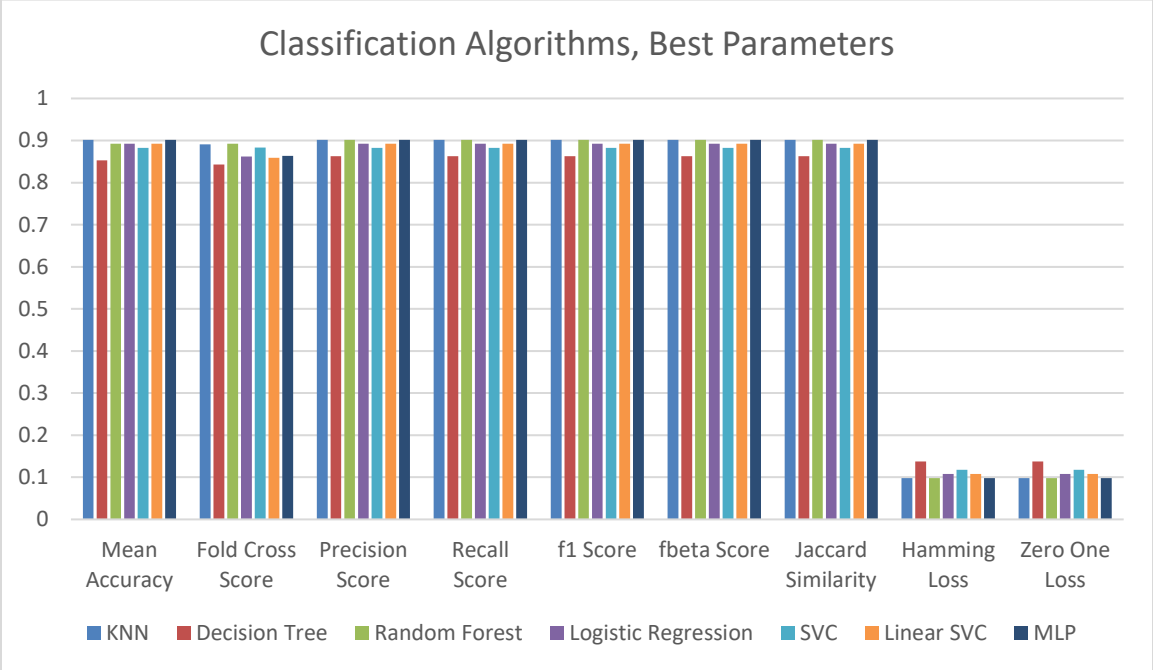
Διάγραμμα 59: Σύγκριση classifiers, λεξικό inquirer, default παράμετροι εισόδου, δεδομένα από Twitter

Ακολουθούν τα αποτελέσματα των classifiers όταν εκτελούνται με τις προτεινόμενες παραμέτρους από το Grid Search και τα δεδομένα εισόδου έχουν δεχθεί προεπεξεργασία.

	KNN	DT	RF	LR	SVC	L. SVC	MLP
Train Time(s)	0.0	0.0	0.016	0.036	0.016	0.0	0.9002
Score Time(s)	0.0	0.0	0.0	0.0	0.0	0.0	0.0013
Mean Accuracy	0.9019	0.8529	0.8921	0.8921	0.8823	0.8921	0.9019
Fold Cross Score	0.8906	0.8428	0.8924	0.8624	0.883	0.8588	0.8632
Prec Score, micro	0.9019	0.8627	0.9019	0.8921	0.8823	0.8921	0.9019
Prec Score, macro	0.6022	0.4369	0.6065	0.223	0.3268	0.223	0.4752
Prec Score, weighted	0.8747	0.8916	0.8901	0.7959	0.8368	0.7959	0.8724
Recall Score, micro	0.9019	0.8627	0.9019	0.8921	0.8823	0.8921	0.9019
Recall Score, macro	0.3663	0.5201	0.4322	0.25	0.3131	0.25	0.2857
Recall Score, weighted	0.9019	0.8627	0.9019	0.8921	0.8823	0.8921	0.9019
f1 Score, micro	0.9019	0.8627	0.9019	0.8921	0.8823	0.8921	0.9019
f1 Score, macro	0.4173	0.4634	0.4769	0.2357	0.3173	0.2357	0.2994
f1 Score, weighted	0.8751	0.8721	0.8905	0.8413	0.858	0.8413	0.8628
fbeta Score, micro	0.9019	0.8627	0.9019	0.8921	0.8823	0.8921	0.9019
fbeta Score, macro	0.4929	0.4453	0.533	0.2279	0.3222	0.2279	0.3434
fbeta Score, weighted	0.8682	0.883	0.8876	0.8134	0.8449	0.8134	0.8512
Jaccard Similarity	0.9019	0.8627	0.9019	0.8921	0.8823	0.8921	0.9019
Hamming Loss	0.0980	0.1372	0.098	0.1078	0.1176	0.1078	0.0980

Πίνακας 58: Σύγκριση classifiers, λεξικό inquirer, βέλτιστοι παράμετροι εισόδου, δεδομένα από Twitter

Οι classifiers παρουσιάζουν μια μικρή βελτίωση, εκτός των Decision Tree και Random Forest, με αποτέλεσμα όλοι οι classifiers να παρουσιάζουν παρόμοια αποτελέσματα. Ενδεικτικά στην μετρική του fold cross score η μεγαλύτερη διαφορά ανάμεσα σε δύο classifiers δεν ξεπερνάει το 0.05. Η οπτική απεικόνιση της απόδοσης τους φαίνεται παρακάτω.



Διάγραμμα 60: Σύγκριση classifiers, λεξικό inquirer, βέλτιστοι παράμετροι εισόδου

Σε ένα ακόμα λεξικό παρατηρείται ότι ο classifier που θα χρησιμοποιηθεί διαδραματίζει μικρό ρόλο, αφού όλοι οι classifiers εξάγουν αποτελέσματα με μικρές διαφορές. Τέλος, το λεξικό του inquirer παρουσιάζει μακράν τα καλύτερα αποτελέσματα σε σχέση με οποιοδήποτε άλλο λεξικό.

7. Επισκόπηση Αποτελεσμάτων

Αυτό το κεφάλαιο υπάρχει πριν το κεφάλαιο των συμπερασμάτων προκειμένου να γίνει μια περίληψη των όσον έχουν γραφτεί σε αυτή την εργασία. Η εργασία έχει ως θέμα την ικανότητα διαφορετικών λεξικών να θέσουν τη σωστή βαθμολογία σε προτάσεις που έχουν δημοσιευθεί από κοινωνικά δίκτυα και παράλληλα να εξετάσει ποιοι classifiers είναι πιο αποτελεσματικοί στην διεκπεραίωση της εργασίας της κατηγοριοποίησης των προτάσεων που έχουν συλλεχθεί.

Αρχικά δεν χρησιμοποίησα μία γνωστή δοκιμασμένη βάση για τη μελέτη της απόδοσης των classifiers, όπως γίνεται στις περισσότερες εργασίες, αλλά δημιούργησα τη δική μου βάση δεδομένων από δεδομένα που εξήγαγα από κοινωνικά δίκτυα. Έλαβα πληροφορίες από δύο κοινωνικά δίκτυα, Facebook και Twitter, χρησιμοποιώντας τα API που προσφέρουν. Η επικοινωνία με το API πραγματοποιήθηκε με τη χρήση γλώσσας Python, συγκεκριμένες πληροφορίες υλοποίησης δίνονται στο *κεφάλαιο 2 Λογισμικό Συλλογής Δεδομένων*. Τα δεδομένα που συνέλεξα είναι δημοσιεύσεις/tweets ανταγωνιστών τεχνολογικών εταιριών και είναι προσβάσιμα στο Διαδίκτυο ακόμα και από χρήστες χωρίς λογαριασμό στα συγκεκριμένα κοινωνικά δίκτυα.

Στη συνέχεια χρησιμοποίησα το εργαλεία *Stanford POSTagger* για τη γραμματική ανάλυση των δημοσιεύσεων που έχω συλλέξει. Το συγκεκριμένο εργαλείο είναι open-source, γραμμένο σε java και το εκτέλεσα για κάθε δημοσίευση που έχω συλλέξει. Με τα αποτελέσματα που έδωσε η εκτέλεση του συγκεκριμένου προγράμματος, ανανέωσα τη βάση δεδομένων.

Εφόσον έχω συλλέξει τα δεδομένα μου, το επόμενο βήμα είναι η οπτικοποίηση αυτών. Ο λόγος που προχωράω στην οπτικοποίηση δεδομένων είναι η πιθανή εύρεση μοτίβων μέσα από τη γραφική απεικόνιση. Μοτίβα τα οποία είναι αδύνατο να αναγνωριστούν μέσα από την ανάγνωση της βάσης δεδομένων. Εκτός από εύρεση πιθανών μοτίβων η οπτικοποίηση βοηθάει στην κατανόηση των δεδομένων, έτσι ώστε να γίνει μία πρώτη απόπειρα ερμηνείας των δεδομένων που έχω συλλέξει.

Τα τελευταία πεδία που πρόσθεσα στη βάση δεδομένων μου είναι η αξιολόγηση των δεδομένων μου από τα λεξικά που έχω χρησιμοποιήσει και έχω αναλύσει *στο κεφάλαιο 4 Χρήση Λεξικών*. Τα χαρακτηριστικά των λεξικών παρουσιάζονται στον πίνακα που ακολουθεί, *'subs'* ορίζω τις συνολικές εγγραφές του λεξικού, *'nz subs'* τις μη μηδενικές εγγραφές, *'range'* το εύρος τιμών που είναι δίνει το λεξικό, *'avg'* η μέση τιμή των εγγραφών και *'var'* η διακύμανση των τιμών.

	subs	**nz subs**	**range**	**avg**	**var**
AFINN	9,164	7,784	[-5,5]	0.5139	2.214
imdb	63,104	63,104	[1,10]	5.7732	1.0373

Amazon/TripAdvisor	9,686	9,686	[1,5]	3.3689	0.9831
Goodreads	10,050	10,050	[1,5]	3.0358	0.8307
Opentable	7,723	7,723	[1,5]	3.2526	0.9165
Opinion Observer	6,789	6789	[-1,1]	-0.409	0.8328
SentiWordNet	147,791	38,459	[-1,1]	-0.0136	0.0403
Subjectivity	8,222	7,629	[-2,2]	-0.498	2.6271
Inquirer	156584	9166	[-30,24]	-0.0081	0.3507

Πίνακας 59: Τα χαρακτηριστικά των λεξικών που χρησιμοποίησα

Η εκτίμηση του σκορ λεξικού από τα υπόλοιπα πεδία πραγματοποιείται μέσω συγκεκριμένων αλγορίθμων. Η εύρεση μοτίβων μέσα από μία βάση δεδομένων και η λήψη απόφασης για κατηγοριοποίηση των δεδομένων είναι μια διεργασία την οποία διεκπεραιώνουν κατά κύριο λόγω αλγόριθμοι classification, clustering και regression. Οι αλγόριθμοι του clustering ομαδοποιούν τα δεδομένα σύμφωνα με τα κοινά χαρακτηριστικά που εμφανίζουν, ενώ οι αλγόριθμοι του classification και regression αναγνωρίζουν σε ποια κατηγορία ανήκει μία εγγραφή. Στην παρούσα εργασία, ορίζω ως κατηγορίες τα σκορ που θέτει το λεξικό για την κάθε εγγραφή και δεν χρησιμοποιώ τεχνικές clustering ή regression, καθώς δεν μπορούν να προσφέρουν κάτι περισσότερο.

Χρησιμοποίησα εφτά (7) διαφορετικούς αλγορίθμους classification, τους οποίους αρχικά τους εκτέλεσα με τις default παραμέτρους εισόδου και έπειτα με τις προτεινόμενες παραμέτρους εισόδου από το Grid Search και εφόσον τα δεδομένα έχουν δεχθεί προεπεξεργασία. Ακόμη τα δεδομένα μου τα χώρισα σε δύο ξεχωριστές βάσεις δεδομένων, η πρώτη περιέχει τα δεδομένα που έχουν εξαχθεί από το Facebook και η δεύτερη δεδομένα που έχουν εξαχθεί από το Twitter.

Για την πρώτη περίπτωση όπου οι classifiers εκτελούνται με τις default παραμέτρους εισόδου, δεχόμενοι δεδομένα από το Facebook, τα σκορ που επιτυγχάνονται φαίνονται στον παρακάτω πίνακα, ως μετρική παραθέτω αυτή του fold cross score.

	KNN	**DT**	**RF**	**LR**	**SVC**	**L. SVC**	**MLP**
AFINN	0.3914	0.2989	0.4154	0.3898	0.4784	-	-
imdb	0.7847	0.7553	0.8361	0.8198	0.8067	-	-
Amazon/TripAdvisor	0.1329	0.1381	0.2122	0.1405	0.1763	-	-
Goodreads	0.1292	0.1627	0.1628	0.1816	0.1292	-	-
Opentable	0.1345	0.1113	0.1414	0.1835	0.1345	-	-
Opinion Observer	0.4014	0.4199	0.4642	0.4732	0.4566	-	-
SentiWordNet	0.4409	0.3588	0.5271	0.494	0.4602	-	-
Subjectivity	0.3159	0.2487	0.3763	0.3465	0.3957	-	-
Inquirer	0.8478	0.8236	0.9133	0.8779	0.8745	-	-

Πίνακας 60: Απόδοση classifiers σε διαφορετικά λεξικά, δεδομένα Facebook, default παράμετροι εισόδου

Στον πίνακα με τις αποδόσεις των classifiers όταν εκτελούνται με τις default παραμέτρους, δεν σημείωσα την απόδοση των Linear SVC και MLP εξαιτίας της αστάθειας που παρουσιάζουν όταν εκτελούνται χωρίς τα δεδομένα εισόδου να έχουν δεχθεί προεπεξεργασία. Όσον αφορά την απόδοση των υπολοίπων classifiers, οι Random Forest, Logistic Regression και SVC ξεχωρίζουν ελαφρώς έναντι των KNN και Decision Tree.

Ο δεύτερος πίνακα που παραθέτω στο κεφάλαιο παρουσιάζει τα αποτελέσματα των classifiers όταν εκτελούνται με default παραμέτρους και τα δεδομένα έχουν δεχθεί προεπεξεργασία.

	KNN	DT	RF	LR	SVC	L. SVC	MLP
AFINN	0.4651	0.4541	0.4784	0.4784	0.4784	0.4323	0.4784
imdb	0.8227	0.8016	0.8324	0.8450	0.8067	0.8542	0.7566
Amazon/TripAdvisor	0.2077	0.1754	0.164	0.2167	0.2151	0.2223	0.1784
Goodreads	0.2004	0.1627	0.1628	0.222	0.2096	0.1812	0.2237
Opentable	0.1727	0.1113	0.1414	0.2067	0.1766	0.1652	0.1766
Opinion Observer	0.451	0.4594	0.4642	0.4947	0.4717	0.4987	0.501
SentiWordNet	0.647	0.6109	0.6407	0.647	0.647	0.647	0.647
Subjectivity	0.4176	0.3837	0.4001	0.4019	0.4019	0.4019	0.4019
Inquirer	0.7936	0.7848	0.8392	0.8097	0.8218	0.7936	0.7848

Πίνακας 61: Απόδοση classifiers σε διαφορετικά λεξικά, δεδομένα Facebook, βέλτιστοι παράμετροι εισόδου

Όταν οι classifiers εκτελούνται με τις βέλτιστες παραμέτρους και εφόσον τα δεδομένα εισόδου έχουν δεχθεί προεπεξεργασία, οι classifiers παρουσιάζουν πανομοιότυπη απόδοση. Χαρακτηριστικά, κανένας classifier δεν παρουσιάζει τα καλύτερα αποτελέσματα σε περισσότερα από τρία διαφορετικά λεξικά.

Αντίστοιχα όταν οι classifiers καλούνται να αξιολογήσουν δεδομένα τα οποία έχουν εξαχθεί από το Twitter, τείνουν να παρουσιάζουν καλύτερα αποτελέσματα. Παρακάτω παρουσιάζω τις αποδόσεις των classifiers όταν εκτελούνται με τις default παραμέτρους εισόδου και τα λεξικά αξιολογούν δεδομένα από το Twitter.

	KNN	DT	RF	LR	SVC	L. SVC	MLP
AFINN	0.5914	0.4924	0.6432	0.6000	0.6123	-	-
imdb	0.7633	0.7997	0.8081	0.731	0.7702	-	-
Amazon/TripAdvisor	0.1867	0.3065	0.3157	0.2641	0.2812	-	-
Goodreads	0.1789	0.2732	0.3088	0.2915	0.2499	-	-
Opentable	0.1907	0.2721	0.3437	0.2649	0.2554	-	-
Opinion Observer	0.5737	0.4887	0.6116	0.5856	0.6169	-	-
SentiWordNet	0.7388	0.6546	0.7291	0.7381	0.7517	-	-
Subjectivity	0.7479	0.808	0.8121	0.7551	0.7689	-	-
Inquirer	0.8568	0.8428	0.8924	0.8501	0.8568	-	-

Πίνακας 62: Απόδοση classifiers σε διαφορετικά λεξικά, δεδομένα Twitter, default παράμετροι εισόδου

Οι αλγόριθμοι όταν καλούνται να κατηγοριοποιήσουν δεδομένα τα οποία έχουν εξαχθεί από το Twitter τείνουν να παρουσιάζουν καλύτερά σκορ. Ο κύριος λόγος που παρατηρείται αυτό το φαινόμενο είναι ότι στα λεξικά υπάρχει μεγαλύτερη συγκέντρωση στη μηδενική τιμή, οπότε οι classifiers έχουν ευκολότερο έργο να φέρουν εις πέρας. Η αιτία πίσω από αυτό το φαινόμενο χρήζει περαιτέρω έρευνας περισσότερο στον τομέα της επεξεργασίας φυσικής γλώσσας και λιγότερο πάνω στον τομέα της μηχανικής μάθησης.

Όταν οι classifiers εκτελούνται με τις βέλτιστες παραμέτρους παρουσιάζουν και καλύτερα αποτελέσματα, τα οποία παραθέτω στον παρακάτω πίνακα.

	KNN	DT	RF	LR	SVC	L. SVC	MLP

AFINN	0.6786	0.5816	0.6740	0.6123	0.6740	0.6123	0.6123
imdb	0.7633	0.7997	0.8081	0.731	0.8083	0.7702	0.7702
Amazon/TripAdvisor	0.3572	0.3065	0.3157	0.2786	0.3291	0.2752	0.3420
Goodreads	0.2004	0.1627	0.1628	0.222	0.2096	0.1812	0.2237
Opentable	0.3478	0.2721	0.3437	0.2648	0.3027	0.2717	0.3131
Opinion Observer	0.6609	0.5703	0.6075	0.6169	0.6346	0.6078	0.6078
SentiWordNet	0.7784	0.7436	0.7439	0.7517	0.7784	0.7517	0.7517
Subjectivity	0.825	0.808	0.8121	0.7608	0.7815	0.7641	0.795
Inquirer	0.8906	0.8428	0.8924	0.8624	0.883	0.8588	0.8632

Πίνακας 63: Απόδοση classifiers σε διαφορετικά λεξικά, δεδομένα Twitter, βέλτιστοι παράμετροι εισόδου

Όπως στα δεδομένα του Facebook, έτσι και στα δεδομένα του Twitter οι classifiers δείχνουν βελτίωση όταν εκτελούνται με τις κατάλληλες παραμέτρους και τα δεδομένα εισόδου δεχθούν προεπεξεργασία. Οι classifiers παρουσιάζουν πανομοιότυπα αποτελέσματα όταν εκτελούνται, με τον classifier του KNN να υπερτερεί ελαφρώς.

Το τελευταίο σημείο στο οποίο εστιάζω σε αυτό το κεφάλαιο είναι η αναζήτηση βέλτιστων παραμέτρων. Για την αναζήτηση βέλτιστων παραμέτρων χρησιμοποίησα δύο διαφορετικές τεχνικές, την Τυχαία Αναζήτηση (Random Search) και την εξαντλητική αναζήτηση (Grid Search). Στους αλγορίθμους των Random Forest και MLP πραγματοποίησα μόνο Random Search, εξαιτίας των πάρα πολλών δυνατών συνδυασμών που μπορούν να πραγματοποιηθούν.

Επίσης ο αλγόριθμος του SVC διαθέτει δύο επιπλέον επιλογές για την μεταβλητή του kernel, linear (γραμμική) και poly (πολυωνυμική). Ο λόγος που δεν εξέτασα επιπλέον αυτές τις δύο επιλογές είναι ο πολύ μεγάλος χρόνος που απαίτησαν για την ολοκλήρωση τους. Ο SVC με linear kernel έχει χρόνο εκπαίδευσης ίσο με 71.95 λεπτά όταν ο kernel rbf χρειάζεται 0.0149 δευτερόλεπτα και ο sigmoid εκτελείται γρηγορότερα από χιλιοστό (0.001) του δευτερόλεπτου. Ο kernel poly έχει άγνωστο χρόνο εκτέλεσης, αφού μέσα σε 120 ώρες αναμονής δεν εκτελέστηκε.

Όσον αφορά την αποτελεσματικότητα του Random Search, δηλαδή πόσο καλοί είναι οι παράμετροι που προτείνονται σε σχέση με αυτούς που προτείνονται από το Grid Search, τα ευρήματα είναι ενθαρρυντικά για την χρησιμοποίηση του Random Search. Χωρίς να έχω κρατήσει λεπτομερές αρχείο με τις αποδόσεις των classifiers ανάλογα με τις παραμέτρους εισόδου με τις οποίες εκτελούνται, η προσωπική μου εκτίμηση είναι ότι η εκτέλεση του Random Search με χρήση του 10% των συνολικών δυνατών συνδυασμών είναι αρκετή. Αν η βελτίωση των classifier είναι πολύ σημαντική και δεν τίθεται θέμα χρόνου, τότε η επιλογή του Grid Search είναι αυτή που συνιστάται.

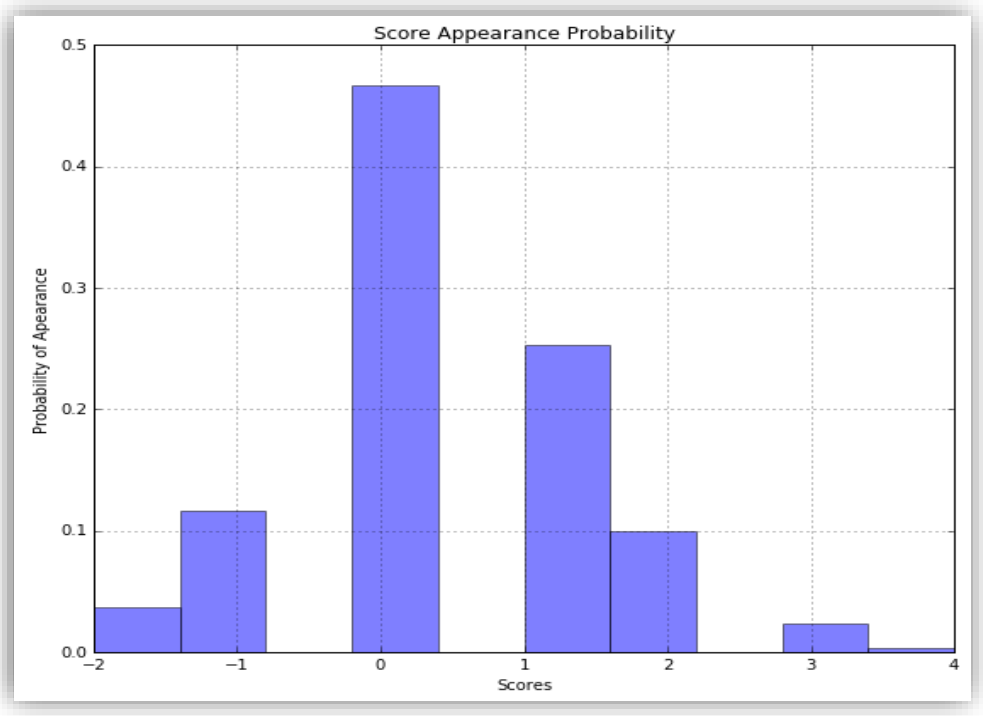
8. Συμπεράσματα και Περαιτέρω Εργασία

Στον επίλογο της εργασίας δεν αναφέρω ξανά τη δομή της εργασίας και τα αποτελέσματα που έχω εξάγει, γιατί αυτή η ανάλυση πραγματοποιείται στο *κεφάλαιο 7 Επισκόπηση Αποτελεσμάτων*. Σε αυτό το κεφάλαιο θα επικεντρωθώ στην ερμηνεία της απόδοσης των λεξικών και θα προτείνω άξονες για την περαιτέρω έρευνα πάνω στο κομμάτι του Sentiment Analysis πάνω σε δεδομένα που έχουν εξαχθεί από microblogging.

Η αρχική ιδέα πάνω στην οποία δημιουργήθηκε αυτή η διπλωματική εργασία ήταν μελέτη των συναισθημάτων που προκαλούν διαφορετικές δημοσιεύσεις στους χρήστες των κοινωνικών δικτύων. Οι πρόσθετες αντιδράσεις που έχει προσθέσει το Facebook δημιούργησαν, θεωρητικά, την τέλεια βάση δεδομένων. Δεκάδες χιλιάδες χρήστες αξιολογούν διαφορετικές προτάσεις, ακόμη και η μη αντίδραση τους προσφέρει δεδομένα προς ανάλυση.

Στην πράξη όμως οι αντιδράσεις δεν ήταν αυτές που περίμενα. Οι χρήστες δεν είχαν εξοικειωθεί με τις επιπλέον αντιδράσεις με αποτέλεσμα να μην τις χρησιμοποιούν, η αντίδραση του ‘like’ ήταν η επικρατέστερη στο συντριπτικό ποσοστό του 99%.

Μετά από αυτή την παρατήρηση η προσέγγιση σε αυτό το πρόβλημα άλλαξε, δεν μπορούσα να αντιστοιχίσω έξι (6) διαφορετικές βαθμολογίες σε έξι (6) διαφορετικά συναισθήματα. Αντίθετα το πρόβλημα έγινε σχεδόν δυαδικό, είτε η δημοσίευση έχει ανταπόκριση στους χρήστες του κοινωνικού δικτύου είτε όχι. Ακόμη ο τρόπος με τον οποίο αξιολογούσαν τις προτάσεις προς ανάλυση τα λεξικά έδειχνε προς αυτή την κατεύθυνση. Όσα λεξικά παρουσίασαν καλά ή ανεκτά αποτελέσματα είχαν υψηλή συγκέντρωση σε μία συγκεκριμένη τιμή. Για να γίνω καλύτερα κατανοητός παραθέτω την πιθανότητα εμφάνισης των διαφορετικών σκορ που αποδίδονται στα δεδομένα που έχω εξάγει από το Facebook και έχουν αξιολογηθεί από το λεξικό του SentiWordNet. Προσοχή δεν παραθέτω τις πιθανότητες βαθμολογίες της κάθε λέξης ξεχωριστά που υπάρχει στο λεξικό, όπως πραγματοποίησα στο *κεφάλαιο 4 Χρήση Λεξικών*, αλλά τη πιθανότητα εμφάνισης σκορ σε όλη την πρόταση.



Διάγραμμα 61: Πιθανότητα εμφάνισης σκορ σε προτάσεις που έχουν εξαχθεί από το Facebook και αξιολογούνται από το λεξικό του SentiWordNet

Το λεξικό του SentiWordNet παρουσιάζει τα καλύτερα αποτελέσματα αν συνυπολογιστεί το γεγονός ότι δεν υπάρχει κάποια τιμή που να συγκεντρώνει πάνω από το 80% των συνολικών αξιολογήσεων. Τα λεξικά των imdb και inquirer τα οποία εμφανίζουν τα καλύτερα αποτελέσματα συγκεντρώνουν τουλάχιστον 75% των συνολικών αξιολογήσεων σε συγκεκριμένη τιμή, την μηδενική (0). Αυτή η συγκέντρωση για το λεξικό του imdb υποδηλώνει το ποσοστό των προτάσεων στο οποίο δεν αναγνωρίζει ούτε μία λέξη. Στο λεξικό του SentiWordNet το ποσοστό των προτάσεων στις οποίες δεν αναγνωρίζει ούτε μία λέξη είναι μόλις 5.6%.

Ως γενική κριτική στα λεξικά που έχω μελετήσει τα χωρίζω σε 3 κατηγορίες, η πρώτη ανάγει την κατηγοριοποίηση σε δυαδικό πρόβλημα (imdb, inquirer, Subjectivity στα δεδομένα του Twitter) με μία συγκεκριμένη τιμή να έχει πολύ μεγάλη εμφάνιση. Τα λεξικά που έχουν υψηλή συγκέντρωση στις τιμές από την μηδενική τιμή μέχρι το μέσο όρο του σκορ που υπάρχουν στις λέξεις του λεξικού (Amazon/TripAdvisor, Goodreads, Opentable) και εμφανίζουν πολύ χαμηλά αποτελέσματα. Η τρίτη κατηγορία λεξικών αξιολογεί τα δεδομένα προς ανάλυση με τέτοιο τρόπο ώστε η βαθμολογία των προτάσεων να θυμίζει κανονική κατανομή. Η κανονική κατανομή για την μελέτη του Sentiment Analysis είναι η πιο αντιπροσωπευτική για τα πραγματικά συναισθήματα που εκφράζουν οι άνθρωποι. Η μέση τιμή της κατανομής δηλώνει αδιαφορία και στις τιμές που βρίσκονται δεξιά ή αρνητικά από αυτήν εκφράζονται θετικά ή αρνητικά συναισθήματα.

Ένα σημείο το οποίο είναι επίσης θέμα έρευνας είναι η σημασία της γραμματικής ανάλυσης στα δεδομένα που έχω συλλέξει. Για τα λεξικά τα οποία έχουν ανάγει το

πρόβλημα σε δυαδικό και συγκεντρώνουν το 80% των σκορ των αξιολογήσεων είναι φυσικό επόμενο τα συγκεκριμένα πεδία στη βάση δεδομένων να μην διαδραματίζουν σημαντικό ρόλο στην πρόβλεψη των classifiers. Η απόδοση τεσσάρων (4) classifiers όταν καλούνται να κατηγοριοποιήσουν δεδομένα τα οποία έχουν εξαχθεί από το Facebook και έχουν αξιολογηθεί από το λεξικό του inquirer φαίνεται στον πίνακα που ακολουθεί.

	KNN	**DT**	**RF**	**LR**	**SVC**
POS	0.8478	0.8236	0.9133	0.8779	0.8745
w/o POS	0.8478	0.8588	0.8820	0.8731	0.8745

Πίνακας 64: Απόδοση classifiers, συμπεριλαμβανομένων ή όχι των μερών του λόγου από τα οποία αποτελούνται τα δεδομένα του Facebook, αξιολόγηση από το λεξικό του inquirer

Από τους classifiers που παρέθεσα κανένας δεν επηρεάστηκε αισθητά. Οι αλγόριθμοι των Decision Tree και Random Forest παρουσιάζουν αστάθεια στις μετρικές, οπότε μπορεί να εξαχθεί το συμπέρασμα ότι σε αυτή την περίπτωση των λεξικών η γραμματική ανάλυση δεν διαδραματίζει σημαντικό ρόλο.

Στη συνέχεια παρουσιάζω την επιρροή που έχουν τα μέρη του λόγου στους classifiers στα δεδομένα του Facebook όταν αξιολογούνται από το λεξικό του Twitter. Εξετάζω τέσσερις (4) διαφορετικές περιπτώσεις, με default παραμέτρους εισόδου και μέρη του λόγου, με τις προτεινόμενες παραμέτρους από το Grid Search και μέρη του λόγου, με default παραμέτρους εισόδου χωρίς τα μέρη του λόγου και βέλτιστους παραμέτρους χωρίς τα μέρη του λόγου.

	KNN	**DT**	**RF**	**LR**	**SVC**
POS	0.5706	0.5031	0.5964	0.6022	0.647
w/o POS	0.5754	0.4342	0.5483	0.6167	0.647
POS Grid	0.647	0.6202	0.6202	0.647	0.647
w/o POS Grid	0.647	0.5957	0.6462	0.647	0.647

Πίνακας 65: Απόδοση classifiers, συμπεριλαμβανομένων ή όχι των μερών του λόγου από τα οποία αποτελούνται τα δεδομένα του Facebook, αξιολόγηση από το λεξικό του SentiWordNet

Το πρώτο πράγμα που πρέπει να σημειωθεί από την παρατήρηση των σκορ των classifiers είναι ότι οι classifiers δέχονται βελτίωση, αν εκτελεστούν με τις κατάλληλες παραμέτρους. Όταν εκτελούνται με τις default παραμέτρους εισόδου η χρησιμοποίηση ή μη των μερών του λόγου επηρεάζει 3 από τους 5 classifiers, αντίθετα όταν οι classifiers εκτελούνται με τις προτεινόμενες παραμέτρους από το Grid Search, η χρησιμοποίηση των μερών του λόγου αποκτά ακόμα μικρότερη σημασία. Οι classifiers των Decision Tree και Random Forest παρουσιάζουν αστάθεια, οπότε η διαφορά του ±2% δεν διαδραματίζει σημαντικό ρόλο.

Η βαρύτητα των μερών του λόγου για την απόδοση των classifiers δεν είναι ξεκάθαρη, αφού στον classifier του Logistic Regression η απουσία τους απέδωσε ελαφρώς καλύτερα αποτελέσματα, ενώ στους Decision Tree και Random Forest χειρότερα. Ακόμη όταν οι classifiers εκτελούνται με βέλτιστες παραμέτρους, η χρήση ή μη των μερών του λόγου αποκτά ακόμα μικρότερη σημασία.

Κατά τη διάρκεια της συγγραφής της εργασίας μου δεν κράτησα λεπτομερή καταγραφή για την απόδοση των classifiers όταν αφαιρώ τα μέρη του λόγου που

αποτελούν την πρόταση, οπότε περαιτέρω διερεύνηση πάνω στη βαρύτητα των μερών του λόγου είναι απαραίτητη.

Η μελλοντική επέκταση και φυσική συνέχεια της εργασίας είναι η δημιουργία ενός λεξικού σχεδιασμένου για την αξιολόγηση δημοσιεύσεων στα κοινωνικά δίκτυα από εταιρείες στον τεχνολογικό τομέα. Για να δημιουργηθεί το συγκεκριμένο λεξικό πρέπει να συλλεχθούν ξανά δεδομένα και να γίνει μελετηθεί ποιες συγκεκριμένες λέξεις είναι αυτές που προκαλούν τις αντιδράσεις στους χρήστες. Είναι απαραίτητο στην μελέτη του αντικειμένου να συμπεριληφθούν τα #hashtags που χρησιμοποιούν οι εταιρείες και η επιρροή τους στο κοινό. Ως βάση δημιουργίας των λεξικών προτείνω τα τρία λεξικά των οποίων η αξιολόγηση στα δεδομένα της βάσης θυμίζουν κανονική κατανομή(AFINN, SentiWordNet, Opinion Observer).

Για τη δημιουργία λεξικού υπάρχει βέβαια και η ενναλακτική επιλογή της δημιουργίας λεξικού που αντιμετωπίζει τα δεδομένα προς επεξεργασία ως δυαδικό πρόβλημα, ανταπόκριση από τους χρήστες ή όχι. Η δημιουργία αυτού του λεξικού είναι πιο εύκολη και για την κατασκευή του απαιτούνται λιγότερα βήματα: συλλογή δεδομένων, μελέτη λέξεων που έχουν ανταπόκριση στις χρήστες.

9. Λίστα Διαγραμμάτων

10. Λίστα Πινάκων

11. Λίστα Κωδίκων

12. Λίστα Εικόνων

References

[1] B. Liu, «Sentiment Analysis and Subjectivity,» σε *Handbook of Natural Language Processing, Second Edition*, N. I. a. F. J. Damerau, Επιμ., Boca Raton, FL, CRC Press, Taylor and Francis Group, 2010, pp. 627-666.

[2] B. Pang και L. Lee, «Opinion Mining and Sentiment Analysis,» *Foundations and Trends® in Information Retrieval,* τόμ. 2, pp. 1-135, 2008.

[3] E. Kouloumpis, T. Wilson και J. Moore, «Twitter Sentiment Analysis: The Good the Bad and the OMG!,» σε *Proceedings of the Fifth International AAAI Conference on Weblogs and Social Meda*, Barcelona, 2011.

[4] A. Pak και P. Paroubek, «Twitter as a Corpus for Sentiment Analysis and Opinion Mining,» σε *Proceedings of the International Conference on Language Resources and Evaluation*, Valleta, Malta, 2010.

[5] K.-L. Liu, W.-J. Li και M. Guo, «Emoticon Smoothed Language Models for Twitter Sentiment Analysis,» σε *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, Toronto,Ontario, Canada, 2012.

[6] B. Pang, L. Lee και S. Vaithyanathan, «Thumbs Up?: Sentiment Classification Using Machine Learning Techniques,» σε *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing - Volume 10*, Association for Computational Linguistics, 2002, pp. 79--86.

[7] «https://twitter.com/,» [Ηλεκτρονικό].

[8] H. Kwak, C. Lee, H. Park και S. Moon, «What is Twitter, a Social Network or a News Media?,» σε *Proceedings of the 19th International Conference on World Wide Web*, Raleigh, North Carolina, USA, ACM, 2010, pp. 591--600.

[9] «https://www.facebook.com/,» [Ηλεκτρονικό].

[10] A. Agarwal, B. Xie, I. Vovsha, O. Rambow και R. Passonneau, «Sentiment Analysis of Twitter Data,» σε *Proceedings of thw Workshop on Language in Social Media*, Portland, Oregon, 2011.

[11] A. Go, R. Bhayani και L. Huang, «Twitter Sentiment Classification using Distant Supervision,» *Processing,* pp. 1-6, 2009.

[12] A. Ortigosa, J. M. Martín και R. M. Carro, «Sentiment analysis in Facebook and its application to e-learning,» *Computers in Human Behavior,* τόμ. 31, pp. 527-541, February 2014.

[13] V. Hatzivassiloglou και K. R. McKeown, «Predicting the semantic orientation of adjectives,» σε *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics*, Madrid, Spain, Association for Computational Linguistics, 1997, pp. 174--181.

[14] https://www.tumblr.com/. [Ηλεκτρονικό].

[15] «https://plus.google.com,» [Ηλεκτρονικό].

[16] R. Feldman, «Techniques and Applications for Sentiment Analysis,» *Commun. ACM,* τόμ. 56, pp. 82--89, April 2013.

[17] H. Wang, D. Can, A. Kazemzadeh, F. Bar και S. Narayanan, «A System for Real-time Twitter Sentiment Analysis of 2012 U.S. Presidential Election Cycle,» σε *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, Jeju, Republic of Korea, 2012.

[18] «https://scrapy.org/,» Open Source. [Ηλεκτρονικό]. [Πρόσβαση December 2016].

[19] «http://nutch.apache.org/,» Apache. [Ηλεκτρονικό]. [Πρόσβαση December 2016].

[20] P. Jack και N. Levitt, «https://webarchive.jira.com/wiki/display/Heritrix,» Apache, January 2014. [Ηλεκτρονικό].

[21] «https://developers.facebook.com/,» Facebook, 2016. [Ηλεκτρονικό].

[22] «https://dev.twitter.com/,» Twitter, Inc., 2016. [Ηλεκτρονικό].

[23] «https://docs.python.org/2/library/urllib2.html,» Python Software Foundation, 20 September 2016. [Ηλεκτρονικό].

[24] «https://docs.python.org/2/library/datetime.html,» [Ηλεκτρονικό].

[25] M. Rodrigues, «https://github.com/PyMySQL,» November 2016. [Ηλεκτρονικό].

[26] «https://www.apachefriends.org/index.html,» [Ηλεκτρονικό].

[27] K. Toutanova, D. Klein, C. Manning και Y. Singer, «Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network,» σε *HLT-NAACL* , Edmonton, Canada , 2003.

[28] «https://docs.python.org/2/library/re.html,» [Ηλεκτρονικό].

[29] «https://continuum.io,» Anaconda Software Distribution, November 2016. [Ηλεκτρονικό].

[30] G. van Rossum και F. L. Drake, «Python Language Reference, Python 2.7.13 documentation,» Python Software Foundation, 11 February 2017. [Ηλεκτρονικό]. Available: https://docs.python.org/2/. [Πρόσβαση 16 February 2017].

[31] G. v. Rossum, «Python tutorial, Technical Report CS-R9526,» σε *Centrum voor Wiskunde en Informatica (CWI)*, Amsterdam, 1995.

[32] W. McKinney, «pandas: a Foundational Python Library for Data,» σε *Python for High Performance Computing*, Seatle, 2011.

[33] J. D. Hunter, «Matplotlib: A 2D graphics environment,» *Computing In Science & Engineering,* pp. 90-95, 2007.

[34] F. {. Nielsen, «AFINN,» σε *ESWC2011 Workshop on 'Making Sense of Microposts'*, 2011.

[35] L. K. Hansen, A. Arvidsson, F. {. Nielsen και E. Colleoni, «Good Friends, Bad News - Affect and Virality in,» σε *The 2011 International Workshop on Social Computing,*, Crete, Greece, 2011.

[36] C. Potts, «http://compprag.christopherpotts.net,» 2011. [Ηλεκτρονικό]. Available: http://compprag.christopherpotts.net/reviews.html#MTURK. [Πρόσβαση Feburary 2017].

[37] C. Potts, «On the negativity of negation,» *Semantics and Linguistic Theory,* τόμ. 20, pp. 636-659, 2010.

[38] [Ηλεκτρονικό]. Available: http://www.imdb.com.

[39] C. Potts, «Developing adjective scales from user-supplied textual metadata,» σε *A Workshop on Restructuring Adjectives in WordNet*, Arlington, VA, 2011.

[40] [Ηλεκτρονικό]. Available: https://www.amazon.com/.

[41] [Ηλεκτρονικό]. Available: https://www.tripadvisor.com.gr/.

[42] [Ηλεκτρονικό]. Available: http://web.stanford.edu/~cgpotts/data/wordnetscales/wn-asr-multicorpus.csv.zip.

[43] [Ηλεκτρονικό]. Available: https://www.goodreads.com/.

[44] [Ηλεκτρονικό]. Available: https://www.opentable.com/.

[45] M. Hu και B. Liu, «Mining and Summarizing Customer Reviews,» σε *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Seattle, WA, USA, ACM, 2004, pp. 168-177.

[46] B. Liu, M. Hu και J. Cheng, «Opinion Observer: Analyzing and Comparing Opinions on the Web,» σε *Proceedings of the 14th International Conference on World Wide Web*, Chiba, Japan, ACM, 2005, pp. 342-351.

[47] G. A. Miller, «WordNet: A Lexical Database for English,» *Communications of the ACM,* τόμ. 38, pp. 39-41, 1995.

[48] C. Fellbaum και G. Miller, WordNet: An Electronic Lexical Database, Cambridge: MIT Press, 1998.

[49] S. Baccianella, A. Esuli και F. Sebastiani, «SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining,» σε *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, N. Calzolari, K. Choukri, B. Maegaard, J. Mariani, J. Odijk, S. Piperidis, M. Rosner και D. Tapias, Επιμ., Valletta, Malta, European Language Resources Association (ELRA), 2010.

[50] [Ηλεκτρονικό]. Available: http://sentiwordnet.isti.cnr.it/SentiWordNet_3.0.0.tgz.

[51] J. Wiebe, T. Wilson και C. Cardie, «Annotating Expressions of Opinions and Emotions in Language,» *Language Resources and Evaluation,* τόμ. 39, pp. 165-210, 2005.

[52] [Ηλεκτρονικό]. Available: http://compprag.christopherpotts.net/code-data/wnscores.py.

[53] C. Potts, «Linguist 287 / CS 424P: Extracting Social Meaning and Sentiment, Stanford, Fall 2010,» 21 September 2010. [Ηλεκτρονικό]. Available: http://web.stanford.edu/class/cs424p/materials/ling287-handout-09-21-lexicons.pdf. [Πρόσβαση 16 February 2017].

[54] S. Blair-Goldensohn, K. Hannan, R. McDonald, T. Neylon, G. A. Reis και J. Reynar, «Building a Sentiment Summarizer for Local Service Reviews,» σε *{WWW} Workshop on {NLP} in the Information Explosion Era (NLPIX)*, Beijing, China, 2008.

[55] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot και E. Duchesnay, «Scikit-learn: Machine Learning in {P}ython,» *Journal of Machine Learning Research,* τόμ. 12, pp. 2825-2830, 2011.

[56] E. Jones, T. Oliphant, P. Petterson και e. al., «{SciPy}: Open source scientific tools for {Python},» 2001-. [Ηλεκτρονικό]. Available: http://www.scipy.org/. [Πρόσβαση 16 Februaray 2017].

[57] S. Rostrup και S. Benthall, σε *Proceedings of the 15th Python in Science Conference (SciPy 2016)*, Austin, Texas, 2016.

[58] S. van der Walt, S. C. Colber και V. Gael, «The NumPy Array: A Structure for Efficient Numerical Computation,» *Computing in Science & Engineering,* τόμ. 13, αρ. 2, pp. 22-30, March-April 2011.

[59] J. J.D., «Matplotlib: A 2D Graphics Environment,» *Computing in Science & Engineering,* τόμ. 9, αρ. 3, pp. 90-95, May 2007.

[60] F. Elibe, M. A. Hall και W. I. H., The WEKA Workbench. Online Appendix for "Data Mining: Practical Machine Learning Tools and Techniques", 4th επιμ., Morgan Kaufmann, 2016.

[61] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann και I. H. Witten, «The WEKA Data Mining Software: An Update,» *SIGKDD Explorations,* τόμ. 11, αρ. 1, 2009.

[62] L. Buitinch, G. Loupper, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. V. J. Layton, A. Joly, B. Holt και V. Gael, «{API} design for machine learning software: experiences from the scikit-learn project,» σε *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, 2013, pp. 108-122.

[63] D. T. Larose, «k-Nearest Neighbor Algorithm,» σε *Discovering Knowledge in Data: An Introduction to Data Mining*, John Wiley & Sons, Inc., 2005, pp. 90-106.

[64] D. T. Larose, «Decision Trees,» σε *Discovering Knowledge in Data: An Introduction to Data Mining*, John Wiley & Sons, Inc., 2005, pp. 107-127.

[65] L. Breiman, «Random Forests,» *Machine Learning,* τόμ. 45, αρ. 1, pp. 5-32, 2005.

[66] A. Cutler, D. R. Cutler και J. R. Stevens, «Random Forests,» σε *Ensemble Machine Learning: Methods and Applications*, C. Zhang και Y. Ma, Επιμ., Boston, MA, Springer US, 2012, pp. 157-175.

[67] C. M. Bishop, «Linear Models for Classification,» σε *Pattern Recognition and Machine Learning*, Springer-Verlag New York, 2006, pp. 179-218.

[68] W. H. Greene, Econometric Analysis, Seventh ed., Boston: Pearson Education, 2012, pp. 803-806.

[69] J. Engel, «Polytomous logistic regression,» *Statistica Neerlandica,* τόμ. 42, αρ. 4, December 1988.

[70] G. A. Seber και A. J. Lee, Linear Regression Analysis, Hoboken, New Jersey: WILEY, 2003.

[71] D. J. MacKay, «Bayesian Interpolation,» *Neural Computation,* τόμ. 4, pp. 415-447, 1992.

[72] D. A. Freedman, Statistical Models: Theory and Practice, Cambridge University Press, 2009, p. 26.

[73] C.-C. Chang και C.-J. Lin, «LIBSVM: A Library for Support Vector Machines,» *ACM Trans. Intell. Syst. Technol.,* τόμ. 2, pp. 27:1--27:27, 3 April 2011.

[74] C.-W. Hsu, C.-C. Chang και C.-J. Lin, «A Practical Guide to Support Vector Classification,» Taipei 106, Taiwan , 2003.

[75] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang και C.-J. Lin, «LIBLINEAR: A Library for Large Linear Classification,» *Journal of Machine Learning Research,* τόμ. 9, pp. 1871--1874, 2008.

[76] P.-N. Tan, M. Steinbach και V. Kumar, «Τεχνητό Νευρωνικό Δίκτυο (Artificial Neural Network - ANN),» σε *Εισαγωγή στην Εξόρυξη Δεδομένων (Introduction to Data Mining)*, Boston, MA, USA, Addison-Wesley Longman Publishing Co., Inc., 2006, pp. 271-281.

[77] F. Rosenblatt, Principles of neurodynamics: perceptrons and the theory of brain mechanisms, Washington: Spartan Books, 1986.

[78] M. Schmidt, E. van den Berg, F. P. Michael και K. Murphy, «Optimizing Costly Functions with Simple Constraints: A Limited-Memory Projected Quasi-Newton Algorithm,» σε *12th International Conference on Artificial Intelligence and Statistics (AISTATS)*, Clearwater Beach, Florida, USA, 2009.

[79] L. Bottou, «Large-Scale Machine Learning with Stochastic Gradient Descent,» σε *Proceedings of COMPSTAT'2010: 19th International Conference on Computational StatisticsParis France, August 22-27, 2010 Keynote, Invited and Contributed Papers*, Heidelberg, Physica-Verlag HD, 2010, pp. 177-186.

[80] A. S. A., «An exact analytical relation among recall, precision, and classification accuracy in information retrieval,» 2002.

[81] R. Jesse, B. Albert, H. Geoff και P. Bernhard, «Scalable and efficient multi-label classification for evolving data streams,» *Machine Learning,* τόμ. 88, pp. 243--272, 2012.

[82] S. B. Kotsiantis, «Supervised Machine Learning: A Review of Classification Techniques,» σε *Emerging Artificial Intelligence Applications in Computer Engineering: Real Word AI Systems with Applications in eHealth, HCI, Information Retrieval and Pervasive Technologies*, 2007.

[83] A. Yeh, «More accurate tests for the statistical significance of result differences,» σε *COLING '00 Proceedings of the 18th conference on Computational linguistics*, 2000.

[84] E. C. J. O. F. D. D. L. A. C. a. A. K. S.M. Beitzel, «Improving automatic query classification via semi-supervised learning,» σε *Fifth IEEE International Conference on Data Mining (ICDM'05)*, 2005.

[85] G. Tsoumakas και I. Vlahavas, «Random k-Labelsets: An Ensemble Method for Multilabel Classification,» σε *Machine Learning: ECML 2007: 18th European Conference on Machine*

Learning, Warsaw, Poland, September 17-21, 2007. Proceedings, K. J. N., K. Jacek, M. R. L. de, M. Stan, M. Dunja και S. Andrzej, Επιμ., Springer Berlin Heidelberg, 2007, pp. 406-417.

[86] P. Domingos, «A Unified Bias-Variance Decomposition for Zero-One and Squared Loss,» σε *Seventeenth National Conference on Artificial Intelligence*, Austin Texas, 2000.

[87] A. Esuli και F. Sebastiani, «SENTIWORDNET: A Publicly Available Lexical Resource for Opinion Mining,» σε *In Proceedings of the 5th Conference on Language Resources and Evaluation (LREC' 06)*, 2006, pp. 417-422.