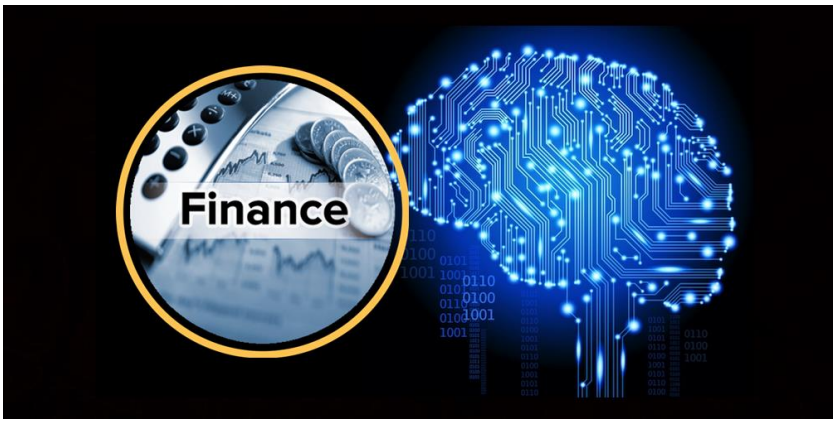
Πανεπιστήμιο Δυτικής Μακεδονίας

Πολυτεχνική Σχολή

Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Διπλωματική Εργασία

Σύγκριση μεθόδων για την εκτίμηση χρηματιστηριακής μετοχής στην αυτοκινητοβιομηχανία με την χρήση αλγορίθμων μηχανικής μάθησης



Φοιτητής: Μακρίδης Φώτιος

Α.Ε.Μ. : 707

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

Ιούλιος 2019, Κοζάνη

Εικόνα Εξωφύλλου: Kevin Chng, Machine Learning Classification Used To Predict Stock, MATLAB Central File Exchange

University Of Western Macedonia

Faculty of Engineering

Department of Electrical and Computer Engineering

Diploma Thesis

Comparison of methods for estimating stock market share in the automotive industry using machine learning algorithms

Student: Makridis Fotios

Student Number: 707

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

July 2019, Kozani

Book Cover: Kevin Chng, Machine Learning Classification Used To Predict Stock, MATLAB ile Exchange

Ευχαριστίες

Με την εκπόνηση της συγκεκριμένης διπλωματικής εργασίας θα ήθελα να ευχαριστήσω αρχικά τον Επίκουρο Καθηγητή του τμήματός μας, κύριο Παναγιώτη Σαρηγιαννίδη, ο οποίος δέχθηκε να συνεργαστεί μαζί μου και μου εμπιστεύτηκε την ανάθεση και την υλοποίηση ενός τέτοιου θέματος, όπως επίσης και δύο πρώην φοιτητές του τμήματος μας – και νυν διδακτορικούς φοιτητές εν συνεχεία-, τον Αναστάσιο Λύτο και τον Πάρι-Αλέξανδρο Καρυπίδη, οι οποίοι δέχθηκαν με τη σειρά τους να συνεργαστούμε και να με βοηθήσουν σε αρκετά τεχνικά κομμάτια της συγκεκριμένης εργασίας.

Δεν θα μπορούσα να παραλείψω επίσης τη ψυχολογική εμψύχωση της οικογένειάς μου και την απεριόριστη ανεκτικότητά τους σε μία μεγάλη περίοδο πίεσης για μένα, μέχρι να καταφέρω να ολοκληρώσω την εργασία, όπως επίσης και την καταλυτική συμβολή των φίλων μου, ούτως ώστε να ανταπεξέλθω στις απαιτήσεις ενός τέτοιου εγχειρήματος.

Περίληψη

Το θέμα της παρούσας διπλωματικής εργασίας πραγματεύεται μερικές από τις τεχνικές μηχανικής μάθησης, οι οποίες μπορούν να χρησιμοποιηθούν για την κατανόηση και επεξεργασία της φυσικής γλώσσας ως προς το συναισθηματικό της ύφος. Σκοπός της παραπάνω ανάλυσης αποτελεί η εκτίμηση των χρηματιστηριακών μετοχών τριών διαφορετικών εταιρειών στο χώρο της αυτοκινητοβιομηχανίας. Τα δεδομένα τα οποία συλλέγονται και χρησιμοποιούνται, πρόκειται για δεδομένα κειμένου τα οποία προέρχονται από δύο διαφορετικές πηγές. Το πρώτο στάδιο μετά τη συλλογή τους είναι η επεξεργασία τους και στη συνέχεια εφαρμόζεται η τεχνική της ανάλυσης συναισθήματος (Sentiment Analysis).

Όπως προαναφέρθηκε, τα δεδομένα των κειμένων τα οποία συλλέγονται προέρχονται από δύο διαφορετικές πηγές. Η πρώτη πηγή είναι το οικονομικό portal Investing.com και η δεύτερη πηγή είναι ένα από τα πιο δημοφιλή κοινωνικά δίκτυα, το Twitter. Τα κείμενα από την πρώτη πηγή προέρχονται από άρθρα δημοσιογράφων που δημοσιεύονται στο Investing.com ενώ τα κείμενα της δεύτερης πηγής είναι αναρτήσεις/δημοσιεύσεις χρηστών στην πλατφόρμα του Twitter (tweets). Τα δεδομένα αυτά αφορούν τρεις (3) αυτοκινητοβιομηχανίες/κολοσσούς, οι οποίες είναι η Tesla, η Ford και η Volkswagen αντίστοιχα.

Η ανάλυση των κειμένων και η εκτίμηση των μετοχών βασίζεται σε τρία (3) διαφορετικά μοντέλα. Το πρώτο μοντέλο αφορά την ανάλυση και επεξεργασία των δεδομένων που προέρχονται μόνο από το Twitter, το δεύτερο μοντέλο εφαρμόζεται σε δεδομένα που προέρχονται μόνο από τα άρθρα του Investing ενώ το τρίτο μοντέλο πρόκειται για τον συνδυασμό των δύο πρώτων περιπτώσεων που προαναφέρθηκαν, δηλαδή τον συνδυασμό δεδομένων τόσο από το Twitter όσο και από το Investing. Επομένως, για κάθε αυτοκινητοβιομηχανία προέκυψαν τρεις συλλογές δεδομένων (Datasets), στις οποίες προστέθηκαν οι πραγματικές εναλλαγές των τιμών μετοχών των εταιρειών, προερχόμενες επίσης από την ιστοσελίδα του Investing.com.

Η επεξεργασία των κειμένων περιλαμβάνει την ομαδοποίηση των δεδομένων ανά ημέρα, την αφαίρεση των σημείων στίξης αλλά και λέξεων χωρίς κάποια ιδιαίτερη αξία, όσον αφορά την ανάλυση συναισθήματος (stop words). Στη συνέχεια έγινε χρήση τριών διαφορετικών λεξικών, τα οποία αξιολογούν τα κείμενα ως προς το συναισθηματικό ύφος, υλοποιημένα με διαφορετικές τεχνικές, αλλά με κοινό στόχο να αποτυπώσουν το συναισθηματικό **φόρτο** της κάθε λέξης. Από κάθε λεξικό, προκύπτει ένας συναισθηματικό τόνος (tone index) για κάθε εγγραφή των datasets.

Επιπλέον, όσον αφορά τις συλλογές δεδομένων των tweets, συλλέχτηκαν αντιδράσεις (user interactions) οι οποίες γίνονται από τους χρήστες στην πλατφόρμα του Twitter και αφορούν τον αριθμό των retweets και των favorites, ενώ επίσης υπολογίστηκε ο αριθμός των αναφορών (mentions - @), των ετικετών (hashtags - #), το σύνολο των επιθέτων, ουσιαστικών και ρημάτων.

Μετά την ολοκλήρωση της συλλογής όλων των δεδομένων και της επεξεργασίας τους, ερευνήθηκαν διάφορες τεχνικές και αλγόριθμοι μηχανικής μάθησης, με τη βοήθεια των οποίων προκύπτουν τα αποτελέσματα και η εκτίμηση των χρηματιστηριακών μετοχών των εταιρειών. Οι ταξινομητές μηχανικής μάθησης (Machine Learning Classifiers) οι οποίοι χρησιμοποιήθηκαν ήταν η Μέθοδος Κ-Κοντινότερων Γειτόνων (k-nearest neighbors algorithm - KNN), η οικογένεια

αλγορίθμων Απόφασης Δένδρου (Decision Tree) και ο αλγόριθμος Τυχαίου Δάσους (Random Forest).

Σε τελικό στάδιο, πραγματοποιήθηκε σύγκριση των μοντέλων που αναπτύχθηκαν και των αποτελεσμάτων τους, ενώ τέθηκαν ορισμένες μελλοντικές επεκτάσεις των διαδικασιών, οι οποίες αποβλέπουν τόσο στον επιστημονικό όσο και στον επιχειρηματικό τομέα.

Λέξεις κλειδιά: Χρηματιστηριακές μετοχές, μηχανική μάθηση, ανάλυση συναισθήματος, Investing.com, Twitter Api, ταξινομητές, Random Forest, k-nearest neighbors algorithm, Decision Tree, λεξικά, crawlers, εξόρυξη δεδομένων, ανάλυση συναισθήματος, εκτίμηση.

Abstract

The topic of the present Master thesis deals with some of the techniques of machine learning, which can be used in order to process and understand the natural language, considering their sentiment analysis. Furthermore, as a result of the above process and analysis, there were made estimations of the stock market price of three (3) different automotive industries. The data that were collected and used, come from two different sources and were processed and applied into them the technique of the Sentiment Analysis.

As mentioned above, the data that were collected from two different sources. The first one is the financial portal Investing.com and the latter one is one of the most popular social media platforms, Twitter. The texts from the first source derive from articles that are posted from journalists in the website Investing.com, whereas the texts from the second source are tweets that are posted from a variety of accounts, in the Twitter platform. These data are about three (3) giant automotive industries, Tesla, Ford and Volkswagen.

Text analysis and stock market price estimations are being held in three (3) different models. The first one includes only the analysis and the processing of the data that come from Twitter, the second one includes only the analysis and the processing of the data that come from Investing.com and the last one is a result of the combination of the two (2) previous models mentioned. Thus, for each company three (3) different datasets arose, in which were added the actual stock market price changes per day, collected also from Investing.com.

The text processing contains the grouping of plain texts per day, the subtraction of the punctuation marks and the subtraction of a specific group of words, named as stop words that are filtered out before the processing of natural language data. This kind of words usually refer to the most common words in a language, such as articles, conjunctions and pronouns. Afterwards, there were used three (3) different lexicons, that appraise the processed texts as for theirs sentimental form and level, implemented with different techniques but with the common goal to reflect the sentimental load of each individual word. For each lexicon, there is a total sum for each day.

Moreover, as for the Twitter Datasets, there were collected some Twitter reactions that are being made by the users and have to do with the number of retweets and favorites, whilst additionally there were calculated the number of mentions (@), the number of hashtags (#) and the numbers of adjectives, nouns and verbs. In the articles' part, it was calculated the total number of articles for each day.

After the completion of the data collection and processing, some techniques and machine learning algorithms were investigated, which aided to study the natural language and the estimation of the stock market prices of the automotive industries. The classifiers that were used were the k-nearest neighbors algorithm (KNN), the Decision Tree family algorithms and the Random Forest algorithm.

In the final part of the thesis, there were made comparisons between the results of the estimations of the models that were built and there were addressed some future extensions that can be applied into, which focus in both the academic and industry sectors.

Keywords: Stock Market prices, machine learning, sentiment analysis, Investing.com, Twitter Api, classifiers, Random Forest, k-nearest neighbors algorithm, Decision Tree, lexicons, crawlers, big data mining, estimation.

Πίνακας Περιεχομένων

Κεφάλαιο 1: Εισαγωγή

1.1 Πρόβλημα και στόχος εργασίας

Είναι γεγονός ότι βιώνουμε μία εποχή ραγδαίας τεχνολογικής ανάπτυξης, η οποία φέρνει συνεχώς αλλαγές στην καθημερινή μας ζωή. Η ολοένα και μεγαλύτερη χρήση του διαδικτύου έχει οδηγήσει και σε μεγάλη συλλογής πληροφορίας. Η πληροφορία μπορεί να προέλθει από διάφορες πηγές, όπως για παράδειγμα τα μέσα κοινωνικής δικτύωσης και τις δημοσιογραφικές πηγές ενημέρωσης.

Με την ραγδαία εξέλιξη της τεχνολογίας και την ολοένα και μεγαλύτερη χρήση των πληροφοριακών συσκευών στην καθημερινότητά μας, βρισκόμαστε αντιμέτωποι με ένα τεράστιο όγκο δεδομένων. Η επιστημονική κοινότητα έχει εντοπίσει αυτή την ευκαιρία για εκμετάλλευση των νέων δεδομένων και κάνει προσπάθειες για την καλύτερη 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.

Το ερώτημα είναι πως μπορεί να αξιοποιηθεί η ανάλυση συναισθήματος και η επεξεργασία της πληροφορίας στον οικονομικό τομέα και συγκεκριμένα το χρηματιστήριο, προκειμένου να εκτιμήσουμε τις χρηματιστηριακές μετοχές διαφόρων εταιρειών. Η προσεκτική ανάλυση και μελέτη των απόψεων, αισθημάτων και συναισθημάτων μπορούν να χρησιμοποιηθούν με σκοπό την εκτίμηση της μελλοντικής πορείας εταιρειών στο χρηματιστήριο.

Στην πράξη οι περισσότερες μέθοδοι προσέγγισης για την ανάλυση συναισθήματος υιοθετούν μία στρατηγική δύο βημάτων [1]. 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 της ανάλυσης συναισθήματος είναι η κατηγοριοποίηση του κειμένου όσον αφορά την χροιά ή την πολικότητά (polarity) του, όπου οι υποκειμενικές προτάσεις κατηγοριοποιούνται ως θετικές ή αρνητικές. Το κύριο ερώτημα στο οποίο απαντάει το ανάλυση συναισθήματος είναι ‘τι σκέφτονται οι άλλοι’, μέσα από ανάλυση κειμένου σε δημοσιεύσεις στα κοινωνικά δίκτυα και σε άρθρα ειδήσεων.

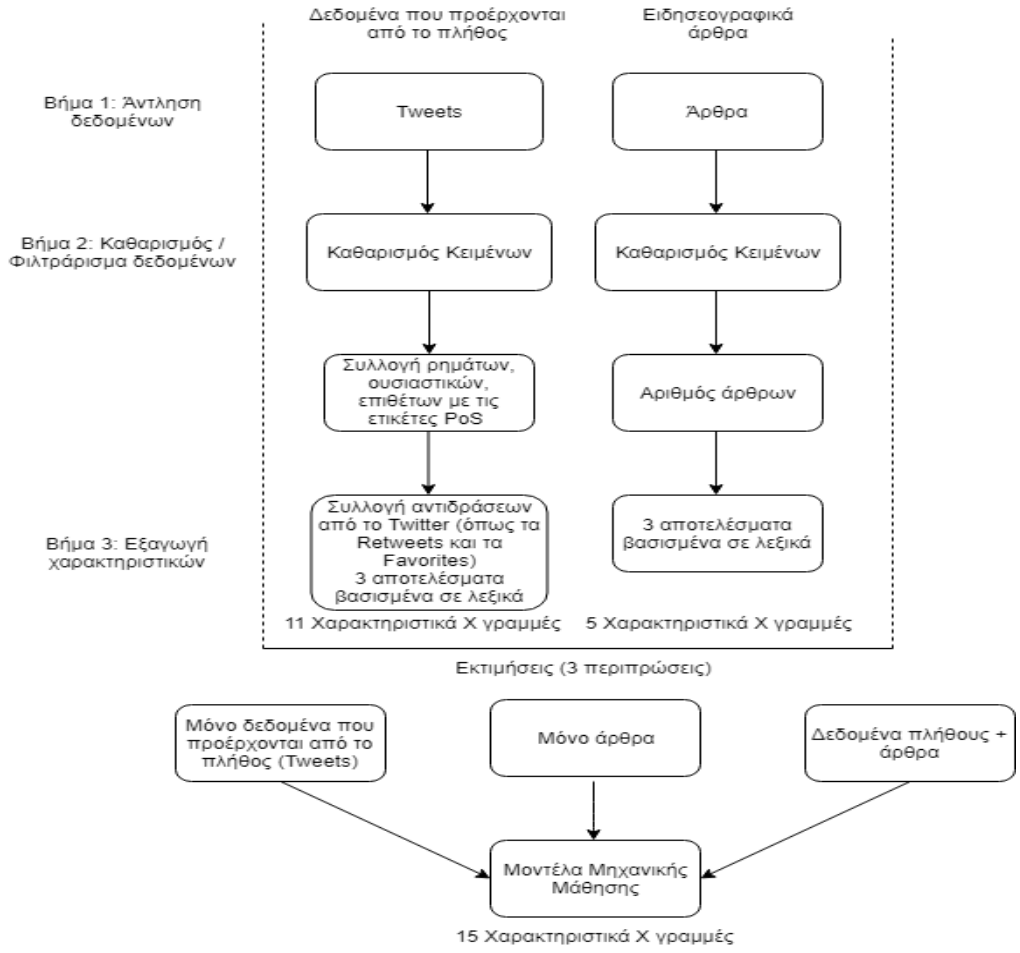
Το βασικό πρόβλημα είναι με ποιον τρόπο θα μπορούσαμε να εξασφαλίσουμε ότι οι πληροφορίες που εξάγονται από την συναισθηματική ανάλυση των κειμένων μπορούν να εκτιμήσουν με ορθό τρόπο την μελλοντική πορεία των μετοχών των εταιρειών στο χρηματιστήριο. Στόχος της συγκεκριμένης εργασίας είναι η ανάπτυξη μεθόδων και τεχνικών, σύμφωνα με τις οι οποίες μπορούμε να αξιοποιήσουμε δύο βασικές παραμέτρους: 1) τις απόψεις και το συναίσθημα του κοινού μέσα από τα μέσα κοινωνική δικτύωσης και 2) τις κριτικές και τις εκτιμήσεις των οικονομικών αναλυτών σε οικονομικά forum.

Επομένως, με πηγές άντλησης των δεδομένων την πλατφόρμα του μέσου κοινωνικής δικτύωσης του Twitter και το οικονομικό portal Investing.com, γίνεται εξόρυξη δεδομένων (data mining) και επεξεργασία φυσική γλώσσας (natural language processing), χρησιμοποιώντας

τεχνικές μηχανικής μάθησης, προκειμένου να εκτιμηθεί η χρηματιστηριακή πορεία των μετοχών τριών εταιρειών της αυτοκίνησης.

1.2 Αρχιτεκτονική εργασίας

Η αρχιτεκτονική δομή της εργασίας χωρίζεται ουσιαστικά σε δύο στάδια, όπως φαίνεται και από την εικόνα 1.



Εικόνα 1 - Αρχιτεκτονική δομή διπλωματικής εργασίας

Στο πρώτο στάδιο, διακρίνουμε τις δύο πηγές δεδομένων, δεδομένα τα οποία προέρχονται από τις δημοσιεύσεις των χρηστών του Twitter και δεδομένα, δηλαδή άρθρα που προέρχονται από το Investing.com. Μετά την συλλογή των δεδομένων από αυτές τις δύο πηγές, ακολουθεί η επεξεργασία τους η οποία χωρίζεται σε τρία βήματα.

Το πρώτο βήμα σε αυτό το στάδιο είναι η άντληση των δεδομένων από τις προαναφερθείσες πηγές, μέσω του λογισμικού συλλογής άρθρων για τον οικονομικό portal Investing.com και μέσω του λογισμικού εξαγωγής tweets από την πλατφόρμα του Twitter. Τα δεδομένα αυτά συλλέχθηκαν με τελική ημερομηνία η οποία ορίστηκε από εμένα, ανάλογα με τον όγκο των δεδομένων που υπήρχε. Χρειάζεται να διευκρινιστεί ότι οι τελικές ημερομηνίες για τις τρεις αυτοκινητοβιομηχανίες είναι τελείως διαφορετικές μεταξύ τους, γεγονός το οποίο επηρεάζει ο όγκος του κάθε συνόδου δεδομένων, αλλά ίσως και την αξιοπιστία των τελικών μοντέλων.

Το δεύτερο βήμα αποτελεί αρχικά τον καθαρισμό των δεδομένων και για τις δύο πηγές, όπως επίσης και η ταξινόμησή τους ανά ημέρα. Γίνεται διαχωρισμός των σημείων στίξης από το υπόλοιπο κείμενο, ενώ υπάρχει και αφαίρεση λέξεων οι οποίες δεν συνεισφέρουν ουσιαστικά στην ανάλυσης του συναισθήματος που ακολουθεί.

Το τελευταίο βήμα της πρώτης φάσης, περιλαμβάνει τον καθορισμό των ανεξάρτητων μεταβλητών (features), οι οποίες είναι διαφορετικές για τις συλλογές δεδομένων που προέρχονται από διαφορετική πηγή. Πιο συγκεκριμένα στα δεδομένα τα οποία προέρχονται από το twitter ανεξάρτητες μεταβλητές ορίζουμε τις αντιδράσεις των χρηστών όπως τα retweets και τα favorites, τον συναισθηματικό τόνο που προκύπτει από τα τρία λεξικά και τα σύνολα των ουσιαστικών, ρημάτων και αντωνυμιών. Σε αυτά τα χαρακτηριστικά, προσθέτουμε και μία εξαρτημένη μεταβλητή, η οποία αποτελεί την εναλλαγή του προσήμου των χρηματιστηριακών μετοχών κάθε εταιρείας ανά ημέρα. Αναφορικά με το κομμάτι των άρθρων, οι πληροφορίες που χρησιμοποιούμε στην ανάλυση αφορούν τον αριθμό των άρθρων της εκάστοτε ημέρας και τα αποτελέσματα τους όσο αφορά τον συναισθηματικό τους τόνο κάνοντας χρήση των τριών λεξικών ενώ ως εξαρτημένη μεταβλητή έχουμε πάλι τις εναλλαγές των τιμών των μετοχών ανά ημέρα. Η τρίτη κατηγορία συλλογής δεδομένων είναι ο συνδυασμός των προηγούμενων δύο, με αποτέλεσμα να προκύπτει ένας τελικός αριθμός από δεκατέσσερις ανεξάρτητες μεταβλητές και μία εξαρτημένη.

Στο δεύτερο στάδιο, ορίζονται τα τρία διαφορετικά μοντέλα για την εκτίμηση των μετοχών. Το πρώτο μοντέλο αποτελείται από τα datasets των άρθρων, το δεύτερο από τα datasets των tweets ενώ το τρίτο από τα datasets τόσο των άρθρων όσο και των tweets. Και στα τρία αυτά μοντέλα, εφαρμόζουμε διάφορες τεχνικές μηχανικής μάθησης με αλγορίθμους οι οποίοι έχουν ως αποτέλεσμα τις εκτιμήσεις των μετοχών σε μορφή δυαδική μορφή (άνοδος/μείωση). Τελικό βήμα αποτελεί η αξιολόγηση των μοντέλων μηχανικής μάθησης καθώς και η μεταξύ τους σύγκριση όσο αφορά την αξιοπιστία τους.

1.3 Δομή εργασίας

Η δομή της συγκεκριμένης διπλωματική εργασίας, χωρίζεται σε θεωρητικό και πειραματικό μέρος. Ο διαχωρισμός αυτός γίνεται προκειμένου να γίνει μία εισαγωγή στην επιστήμη της τεχνητής νοημοσύνης και του τομέα των «Μεγάλων Δεδομένων» και στη συνέχεια να αναλυθούν όλα τα εργαλεία που χρησιμοποιήθηκαν, με βάση τα οποία προέκυψαν οι εκτιμήσεις και τα αποτελέσματα.

Ο συγκεκριμένος τρόπος διαχωρισμού της εργασίας δίνει την δυνατότητα στον αναγνώστη να κατανοήσει με σχετική ευκολία τη φιλοσοφία του εγχειρήματος, τους τρόπους ανάπτυξης και τα συμπεράσματα που προκύπτουν.

Ενδεικτικά η δομή:

- Παρουσίαση του θεωρητικού υπόβαθρου της επιστήμης των δεδομένων και των τεχνικών που χρησιμοποιούνται.
- Αναφορά των εργαλείων με τα οποία έγινε η επεξεργασία όλων των δεδομένων, και των δεδομένων τα οποία χρησιμοποιήθηκαν για την εκτίμηση των μετοχών.
- Συλλογή των δεδομένων που χρησιμοποιήθηκαν για τη δημιουργία των συλλογών δεδομένων. Πως λειτουργούν τα λογισμικά εξαγωγής δεδομένων, ποιες εταιρίες επέλεξα και γιατί, ποια είναι αυτά τα δεδομένα που χρησιμοποιούνται και πως.
- Δημιουργία και επεξεργασία των συγκεκριμένων συλλογών, η οποία περιλαμβάνει την ταξινόμηση των κειμένων, την αφαίρεση περιττών λέξεων και σημείων στίξης, τον καθορισμό των ανεξάρτητων μεταβλητών και της εξαρτημένης και τέλος το χωρισμό σε τρία μοντέλα συλλογών δεδομένων.
- Η εξαγωγή του συναισθηματικού τόνου των δεδομένων σύμφωνα με τα τρία λεξικά που επιλέχθηκαν να χρησιμοποιηθούν, μέτρηση του συνόλου των κειμένων που αφορούν τα άρθρα και μέτρηση σημαντικών σημείων του λόγου, όπως ρήματα, ουσιαστικά, αντωνυμίες που αφορούν τα tweets.
- Εφαρμογή των τεχνικών μηχανικής μάθησης και των αλγορίθμων, ούτως ώστε να προκύψουν οι εκτιμήσεις και να αξιολογηθούν τα αποτελέσματα.
- Αξιολόγηση και σύγκριση των μοντέλων, εξαγωγή συμπερασμάτων και ερμηνεία αποτελεσμάτων.
- Στο τελικό στάδιο της εργασίας, γίνεται αναφορά στα συμπεράσματα που προκύπτουν από τη χρήση όλων των μεθόδων και των βημάτων που ακολουθήθηκαν και προτείνονται μερικές μελλοντικές επεκτάσεις του συγκεκριμένου εγχειρήματος.

Κεφάλαιο 2: Έννοιες και Ορισμοί

2.1 Μεγάλα δεδομένα και εξόρυξη κειμένου

Τι ακριβώς σημαίνει ο ορός «Μεγάλα Δεδομένα»; Αρχικά, σαν έννοια ο όρος είναι αρκετά ασαφής. Τα «Μεγάλα Δεδομένα» είναι ένας όρος ο οποίος περιγράφει την μεγάλη συλλογή από δεδομένα – τα οποία μπορεί να είναι είτε δομημένα είτε όχι – τα οποία «πλημμυρίζουν» μία επιχείρηση σε καθημερινή βάση. Ωστόσο, δεν είναι ο όγκος των δεδομένων που είναι σημαντικός. Είναι το τι κάνουν οι επιχειρήσεις με τα δεδομένα που είναι σημαντικά. Με τη βοήθεια των μεγάλων δεδομένων, οι εταιρείες μπορούν να αναλύσουν πολλούς τομείς της επιχείρησής τους, με αποτέλεσμα να οδηγηθούν σε καλύτερες αποφάσεις και στρατηγικές κινήσεις.

Ενώ ο όρος «Μεγάλα Δεδομένα» θεωρείται αρκετά νέος, οι ενέργειες άντλησης και αποθήκευσης τεράστιου όγκου δεδομένων για μελλοντική ανάλυση, είναι αρκετά παλαιές. Η ιδέα άρχισε να ενσαρκώνεται τα πρώτα χρόνια της νέας χιλιετίας, όπου ο αναλυτής επιχειρήσεων Doug Laney όρισε το νέο ρεύμα των «Μεγάλων Δεδομένων» ως τα τρία V: 1) Volume – Όγκος 2) Velocity – Ταχύτητα 3) Variety – Ποκιλομορφία. [2] [3]

Ο τομέας στον οποίο στοχεύει αυτή η εργασία είναι αυτός της οικονομίας. Οι εφαρμογές των «Μεγάλων Δεδομένων» στον οικονομικό τομέα είναι πολλές, περιλαμβάνοντας την ανάλυση των μέσων κοινωνικής δικτύωσης, τις αναλύσεις διαδικτύου, την διαχείριση ρίσκου και την ανίχνευση απάτης. Ένας από τους πιο επιτυχημένους τρόπους εξαγωγής δεδομένων από ένα μεγάλο όγκο δεδομένων, είναι με την τεχνική της εξόρυξης κειμένου. [4]

Ο σκοπός της εξόρυξης κειμένου (γνωστή και ως εξόρυξη δεδομένων κειμένου και αναλύσεις κειμένου) είναι η ανάλυση εγγράφων κειμένου (όπως e-mail, απλών κειμένων, εκθέσεων, άρθρων). Αποτέλεσμα αυτής της εξόρυξης είναι δεδομένα, τα οποία μετατρέπονται σε ποικίλους τύπους λήψης και διαχείρησης αποφάσεων. Οι τεχνικές ανάλυσης μπορεί να είναι είτε γλωσσικές, είτε στατιστικές, είτε αναλύσεις μηχανικής μάθησης. [5]

Η συγκεκριμένη εργασία συνδυάζει την εξόρυξη δεδομένων κειμένου στα μέσα κοινωνικής δικτύωσης, αντλώντας κείμενα και άρθρα, τα οποία αναλύονται με τη βοήθεια τεχνικών της μηχανικής μάθησης.

2.2 Εξόρυξη Μέσων Κοινωνικής Δικτύωσης

Τα μέσα κοινωνικής δικτύωσης «σπάνε» τα όρια μεταξύ του πραγματικού κόσμου και του εικονικού κόσμου. Μπορούμε πλέον, να ενσωματώσουμε κοινωνικές θεωρίες με υπολογιστικές μεθόδους, προκειμένου να εξετάσουμε πως ο καθένας ξεχωριστά συμπεριφέρεται και πως

δημιουργούνται οι κοινότητες των κοινωνικών δικτύων. Σαφέστατα, όταν μιλάμε για τα δεδομένα των κοινωνικών δικτύων, καταλαβαίνουμε ότι μιλάμε για έναν τεράστιο όγκο ανεπεξεργαστών δεδομένων, ανάμεσα σε διαφορετικές πλατφόρμες (π.χ. Facebook, Twitter, Instagram), τις οποίες χρησιμοποιούν δισεκατομμύρια χρήστες. Η μοναδικότητα των δεδομένων των μέσων κοινωνικής δικτύωσης δημιουργούν την ανάγκη επεξεργασίας τους, με τεχνικές οι οποίες μπορούν αποτελεσματικά να εξάγουν συμπεράσματα για το περιεχόμενο των δεδομένων και να το κρίνουν , ως προς την αντίδραση, το συναίσθημα και τον τόνο του.

Οι τεχνικές αυτές συνθέτουν αυτό που αποκαλούμε εξόρυξη των Μέσων κοινωνικής Δικτύωσης. Εξόρυξη των Μέσων Κοινωνικής Δικτύωσης είναι η διαδικασία της αναπαράστασης, ανάλυσης και εξαγωγής πραγματοποιήσιμων προτύπων, από τα δεδομένα των μέσων κοινωνικής δικτύωσης. Η εξόρυξη αυτή εισάγει αλγορίθμους ικανούς να ερευνήσουν τα δεδομένα τεραστίου όγκου που προκύπτουν από τα μέσα κοινωνικής δικτύωσης και να τα αξιολογήσει συνδυάζοντας θεωρίες και μεθοδολογίες, με διαφορετικές αρχές, όπως την επιστήμη των υπολογιστών, τη μηχανική μάθηση, την ανάλυση κοινωνικών δικτύων, την κοινωνιολογία, τη στατιστική και τα μαθηματικά. Μερικές από τις δύσκολες προκλήσεις που αντιμετωπίζει η διαδικασία αυτή, έχει να κάνει με τον περιορισμένο αριθμό δεδομένων που μπορούν να αντληθούν ανά ημέρα, μέσω των Api (Application Programming Interface - Διεπαφή Προγραμματισμού Εφαρμογών), την διασφάλιση της εγκυρότητας των δεδομένων και των πηγών και την απομάκρυνση του θορύβου από τα κείμενα.

Ουσιαστικά, στην παρούσα εργασία εφαρμόζεται η εξόρυξη δεδομένων στην κοινωνική πλατφόρμα του Twitter, χρησιμοποιώντας έναν Twitter crawler για να αντλήσω τα δεδομένα μου, τα οποία στα συνέχεια τα αναλύσω με αλγορίθμους και τεχνικές μηχανικής μάθησης. Τα προβλήματα που αντιμετώπισα ήταν ο περιορισμός των δεδομένων από την πλατφόρμα του Twitter, το πρόβλημα της διασφάλισης της εγκυρότητας των δεδομένων και η ανάγκη για καθαρισμό των κειμένων από περιττές λέξεις και σημεία στίξης. [6]

2.3 Μηχανική Μάθηση

Μηχανική μάθηση είναι μία κατηγορία της επιστήμης των υπολογιστών, η οποία κάνει χρηση δεδομένων από έναν αλγόριθμο ο οποίος εκτελείται σε μια υπολογιστική μηχανή, με σκοπό τη σταδιακή βελτίωσή της κατά την εκτέλεση μιας λειτουργίας. [7] Η μηχανική μάθηση επικεντρώνεται κυρίως στην ανάπτυξη προγραμμάτων τα οποία μπορούν να παράγουν αποτελέσματα ακόμα και όταν η είσοδος του είναι διαφορετική κάθε φορά. [8]

Ως παράδειγμα μπορούμε να θεωρήσουμε τη λειτουργία της ροής ειδήσεων (News Feed) του Facebook, το οποίο χρησιμοποιεί Μηχανική Μάθηση, ούτως ώστε να παράγει εξατομικευμένες πληροφορίες για κάθε έναν χρήστη ξεχωριστά, σύμφωνα με τις προσωπικές του προτιμήσεις. Αν ένας χρήστης διαβάζει συχνά τις καταστάσεις κάποιων συγκεκριμένων ατόμων, τότε η ροή ειδήσεων θα αρχίσει να προσαρμόζεται σε αυτή του την συμπεριφορά. Η λειτουργία η οποία κρύβεται πίσω από αυτό το μηχανισμό πρόκειται για μία στατιστική ανάλυση συνδυασμένη

με μηχανισμούς εκτιμήσεων, με σκοπό να εξαχθούν κάποια μοτίβα στα δεδομένα που θα δέχεται ο κάθε χρήστης. [9] [10]

Τα προβλήματα που μπορούν να λυθούν με τη βοήθεια της μηχανικής μάθησης ποικίλουν σε αριθμό, αλλά μπορούν να ταξινομηθούν σε τρεις μεγάλες κατηγορίες: Επιτηρούμενη Μάθηση (ή ακόμα και μάθηση με επίβλεψη – Supervised Learning), Μη επιτηρούμενη Μάθηση (ή ακόμα και μάθηση χωρίς επίβλεψη – Unsupervised Learning), Εξαναγκασμένη Μάθηση (Reinforcement Learning).

Οι διαφορές των τριών αυτών κατηγοριών αφορούν κυρίως το στάδιο της εκπαίδευσης και λύσης των εκάστοτε προβλημάτων. Στην περίπτωση της Μηχανικής Μάθησης με Επίβλεψη, το στάδιο της εκπαίδευσης βασίζεται σε ένα σύνολο ήδη απαντημένων ερωτήσεων (Training Set). Στην δεύτερη περίπτωση, της Μάθησης χωρίς Επίβλεψη, το στάδιο της εκπαίδευσης βασίζεται σε ένα ίδιας μορφής σύνολο όπως και προηγουμένως, αλλά χωρίς να είναι γνωστές εκ τον προτέρων οι απαντήσεις (Clustering). Τέλος στην κατηγορία της Εξαναγκασμένης Μάθησης, το πρόγραμμα αλληλεπιδρά με ένα δυναμικό περιβάλλον από το οποίο λαμβάνει μία θετική ή αρνητική βαθμολογία, ανάλογα με την ενέργεια που εκτελεί σε αυτό. Με αυτόν τον τρόπο εκπαιδεύεται στο να μαθαίνει και να αξιολογεί της ενέργειες του καθώς αυτές προσαρμόζονται μέσα στο περιβάλλον του. [11] [12]

Το πρόβλημα της επεξεργασίας συναισθημάτων υπόκειται κυρίως στην πρώτη κατηγορία, δηλαδή αυτήν της Επιτηρούμενης Εκπαίδευσης, αφού για να απαντηθούν τα ερωτήματα τα οποία τίθενται, θα πρέπει το στάδιο της εκπαίδευσης να βασιστεί σε ήδη γνωστά δεδομένα. Ύστερα θα μπορέσουν να γίνουν οι κατάλληλες ενέργειες και υποθέσεις και να εξαχθούν αποτελέσματα για τα άγνωστα πλέον δεδομένα. Η συγκεκριμένη εργασία, βασίζεται στην Επιτηρούμενη Μάθηση, καθώς τα δεδομένα των κειμένων που αναλύονται ανήκουν στην επεξεργασία της ανάλυσης συναισθήματος.

2.4 Ανάλυση συναισθήματος

Ανάλυση συναισθήματος ονομάζεται το πεδίο της επιστήμης, το οποίο μελετάει και αναλύει τις απόψεις, τα συναισθήματα, τις αξιολογήσεις, και τις εκτιμήσεις των ανθρώπων προς ορισμένες οντότητες όπως π.χ. προϊόντα, υπηρεσίες, οργανισμούς, άλλα άτομα, εκδηλώσεις κτλ. [13]

Με άλλα λόγια με τον ορισμό της ανάλυσης συναισθήματος αναφερόμαστε στην διαδικασία με την οποία χαρακτηρίζουμε και αποφασίζουμε το συναισθηματικό τόνο ή ύφος ενός συνόλου λέξεων, είτε αυτό πρόκειται για κάποιο άρθρο ή φράση κτλ. κυρίως στο χώρο του διαδικτύου. Η συγκεκριμένη υπηρεσία, ειδικά στον 21ο αιώνα, αποτελεί ένα ιδιαίτερα χρήσιμο εργαλείο, το οποίο μπορεί να χρησιμοποιηθεί σε τομείς όπως η ηλεκτρονική αγορά προϊόντων (e-commerce), η οικονομία, η πολιτική, και γενικά σε τομείς που έχουν να κάνουν με την ανίχνευση συναισθημάτων και την έκφραση της ανθρώπινης γνώμης (opinion extraction). Ακόμα, η συγκεκριμένη υπηρεσία, σε επόμενα στάδια, θα μπορούσε να εφαρμοστεί για να εκτιμηθούν και να αποφευχθούν περιπτώσεις αυτοκτονιών, δηλαδή να αναλυθεί και να αξιολογηθεί ο

συναισθηματικός τόνος της γραφής και της έκφρασης ενός ανθρώπου με αυτοκτονικές τάσεις και εν τέλει να αποφευχθεί ένα τέτοιο περιστατικό. Για τις ανάγκες της παρούσας εργασίας θα ασχοληθούμε με τον τομέα των οικονομικών και του χρηματιστηρίου και συγκεκριμένα εκτίμηση χρηματιστηριακών μετοχών, βασισμένη στη ανάλυση συναισθήματος κειμένων από το Twitter και το Investing.com.

Οι νέες τεχνολογίες και συγκεκριμένα το τεράστιο πλήθος πληροφοριών που βρίσκονται στο διαδίκτυο, δημιούργησαν την ανάγκη ανάπτυξης μιας επιστήμης η οποία με τον συνδυασμό των τομέων των μαθηματικών, της στατιστικής, της τεχνητής νοημοσύνης και της επιστήμης των γλωσσών, θα μπορούσε να εξάγει χρήσιμες και απτές πληροφορίες και γνώσεις μέσα από έναν μεγάλο όγκο δεδομένων.

2.5 Εκτίμηση χρηματιστηριακών μετοχών και Μηχανική Μάθηση

Είναι αρκετά συχνό φαινόμενο έμποροι με αρκετά χρήματα να αγοράζουν μετοχές και ισοτιμίες σε αρκετά χαμηλές τιμές και στη συνέχεια να τις πουλάνε σε υψηλή τιμή. Η τάση της εκτίμησης τιμών μετοχών δεν είναι κάτι νέο και επεξεργάζεται συνεχώς από πολλούς οργανισμούς. Υπάρχουν δύο τρόποι με τους οποίους μπορούν να αναλυθούν οι μετοχές.

Ο πρώτος είναι η θεμελιώδης ανάλυση, στην οποία οι επενδυτές εξετάζουν παραμέτρους που επηρεάζουν την αξία της μετοχής, όπως η απόδοση της βιομηχανίας, η οικονομίας, το πολιτικό κλίμα κτλ. και εξετάζουν το ενδεχόμενο εάν θα επενδύσουν ή όχι.

Ο δεύτερος είναι η τεχνική ανάλυση, η οποία είναι η εξέλιξη των αξιών των μετοχών, με την έννοια της μελέτης των στατιστικών που προκύπτουν από τη δραστηριότητα της αγοράς, όπως οι παρελθοντικές τιμές και ο όγκος των τιμών. Η συγκεκριμένη εργασία, χρησιμοποιεί το δεύτερο τρόπο ανάλυσης, τον τεχνικό.

Τα τελευταία χρόνια, με την ολοένα και μεγαλύτερη προβολή της μηχανικής μάθησης και της χρησιμοποίησης των μεθόδων της, πολλοί επενδυτές εκμεταλλεύονται τα αποτελέσματα των τεχνικών της στον οικονομικό τομέα, μέσω της εκτίμησηςς των τιμών των μετοχών και πολλοί από αυτούς έχουν καταφέρει αξιομνημόνευτα αποτελέσματα. [14] [15]

Το χρηματιστήριο ακολουθεί ένα τυχαίο μονοπάτι, κάτι το οποίο σημαίνει ότι η καλύτερη εκτίμηση που μπορούμε να έχουμε για την αυριανή τιμή μιας μετοχής, είναι η σημερινή τιμής της. Αδιαμφισβήτητα, η εκτίμηση των δεικτών των τιμών είναι αρκετά δύσκολη, εξαιτίας της μεταβλητότητας του χρηματιστηρίου, το οποίο χρειάζεται ένα ισχυρό και ακριβές μοντέλο. Οι δείκτες τιμών μπορούν να επηρεαστούν από διάφορες παραμέτρους, όπως το κλείσιμο της προηγούμενης μέρας και το λόγο του δείκτη τιμής προς τα κέρδη (P/E Ratio – Price to Earning Ratio), με αποτέλεσμα να επηρεάσουν αρκετά την εμπιστοσύνη των επενδυτών. Γίνονται πολλές προσπάθειες να εκτιμηθομε όσο το δυνατόν καλύτερη ακρίβεια αυτές οι τιμές με τη Μηχανική Μάθηση. Η συγκεκριμένη εργασία, εστιάζει σε βραχυπρόθεσμες εκτιμήσεις, οι οποίες αφορούν την επόμενη ημέρα, ενώ οι τεχνικές Μηχανικής Μάθησης που χρησιμοποιούνται είναι η Μέθοδος Κ-Κοντινότερων Γειτόνων (k-nearest neighbors algorithm - KNN), η οικογένεια

αλγορίθμων Απόφασης Δένδρου (Decision Tree) και ο αλγόριθμος Τυχαίου Δάσους (Random Forest). [16]

Κεφάλαιο 3: Βασικά εργαλεία υλοποίησης

3.1 Python

Η Python είναι μία υψηλού επιπέδου γλώσσα προγραμματισμού με δυναμική σημασιολογία. Η υψηλού επιπέδου δομές δεδομένων της, σε συνδυασμό με τη δυναμική πληκτρολόγηση και συνοχή της, την κάνει πολύ ελκυστική για γρήγορη ανάπτυξη εφαρμογών, καθώς και για χρήση ως γλώσσα ανάπτυξης σεναρίων. Η Python χαρακτηρίζεται από απλότητα, διότι είναι πολύ εύκολο να μάθει κάποιος τη σύνταξή της, με αποτέλεσμα της μείωση του κόστους συντήρησης των προγραμμάτων.

Η Python υποστηρίζει πακέτα και έτοιμες δομές λογισμικού (modules), κάτι το οποίο ενθαρρύνει την τροποποίηση των δομών λογισμικού και την επαναχρησιμοποίηση του κώδικα. Ο διερμηνέας της Python (interpreter) και η εκτενής, βασική βιβλιοθήκη, είναι διαθέσιμα σε πηγαία ή δυαδική μορφή, με ελεύθερη διανομή για όλες τις εμπορικές πλατφόρμες και λειτουργικά συστήματα. Η τελευταία έκδοση της Python είναι η 3.7.3, η οποία κυκλοφόρησε τον Μάρτιο του 2019, είναι συμβατή με τις αρχιτεκτονικές των 32 και 64 bit. [17]

3.2 Python IDE: PyCharm

Ως περιβάλλον ανάπτυξης των αλγορίθμων και των σεναρίων σε Python, χρησιμοποιήθηκε το Python IDE (Ολοκληρωμένο περιβάλλον ανάπτυξης - Integrated Development Environment), το PyCharm, το οποίο αναπτύχθηκε και διανέμεται από την Jetbrains.

Το PyCharm προσφέρει έναν έξυπνο βοηθό για τη συγγραφή κώδικα και διαθέτει έναν πολύ εύχρηστο κειμενογράφο, ο οποίος διευκολύνει αρκετά τη διαδικασία του προγραμματισμού και της αποσφαλμάτωσης. Διαθέτει επίσης έτοιμα εργαλεία ανάπτυξης, όπως τον αποσφαλματωτή και δικό του τερματικό.

Το PyCharm διατίθεται σε δύο εκδόσεις. Την PyCharm Community Edition και την Pycharm Professional Edition, με τη διαφορά των δύο να έγκειται στο γεγονός ότι η πρώτη είναι δωρεάν ενώ η δεύτερη είναι επί πληρωμή και στη δεύτερη έκδοση υπάρχουν περισσότερα εργαλεία προγραμματιστικής ανάπτυξης. Η ανάπτυξη της συγκεκριμένης εργασίας καλύφθηκε από τη δωρεάν έκδοση, η οποία περιέχει τον έξυπνο κειμενογράφο, τον γραφικό αποσφαλματωτή, τον έξυπνο επιθεωρητή κώδικα και το διαχειριστή πηγαίου κώδικα (VCS – Version Control System) [18]

3.3 Jupyter Notebook

Το Jupyter Notebook είναι μία εφαρμογή ανοιχτού πηγαίου-κώδικα, η οποία χρησιμοποιείται για το διαμοιρασμό εγγράφων τα οποία περιέχουν κώδικα, εξισώσεις, οπτικοποιήσεις και κείμενο. Το Jupyter Notebook προέρχεται από το IPython project και το όνομα προέρχεται από τις προγραμματιστικές γλώσσες τις οποίες υποστηρίζει : Julia, Python και R. Το Jupyter διανέμεται με τον πυρήνα IPython, ο οποίος επιτρέπει να γραφτούν προγράμματα σε Python.

Το Jupyter επιτρέπει να γραφτεί κώδικας σε ξεχωριστά κελιά, τα οποία εκτελούνται αυτόνομα. Αυτό επιτρέπει στον χρήστη να ελέγχει ένα συγκεκριμένο κομμάτι του κώδικα του, χωρίς να είναι απαραίτητο να τρέχει το πρόγραμμα από την αρχή. Αυτό είναι πολύ σημαντικό για την επιστήμη των δεδομένων και τις τεχνικές μηχανικών μάθησης, καθώς επιτρέπει στον χρήστη καλύτερη εκπαίδευση των αλγορίθμων του και την καλύτερη επίβλεψή τους, με αποτέλεσμα την ευκολότερη αποσφαλμάτωση.

Επομένως, για τη διαδικασία της εκπαίδευσης των ταξινομητών των αλγορίθμων, επίλεξα το Jupyter Notebook ως περιβάλλον για την ανάπτυξή τους. [19]

3.4 Λογισμικά Συλλογής Δεδομένων

Προκειμένου να συλλεξω τα δεδομένα από το Twitter και το Investing.com, αλλά και τις μεταβολές των τιμών των μετοχών ανά ημέρα, αξιοποίησα τρία (διαφορετικά) λογισμικά ανίχνευσης δεδομένων στο διαδίκτυο (crawlers). Στην ουσία, τα λογισμικά αυτά «διαβάζουν» ιστοσελίδες και συλλέγει πληροφορίες, ανάλογα με το πως έχει προγραμματιστεί από τον χρήστη, όπως για π.χ. κείμενα, e-mail, RSS feeds. Οι πιο γνωστοί ανιχνευτές διαδικτύου είναι οι scapy [20], Apache Nutch [21] και Heritrix. [22]

Επιπρόσθετα, για τη συλλογή των άρθρων και των τιμών μετοχών, χρησιμοποιήθηκε και το εργαλείο XPath, το οποίο βοηθάει στην άντληση των δεδομένων, συλλέγοντας τα δεδομένα που ο περιηγητής χρειάζεται να γνωρίζει, όπως π.χ. ο ιστότοπος ενός άρθρου, ο τίτλος του και η ημερομηνία έκδοσής του. [23]

3.5 Λεξικά Επεξεργασίας Φυσικής Γλώσσας

Βασική μέθοδος επεξεργασίας για την ανάλυση συναισθήματος οποτελεί η χρήση λεξικών για την επεξεργασία φυσικής γλώσσας (ΕΦΓ – NLP, Natural Language Processing). Για να γίνει κατανοητή η έννοια αυτών των λεξικών, πρέπει να γίνει μία διάκριση. Δεν αναφερόμαστε σε λεξικά με τη γνωστή έννοια των εξηγήσεων μια φυσική γλώσσας (π.χ. Ελληνικά) ή τα μεταφραστικά λεξικά (ή δίγλωσσα - π.χ. Ελληνοαγγλικό).

Τα λεξικά τα οποία χρησιμοποιούνται για την ανάλυση συναισθήματος αποτελούνται από ένα σύνολο με λέξεων και ετικετών, οι οποίες καθορίζουν το συναισθηματικό τόνο της κάθε λέξης. Πρόκειται για λίστες λέξεων οι οποίες έχουν χαρακτηριστεί με μία βαθμολογία (συνήθως +5, -5) και μπορούν να χρησιμοποιηθούν για να προσδώσουν μία βαθμολογία σε μία ολόκληρη πρόταση. Στη συνέχεια, αυτή η βαθμολογία μπορεί να χαρακτηρίσει το συναισθηματικό τόνο της πρότασης, αφού για παράδειγμα η χρήση αρνητικά βαθμολογημένων λέξεων σημαίνει ότι κατά πάσα πιθανότητα η πρόταση στο σύνολό θα εννοεί κάτι το αρνητικό. Αντίθετα, η χρήση θετικά βαθμολογημένων λέξεων γίνεται στην περίπτωση που η πρόταση στο σύνολό της εννοεί κάτι το θετικό. [24]

Στον τομέα της ανάλυσης συναισθήματος υπάρχουν πολλά λεξικά που μπορούν να χρησιμοποιηθούν. Η διαφορά τους έγκειται στον τομέα για τον οποίο προορίζονται (πχ. λεξικά γενικής χρήσης, οικονομικά λεξικά, πολιτικά κτλ.) και στο πλήθος των λημμάτων που περιέχουν.

Τα λεξικά που χρησιμοποιήθηκαν ήταν τρία (3): 1) Bing Liu Lexicon, 2) Loughran McDonald Lexicon, 3) Harvard IV-4 Lexicon.

3.6 Επεξεργασία Φυσικής Γλώσσας με τον αλγόριμθο ανάθεσης ετικετών

Ο Μέρος-Του-Λόγου αλγόριθμος ανάθεσης ετικετών (Part-Of-Speech - POS Tagger) είναι ένα λογισμικό το οποίο δέχεται ως είσοδο ένα κομμάτι κειμένου σε μία γλώσσα και αναθέτει ένα μέρος του λόγου σε κάθε μία λέξη του κειμένου, πχ ουσιαστικά, ρήματα, επίθετα κτλ. Το συγκεκριμένο λογισμικό είναι γραμμένο σε Python, και συγκεκριμένα γίνεται εισαγωγή της βιβλιοθήκης NLTK, παρόλα αυτά υπάρχουν εκδόσεις και σε κάποιες άλλες γλώσσες. Έχει δοθεί ιδιαίτερη έμφαση στην ταχύτητα, την ευκολία στην χρήση και την διαδικασία εκτέλεσης. Η τελευταίες εκδόσεις του λογισμικού, περιλαμβάνουν τρία μοντέλα για την Αγγλική γλώσσα, και από ένα για την Αραβική ,την Κινέζικη, την Γαλλική και την Γερμανική γλώσσα. Παρόλα αυτά ο αλγόριθμος ετικετών μπορεί να εκπαιδευτεί σε οποιαδήποτε φυσική γλώσσα με την κατάλληλη συλλογή δεδομένων.

Ο αλγόριθμος ανάθεσης μερών του λόγου αναπτύχθηκε από την ομάδα Natural Processing Group του Πανεπιστημίου του Stanford [25]. Η ομάδα αποτελείται από καθηγητές, υποψήφιους

διδάκτορες, προγραμματιστές και φοιτητές, οι οποίοι εργάστηκαν για τη δημιουργία εργαλείων, τα οποία επιτρέπουν την επεξεργασία και την ανάλυση της φυσικής γλώσσας.
Το αντικείμενο εργασίας τους περιστρέφεται γύρω από την βασική έρευνα στην υπολογιστική γλωσσολογία για εφαρμογές στην τεχνολογία της ανθρώπινης γλώσσας και καλύπτει τομείς όπως η κατανόηση προτάσεων, η αυτόματη απάντηση ερωτήσεων, η μηχανική μετάφραση, η ανάλυση συναισθήματος καθώς και εφαρμογές επεξεργασίας φυσικής γλώσσας στις ψηφιακές ανθρωπιστικές και υπολογιστικές κοινωνικές επιστήμες.

Το πρότυπο λογισμικό αναπτύχθηκε από την Kristina Toutanova, ενώ σε βελτιώσεις γύρω από την ταχύτητά του, την χρηστικότητα του και την πολυγλωσσική του υποστήριξη συνετέλεσαν και οι Dan Klein, Christopher Manning, William Morgan, Anna Rafferty, Michel Galley και John Bauer.

Οι βασικές έννοιες, ορισμοί και τρόπος λειτουργίας περιγράφεται στις δύο εργασίες “Enriching the Knowledge Sources Used in a Maximum Entropy Part-of-Speech Tagger” των Kristina Toutanova και Christopher D. Manning το 2000 στο “Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC-2000)” και “Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network” των Kristina Toutanova, Dan Klein, Christopher Manning, και Yoram Singer το 2003 στο HLT-NAACL 2003. [26]

3.7 Αλγόριθμοι Μηχανικής Μάθησης

Η εκτίμηση των χρηματιστηριακών μετοχών των αυτοκινητοβιομηχανιών, έγινε με τη βοήθεια αλγορίθμων μηχανικής μάθησης. Οι αλγόριθμοι αυτοί χρησιμοποιούν τους λεγόμενους ταξινομητές (classifiers), οι οποίοι αναλαμβάνουν το κομμάτι της εκπαίδευσης, μέσω των εκπαιδευμένων συνόλων δεδομένων (training Dataset) αλλά και το κομμάτι της δοκιμής των εκπαιδευμένων μοντέλων, μέσω του test dataset. Παρόλο που το δοκιμαστικό σύνολο δεδομένων είναι ανεξάρτητο από το εκπαιδευμένο, ακολουθεί την ίδια κατανομή πιθανότητας.

Οι ταξινομητές που χρησιμοποιήθηκαν ήταν τρεις: 1) Η Μέθοδος Κ-Κοντινότερων Γειτόνων (k-nearest neighbors algorithm - KNN), 2) η οικογένεια αλγορίθμων Απόφασης Δένδρου (Decision Tree) και 3) ο αλγόριθμος Τυχαίου Δάσους (Random Forest).

Κεφάλαιο 4: Συλλογή Δεδομένων

Το πρώτο κομμάτι της υλοποίησης αποτελείται από τη συλλογή των απαραίτητων άρθρων για κάθε αυτοκινητοβιομηχανία. Υπενθυμίζεται πως οι τρεις αυτοκινητοβιομηχανίες που επιλέγηασν είναι η Tesla, η Volkswagen και η Ford. Για την Tesla συλλέχθηκαν δεδομένα για το διάστημα από 10-6-2019 έως 18-2-2014, για την Volkswagen από 10-6-2019 έως 7-6-2014 ενώ για την Ford από 10-6-2019 έως 13-02-2014. Ο λόγος που επιλέχθηκαν αυτές οι τελικές ημερομηνίες για κάθε εταιρεία, έχει να κάνει την πυκνότητα των άρθρων και το πόσο ικανοποιητική είναι, δηλαδή η τελευταία ημερομηνία στην οποία βρέθηκε ικανοποιητικός όγκος δεδομένων, ενώ είναι ικανοποιητικό το γεγονός ότι για κάθε εταιρεία η χρονική περίοδος ανάμεσα στην οποία έγινε συλλογή των δεδομένων, είναι από πέντε χρόνια και πάνω.

Οι λόγοι που επηλέχθηκαν αυτές τις εταιρείες είναι κυρίως δύο. Αρχικά, αποτελούν αυτοκινητοβιομηχανίες/κολοσσούς στο χώρο της αυτοκίνησης, ενώ είναι και καινοτόμες αυτοκινητοβιομηχανίες στο κομμάτι της παραγωγής των ηλεκτροκίνητων οχημάτων παγκοσμίως. [27]

Όπως αναφέρθηκε και στο κομμάτι των ορισμών, η συλλογή των δεδομένων πραγματοποιήθηκε με τη βοήθεια των web crawlers ή αλλιώς προγράμματα ανίχνευσης του διαδικτύου.

Προτού ξεκινήσει η ανάλυση των λογισμικών, χρειάζεται να γίνει μία μικρή διευκρίνιση. Πολλές φορές, σε ορισμούς συναρτήσεων, θα συναντήσουμε τη δεσμευμένη λέξη-κλειδί ***self.*** **Η λέξη αυτή αναφέρεται στο αντικείμενο εκείνο το οποίο καλεί τις συναρτήσεις της κλάσης του.** [28] Για παράδειγμα έχουμε:

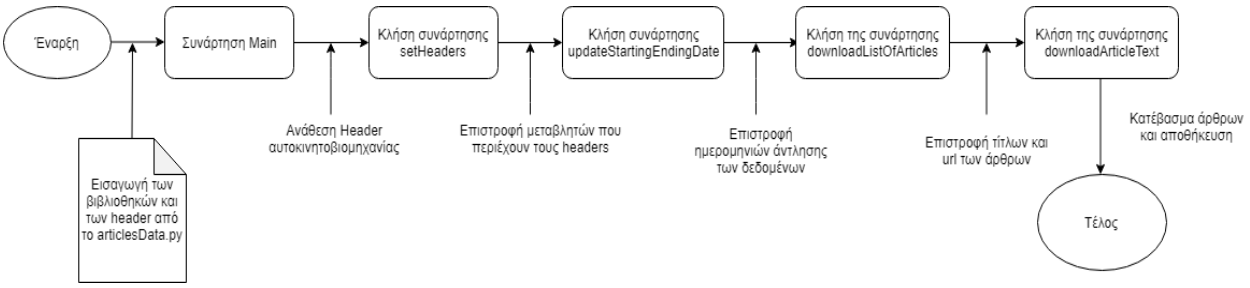
```
class SomeClass:
    def __init__(self):
        self.arr = []
        #All SomeClass objects will have an array arr by default

    def insert_to_arr(self, value):
        self.arr.append(value)

if __name__ == “__main__”:


obj1 = SomeClass()a
obj2 = SomeClass()
obj1.insert_to_arr(6)
```

Επομένως, η λέξη ***self,*** θα δείξει με τη σειρά πρώτα στο αντικείμενο ***obj1***, μετά στο ***obj2*** και τέλος στο ***obj1*** πάλι.

4.1 Λογισμικό για τη συλλογή άρθρων από το Investing.com

Το πρώτο λογισμικό που θα αναλυθεί, είναι αυτό της εξαγωγής άρθρων από το οικονομικό portal Investing.com. Πρόκειται για έναν ανιχνευτή διαδικτύου, ο οποίος αποτελείται από τρία προγράμματα python τα οποία επικοινωνούν μεταξύ τους. Το πρόγραμμα στέλνει POST request στην πλατφόρφα του Investing, ζητώντας τα ιστορικά δεδομένα για κάθε εταιρεία.

Παρακάτω παρουσιάζονται τα τρία πηγαία αρχεία ***Articles.py***, ***articlesData.py*** και ***sources.py***, με τα οποία γίνεται η συλλογή των άρθρων από το Investing.com.



Εικόνα 2 - Διάγραμμα ροής του αλγορίθμου εξαγωγής των άρθρων

Articles.py

Στις πρώτες εντολές του συγκεκριμένου προγράμματος, παρατηρούμε ότι εισάγονται οι βιβλιοθήκες python που θα μας χρειαστούν και θα χρησιμοποιήσουμε, όπως η lxml, η οποία βοηθάει στην επεξεργασία σελίδων XML και HTML σε γλώσσα python και η βιβλιοθήκη Pandas, η οποία προσφέρει εργαλεία για ανάλυση δεδομένων.

Η επικοινωνία του προγράμματος αυτού με τα υπόλοιπα δύο (2) γίνεται μέσα από τις εντολές ***from articles.articlesData import * και from articles.sources import *,*** όπου το πρόγραμμα κάνει εισαγωγή όλων των αντικειμένων από τα δύο προγράμματα και αυτό φαίνεται από το import *, που δίνει εντολή στο πρόγραμμα να εισάγει όλες τις διαθέσιμες συναρτήσεις και κλάσεις από το αρχείο στο οποίο αναφέρεται. Η πρώτη λέξη μετά τη δεσμευμένη λέξη from,

υποδεικνύει το φάκελο προέλευσης του προγράμματος (articles) και η δεύτερη μετά την τελεία, υποδεικνύει το όνομα του προγράμματος που εισάγεται (articlesData, sources)

Αμέσως μετά στέλνονται οι header παράμετροι http, όπου περιέχουν τα στοιχεία του User-Agent που χρησιμοποιεί ο Web crawler για να επικοινωνήσει με τη Διεπαφή Προγραμματισμού Εφαρμογών του Investing.com, ενώ επίσης περιέχεται και η αναφερόμενη πλατφόρμα (www.Investing.com), μέσω της **εντολής *'referer': "https://www.Investing.com*"** . Τα στοιχεία του U-A περιέχονται στο παρακάτω string: ***'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.86 Safari/537.36'***.

Main συνάρτηση του προγράμματος

Είναι ουσιαστικά ο κύριος κορμός του αλγορίθμου, μέσα από την οποία ξεκινάει η λειτουργία του προγράμματος και δίνεται το έναυσμα για τη λήψη των άρθρων. Οι πληροφορίες των headers που εισάγονται από το πρόγραμμα ***articlesData.py,*** αποθηκεύονται στη μεταβλητή ***indicator.*** Ο πίνακας με τους headers για κάθε αυτοκινητοβιομηχανία αποθηκεύεται στις μεταβλητές ***tesla_Investing, ford_Investing και vw_Investing,*** οπότε ο ***indicator*** παίρνει τη μορφή ***indicator = tesla_Investing*** ή ***indicator = ford_Investing*** ή ***indicator = vw_Investing.*** Μετά τον ορισμό του δείκτη (indicator), ξεκινάει μία σειρά από βήματα τα οποία οδηγούν στη λήψη των άρθρων που ο χρήστης επιθυμεί, ανάλογα με την ημερομηνία που ορίζει.

Κύρια κλάση *class ArticlesData()*

Αποτελεί και την μοναδική κλάση του αλγορίθμου, μέσα στην οποία υπάρχουν ορισμένες όλες οι συναρτήσεις. Καλείται αρχικά από τη συνάρτηση ***main*** του αλγορίθμου, με την εντολή ***ad = ArticlesData(indicator)***, όπου η μεταβλητή ad αρχικοποιείται ως αντικείμενο κλάσης ArticlesData και η μεταβλητή indicator περιέχει τους headers της εταιρείας που θέλουμε να εξάγουμε τα δεδομένα (θα αναλυθεί παρακάτω).

Συνάρτηση *def __init__(self, indicatorData)*

Είναι η μέθοδος constructor – συνάρτηση κατασκευής, όπου αναλαμβάνει την αρχικοποίηση και τον ορισμό όλων των μεταβλητών που θα χρησιμοποιηθούν στην κλάση.

Συνάρτηση *def setHeaders(self, headers)*

Συνάρτηση που έχει ως όρισμα τους headers που γίνονται εισαγωγή από τη συνάρτηση ***articlesData.py*** (θα αναλυθεί παρακάτω). Γίνεται ορισμός των headers, οι οποίοι θα χρησιμοποιηθούν αργότερα στην κλάση προκειμένου να αντληθούν τα στοιχεία που επιθυμούμε. Καλείται από τη συνάρτηση main μέσω της εντολής ***ad.setHeaders(headers)***

Συνάρτηση *def updateStartingEndingData(self, startingDate, endingDate)*

Όπως υποδεικνύει και το όνομα της συνάρτησης, αποτελεί τη συνάρτηση που ορίζονται οι επιθυμητές ημερομηνίες, ανάμεσα στις οποίες θα εξαχθούν τα άρθρα. Δέχεται ως ορίσματα τις ημερομηνίες έναρξης και λήξης, ενώ καλείται από τη συνάρτηση ***main***, μέσω της εντολής ***ad.updateStartingEndingData(datetime.data(YYYY-MM-DD), datetime.date(YYYY,MM,DD)).***

Συνάρτηση *def downloadListOfArticlesRepeatedly(self)*

Σκοπός της συγκεκριμένης συνάρτησης είναι η επαναλαμβανόμενη διαδικασία της αναζήτησης άρθρων, ανάλογα με την επιθυμητή ημερομηνία που έχει εισαχθεί από τον χρήστη. Η συνάρτηση αυτή καλεί μέσα από μία επανάληψη τη συνάρτηση ***downloadListOfArticles*** (θα αναλυθεί ακριβώς παρακάτω), η οποία ευθύνεται κιόλας για τη συλλογή των τίτλων των άρθρων. Στέλνει το ***'url_api'*** της πλατφόρμας στην παραπάνω συνάρτηση, το οποίο εισάγεται από τους headers του προγράμματος ***articlesData.py***, ενώ επίσης στέλνει και έναν μετρητή ***page_counter***, ο οποίος μετράει τις σελίδες που προσπελάστηκαν κατά τη αναζήτηση της λίστας των άρθρων. Η επαναληπτικότητα της συνάρτησης σταματάει όταν ο scrapper προσπελάσει τη σελίδα με την πιο παλαιά ημερομηνία που έχει ορίσει ο χρήστης. Η συνάρτηση δημιουργεί τελικώς μία λίστα, η οποία περιέχει μία λίστα με τα άρθρα και url που βρέθηκαν, τα οποία θα εξαχθούν στη συνέχεια από τη συνάρτηση ***downloadArticleText*** (θα αναλυθεί παρακάτω). Καλείται από τη ***main*** συνάρτηση του αλγορίθμου, μέσα από την εντολή ***ad.downloadListOfArticlesRepeatedly()***

Συνάρτηση *def downloadListOfArticles(self, API_url)*

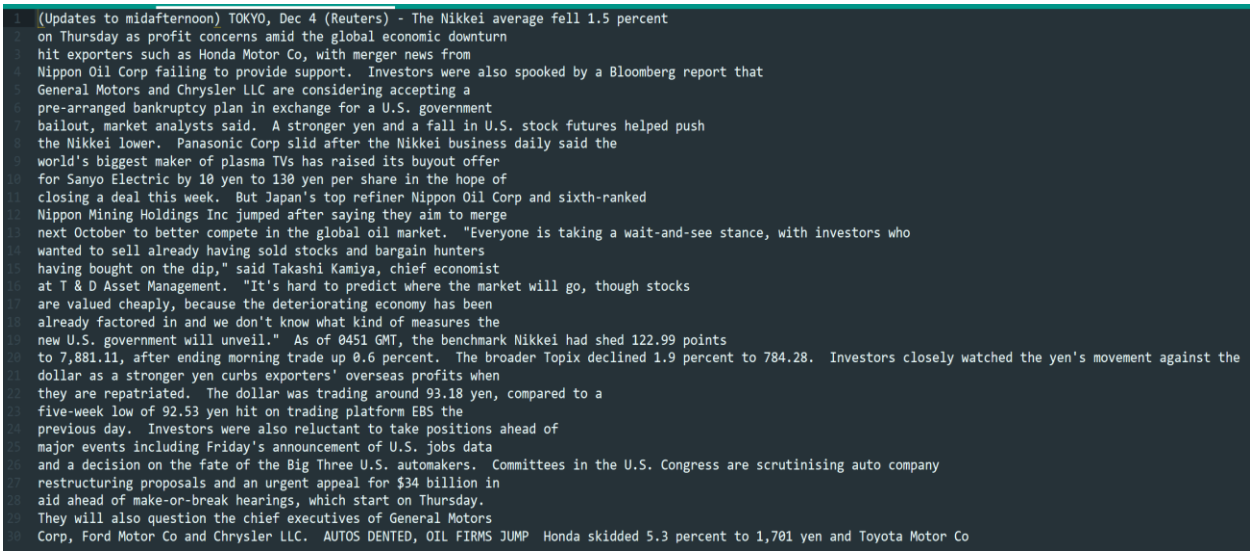
Η συγκεκριμένη συνάρτηση αποτελεί τη συνάρτηση η οποία κατεβάζει τους τίτλους των επιθυμητών άρθρων και τα url που περιέχουν το κείμενό τους, σύμφωνα με τις ημερομηνίες που ορίστηκαν από τον χρήστη. Καλείται επαναληπτικά από τη συνάρτηση ***downloadListOfArticlesRepeatedly,*** στην οποία στέλνει τους τίτλους των άρθρων και τα url που περιέχουν το κείμενο των άρθρων, και έτσι η δεύτερη δημιουργεί τη λίστα με τα άρθρα που πρόκειται να εξαχθούν αργότερα από τη συνάρτηση ***downloadArticleText*** (θα αναλυθεί ακριβώς παρακάτω)**.** Δέχεται ως παραμέτρους το '***url_api'*** και το μετρητή των σελίδων (***page_counter***) και με τη βοήθεια της βιβλιοθήκης lxml που γίνεται εισαγωγή στην αρχή του αλγορίθμου, αποθηκεύει τους τίτλους των άρθρων, τα links των άρθρων και τις ημερομηνίες. Κομβικό ρόλο παίζει η χρησιμοποίηση των xpath html links, τα οποία εισάγονται από την ***articlesData.py*** και περιέχουν τις πληροφορίες σχετικά με το url, τον τίτλο και την ημερομηνία του άρθρου ('***xpath_articles_title'***, '***xpath_articles_link'***, '***xpath_articles_date'***). Μετά την ολοκλήρωση της συνάρτησης, υπάρχει μία προαιρετική εμφάνιση της λίστας από τη συνάρτηση ***printListOfArticles***.

Συνάρτηση *def downloadArticleText(self)*

Αποτελεί τη συνάρτηση η οποία κατεβάζει το περιεχόμενο των άρθρων, που αποθηκεύτηκαν στη λίστα της ***downloadListOfArticlesRepeatedly.*** Αρχικά, ελέγχει την πηγή των άρθρων που έχει ορισθεί από τους headers (ο αλγόριθμος λειτουργεί και για άλλες οικονομικές πύλες, όπως το reuters) και με το πέρας του ελέγχου, για κάθε άρθρο μέσα στη λίστα, η συνάρτηση αυτή επισκέπτεται το url που το περιέχει και κατεβάζει το κείμενό του. Στη συνέχεια, ακολουθείται η διαδικασία της αποθήκευσης όλων των άρθρων που βρέθηκαν, με το κάλεσμα της συνάρτησης ***saveArticle*** (θα αναλυθεί ακριβώς από κάτω). Στην ***saveArticle*** στέλνονται το όνομα του φακέλου προορισμού-αποθήκευσης των άρθρων (εφόσον υπάρχει), το όνομα του άρθρου, όπου είναι η ημερομηνία και ο τίτλος, όπως επίσης και το κείμενο που θα αποθηκευτεί**.** Η συνάρτηση καλείται από ***main*** συνάρτηση, με την εντολή ***ad.downloadArticleText().***

Συνάρτηση *saveArticle(self, directory, filename, text)*

Όπως προαναφέρθηκε, η συγκεκριμένη συνάρτηση είναι υπεύθυνη για την αποθήκευση όλων άρθρων. Η αποθήκευση γίνεται ξεχωριστά για κάθε άρθρο, το οποίο αποθηκεύεται σε μορφή .txt. Δέχεται ως ορίσματα το φάκελο προορισμού (εάν υπάρχει, αλλιώς δημιουργείται μέσα στη συνάρτηση), το όνομα του αρχείου και το κείμενο, αποθηκευμένο σε κωδικοποίηση utf-8. Παράδειγμα τελικού αποθηκευμένου αρχείου αποτελεί το αρχείο ‘2008-12-04-Nikkei falls 1.5 pct as yen, economy hit exporters.txt’, το οποίο είναι άρθρο που δημοσιεύτηκε στις τέσσερις Δεκεμβρίου 2008, με όνομα τον τίτλο του άρθρου.



Εικόνα 3 - Στιγμιότυπο από το αρχείο ‘2008-12-04-Nikkei falls 1.5 pct as yen, economy hit exporters.txt’

Παρακάτω ακολουθεί ο πηγαίος κώδικας του προγράμματος ***articles.py***

```
from lxml.etree import fromstring
from lxml import html
import os, datetime, calendar, requests
import mechanize

import pandas as pd

from articles.articlesData import *
from articles.sources import *

# set https header parameters
headers = {
   'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.86 Safari/537.36',
   'referer': "https://www.Investing.com",
}

class ArticlesData():

   def __init__(self, indicatorData):
      self.parameters = {}
      self.articles = {}
      self.indicatorData = indicatorData

      self.article_counter = 0
      self.false_flag = 0


   #set https header for request
   def setHeaders(self, headers):
      self.headers = headers


   #desired time period from/to
   def updateStartingEndingDate(self, startingDate, endingDate):
      self.parameters['st_date'] = startingDate
      self.parameters['end_date'] = endingDate


   #download article titles, urls and dates for desired dates
   def downloadListOfArticles(self, API_url):
      #creates a dictionary included all the articles in the desired time
period
      if DEBUG:
         print("[+] Downloading data from: " + str(API_url))

      page = requests.get(API_url, headers=self.headers)
      tree = html.fromstring(page.content)
```

```
      article_titles = tree.xpath(self.indicatorData['xpath_articles'] +
self.indicatorData['xpath_articles_title'])
      article_title_links = tree.xpath(self.indicatorData['xpath_articles']
+ self.indicatorData['xpath_articles_link'])
      article_dates =  tree.xpath(self.indicatorData['xpath_articles'] +
self.indicatorData['xpath_articles_date'])


try:
         article_titles.remove(" ")
      except ValueError:
         pass



      #the problem is caused because of the sponsored article, which does
not have a date
      #the sponsored is deleted
      if len(article_titles) > len(article_dates):
         del(article_titles[3])
         del(article_title_links[3])

      # there is also an add sometimes in the 6th position after the
sponsored
      if len(article_titles) > len(article_dates):
         del(article_titles[5])

      # there is also an add sometimes in the 8th position after the
sponsored
      if len(article_titles) > len(article_dates):
         del(article_titles[7])

      # if more than two adds and one sponsored is appeared delete the
first
      while (len(article_titles) > len(article_dates)):
         del(article_titles[0])



      #if ~40 articles are outdated stop the procedure
      for i in range(len(article_titles)):
         #check if 'date' is formet like "Dec 22, 2017" or "11 hours
before" or "am" etc
         #if it is set today for date
         if "hour" in article_dates[i] or "minute" in article_dates[i] or
"second" in article_dates[i] or "am" in article_dates[i] or "pm" in
article_dates[i]:
            article_dates[i] = datetime.date.today()
            article_date = article_dates[i]
         else:
            if self.indicatorData['name_API'] == 'reuters':
```

```
                #find differencies in dates #just a comma
                article_date =
datetime.datetime.strptime(article_dates[i].replace('\xa0-\xa0',''), "%b %d
%Y").date()
            elif self.indicatorData['name_API'] == 'Investing':
                article_date =
datetime.datetime.strptime(article_dates[i].replace('\xa0-\xa0',''), "%b
%d, %Y").date()
#         print(article_date)

          #check if article date is in desired date period
          if self.parameters['st_date'] <= article_date and article_date <=
self.parameters['end_date']:
             article_title =
article_titles[i].replace("\n","").replace("\t","")
             if self.filterArticle(article_title):
                self.articles[self.article_counter] = {'article_title' :
article_title, 'article_title_link' : article_title_links[i],
'article_date' : article_date}
                self.article_counter += 1
          else:
             self.false_flag += 1
             if self.false_flag >= len(article_titles): #almost 40
                return False
       return True


   #search also on next pages for articles in the desired time period
   def downloadListOfArticlesRepeatedly(self):
      #creates a dictionary included all the articles in the desired time
period
       page_counter = 274
       while self.downloadListOfArticles(self.indicatorData['url_API'] +
str(page_counter)):
          page_counter += 1
       if DEBUG:
          print("[+] Scraped " + str(page_counter) + " pages - Articles
found: " + str(len(self.articles)))


   #download all articles from article list dictionary
   def downloadArticleText(self):
      #create directory "/results" if doesn't exists
      if not os.path.exists("results"):
         os.makedirs("results")

      #for every article in the dictionary, visit url and scrape text
      for key in sorted(self.articles.keys()):
         #if the artcile is IN the website
         if "http" not in self.articles[key]['article_title_link']:
```

```
            if self.indicatorData['name_API'] == 'reuters':
               article_url = 'https://www.reuters.com' +
self.articles[key]['article_title_link']
            elif self.indicatorData['name_API'] == 'Investing':
               article_url = 'https://www.Investing.com' +
self.articles[key]['article_title_link']
         #or is from another source
         else:
            article_url = self.articles[key]['article_title_link']
         if DEBUG:
            print("[+] Downloading article from: " + article_url)

         article = requests.get(article_url, headers=headers)
         tree = html.fromstring(article.content)
         #check for every possible source (file: articles/sources.py)
         for source in sources:
            article_text = '
'.join(tree.xpath(sources[source]['xpath_article']))
            if article_text is not "":
               if DEBUG:
                  print("[+] Source: " + str(source))
                  print("[+] Saving data: results/" +
str(self.articles[key]['article_date']) + "/" +
self.articles[key]['article_title'] + ".txt")
               self.saveArticle("results/" +
str(self.articles[key]['article_date']),
str(self.articles[key]['article_date']) +"-" +
self.articles[key]['article_title'].replace("/"," ").replace("<","
").replace(">"," ").replace(":"," ").replace('"', " ").replace("\"" , "
").replace("|", " ").replace("?", " ").replace("*", " ") + ".txt",
article_text)


   #save article in .txt format
   def saveArticle(self, directory, filename, text):
      try:
         file = open(directory + "/" + filename, "w", encoding='utf-8')
      except FileNotFoundError:
         if not os.path.exists(directory):
            os.makedirs(directory)
         file = open(directory + "/" + filename, "w", encoding='utf-8')
      file.write(text)
      file.close()


   #print article dictionary
   def printListOfArticles(self):
      for key in self.articles:
```

```
         print("Title: " + self.articles[key]['article_title'] + "\nURL: "
+ self.articles[key]['article_title_link'] +  "\nDate: " +
str(self.articles[key]['article_date']) + "\n")

if __name__ == "__main__":

   DEBUG = True

   indicator = vw_Investing

   ad = ArticlesData(indicator)
   ad.setHeaders(headers)
   #DATE FORMAT: YYYY, MM, DD
   ad.updateStartingEndingDate(datetime.date(2014, 7, 6),
datetime.date(2019, 5, 15))
   ad.downloadListOfArticlesRepeatedly()
#  ad.printListOfArticles()

   ad.downloadArticleText()
   print("[+] DONE")
```

Κώδικας 1 - Κύριο πρόγραμμα Articles.py, με το οποίο συλλέγονται τα δεδομένα.

ArticlesData.py

Πρόκειται για το πρόγραμμα που περιέχει τους headers για τις τρεις (3) αυτοκινητοβιομηχανίες. Αναλυτικά, περιέχει τις παρακάτω πληροφορίες αποθηκευμένες σε μορφή λεξικού (dictionary):[1]

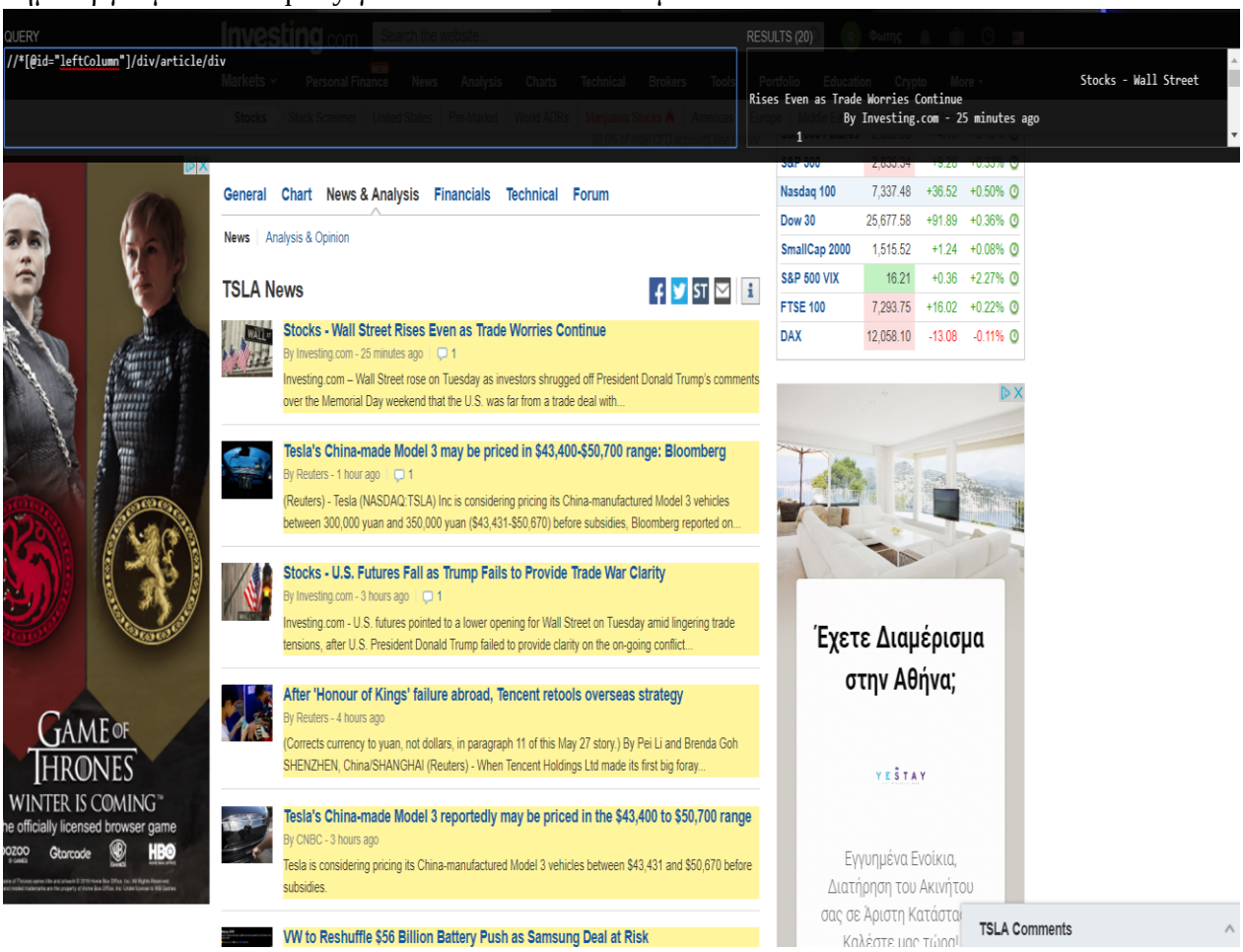
- Το όνομα του λεξικού
- Το όνομα της διεπαφής της εταιρείας στην πλατφόρμα του Investing ***('name_API'***)
- Τον HTML κώδικα που περιέχει το κείμενο του άρθρου ('***xpath_articles***')
- Τον HTML κώδικα που περιέχει τον τίτλο του άρθρου ('***xpath_articles_title'***)
- Τον HTML κώδικα που περιέχει το link του άρθρου ('***xpath_articles_link'***)
- Τον HTML κώδικα που περιέχει την ημερομηνία που είναι γραμμένο το άρθρο ('***xpath_articles_date***')

Τα αντικείμενα του λεξικού είναι προσπελάσιμα μέσω του index (δείκτη) του κάθε αντικειμένου. Για παράδειγμα, για να εμφανίσουμε την ημερομηνία δημιουργίας ενός άρθρου της tesla, αρκεί να δώσουμε την εντολή print(tesla_Investing['xpath_articles_date'].

Στην παρακάτω εικόνα, η οποία είναι από την ιστοσελίδα του Investing, παρατίθεται ένα παράδειγμα χρήσης του εργαλείου xpath, προκειμένου να 'μαρκάρουμε' το περιεχόμενο των άρθρων μέσω του HTML κώδικα. Αφού το ενεργοποιήσουμε στο φυλλομετρητή που χρησιμοποιούμε (πάνε μέρος της οθόνης), κρατάμε πατημένο το κουμπί shift και μετακινούμε τον

[1] Ένα λεξικό είναι μία μορφή συλλογής δεδομένων σε python, η οποία είναι μη-καθορισμένη, τροποποιήσιμη και υπό μορφή δεικτών. [53]

κέρσορα του ποντικιού επάνω στο κείμενο που θέλουμε να αποσπάσουμε. Το εργαλείο θα δημιουργήσει αυτόματα το query του κειμένου που μαρκάραμε (πάνω αριστερά), ενώ επίσης θα εμφανίσει και το αποτέλεσμα του συγκεκριμένου query (πανω δεξιά). Επιπλέον, υπογραμμίζεται και το περιεχόμενο του κειμένου που επιλέξαμε (στο κέντρο της ιστοσελίδας). Στη συνέχεια, αντιγράφουμε τον HTML κώδικα που έχουμε στο query και τον μεταφέρουμε στον header της εταιρείας που επιλέξαμε (στη συγκεκριμένη περίπτωση είναι η Tesla). Στην προκειμένη περίπτωση, θα έχουμε: **'xpath_articles_title' : '/a[@class="title"]/text()'.** Με ανάλογο τρόπο, δημιουργούμε και τα query για τα υπόλοιπα αντικείμενα των headers.



Εικόνα 4 - Στιγμιότυπο από τη χρήση του εργαλείου xPath, προκειμένου να αποσπάσουμε τους HTML κώδικες για τους headers.

Παρακάτω ακολουθεί ο πηγαίος κώδικας του προγράμματος ***articlesData.py***

```
'''
INVESTING.COM
*   sometimes 'articleItem' is 'articleItem ' this patch corrects this ->
node()
*                sometimes         '/span[@class='articleDetails']'        is
'/div[@class='articleDetails'] ' this patch corrects this -> node()
REUTERS.COM
```

```
*   sometimes 'news-headline-list' is 'news-headline-list   ' this patch
corrects this -> node()
*  sometimes 'story' is 'story ' this patch corrects this -> node()
'''

#

#https://www.Investing.com/equities/volkswagen-vz-news/
vw_Investing = {
   'name_API' : 'Investing',
   'url_API' : 'https://www.Investing.com/equities/volkswagen-vz-news/',
   'xpath_articles' : '//*[@id="leftColumn"]/div/article/div',
   'xpath_articles_title' : '/a[@class="title"]/text()',
   'xpath_articles_link' : '/a[@class="title"]//@href',
   'xpath_articles_date' : '/node()/span[@class="date"]/text()',

}

#https://www.Investing.com/equities/tesla-motors-news/
tesla_Investing = {
   'name_API' : 'Investing',
   'url_API' : 'https://www.Investing.com/equities/tesla-motors-news/',
   'xpath_articles' : '//*[@id="leftColumn"]/div/article/div',
   'xpath_articles_title' : '/a[@class="title"]/text()',
   'xpath_articles_link' : '/a[@class="title"]//@href',
   'xpath_articles_date' : '/node()/span[@class="date"]/text()',
}

#https://www.Investing.com/equities/ford-motor-co-news/
ford_Investing = {
   'name_API' : 'Investing',
   'url_API' : 'https://www.Investing.com/equities/ford-motor-co-news/',
   'xpath_articles' : '//*[@id="leftColumn"]/div/article/div',
   'xpath_articles_title' : '/a[@class="title"]/text()',
   'xpath_articles_link' : '/a[@class="title"]//@href',
   'xpath_articles_date' : '/node()/span[@class="date"]/text()',

}
```

Κώδικας 2: Πηγαίο πρόγραμμα το οποίο περιέχει τους βασικούς headers με τις πληροφορίες που χρειαζόμαστε για το API του Investing.com.

sources.py

Στο τελευταίο πρόγραμμα του πρώτου λογισμικού, έχουμε τις πηγές από τις οποίες αντλούμε τα άρθρα.

```
sources = {
'Investing_v0'              :              {'xpath_article'              :
'//section[@id="leftColumn"]/div[@class="WYSIWYG articlePage"]//text()'},
   'Investing_v1' : {'xpath_article' : '//div[@class="arial_14 clear WYSIWYG
newsPage"]/p/text()'},
   'Investing_v2'             :             {'xpath_article'             :
'//section[@id="leftColumn"]/node()/p/text()'},
}
```

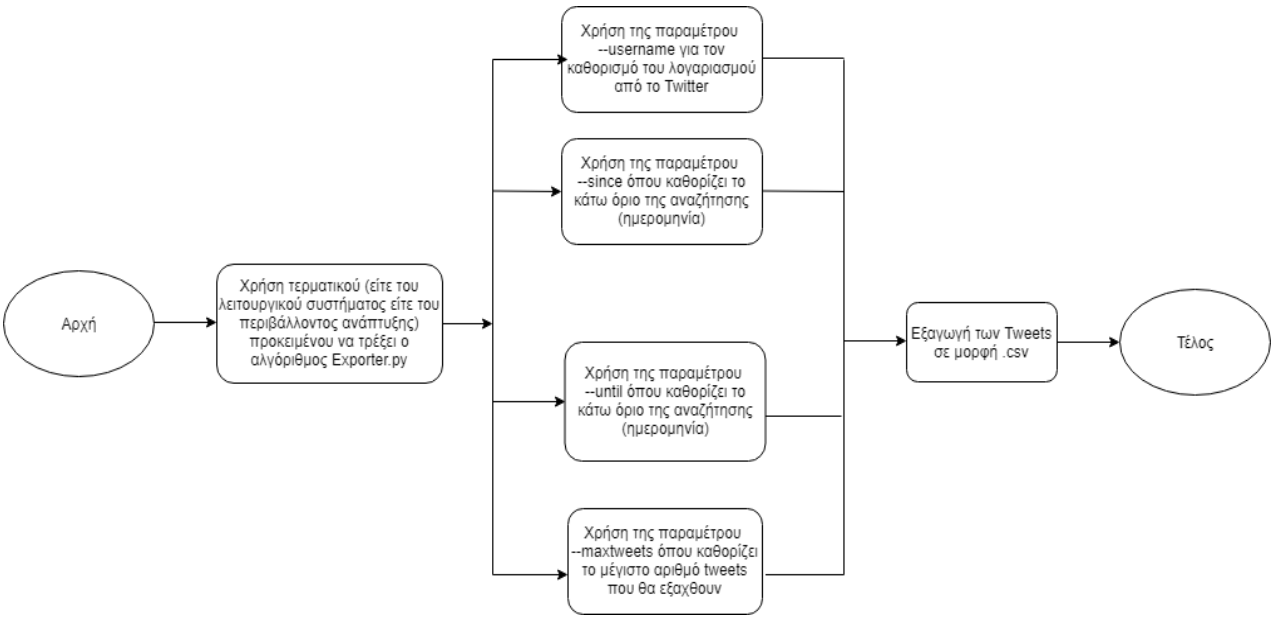
Κώδικας 3 - Πηγαίο πρόγραμμα το οποίο περιέχει τους headers επικοινωνίας με το mainArticleBody των άρθρων, σε βασικά οικονομικά portal. Η σάρωση γίνεται με κομμάτι HTML κώδικα, το οποίο βρίσκεται μέσω του εργαλείου xPath. [29]

4.2 Λογισμικό για τη συλλογή δεδομένων από το twitter

Στο σημείο αυτό, θα γίνει ανάλυση του λογισμικού με το οποίο εξήγαγα τα δεδομένα από το Twitter. Πρόκειται για έναν αλγόριθμο του Jefferson Henrique [30], με τον οποίο λύνεται ένα αρκετά μεγάλο πρόβλημα που προκαλεί η επίσημη Διεπαφή Προγραμματισμού Εφαρμογών του Twitter.

Συγκεκριμένα, το Twitter δεν επιτρέπει την εξαγωγή tweet από λογαριασμούς, σε χρονικό διάστημα που υπερβαίνει τη μία εβδομάδα. Υπάρχουν εργαλεία τα οποία επιτρέπουν την πρόσβαση σε παλαιότερα tweets, όπως απαιτούν πληρωμή για την απόκτησή τους. Η ιδέα αυτού του αλγορίθμου βασίζεται στο γεγονός ότι όταν κάποιος μπαίνει στο Twitter και επιθυμεί να ψάξει παλαιότερα tweets, χρησιμοποιεί τον scroll loader προκειμένου να έχει πρόσβαση σε παλαιότερα tweets, όλα μέσα από ένα αλφαριθμητικό τύπου JSON[2]. Μέσα από μία επαναληπτική διαδικασία, ο συγκεκριμένος αλγόριθμος εκμεταλλεύεται στο έπακρο την αναζήτηση του Twitter στους φυλλομετρητές και μπορεί να αναζητήσει μέχρι και τα παλαιότερα αρχεία.

[2] Json – JavaScript Object Notation. Αποτελεί μία σύνταξη για αποθήκευση και ανταλλαγή δεδομένων. Είναι ουσιαστικά απλό κείμενο, γραμμένο σε μορφή JavaScript αντικειμένων. [29]



Εικόνα 5 - Διάγραμμα ροής του αλγορίθμου εξαγωγής των Tweets (Exporter.py)

Σύνθεση

- ***Tweet***: Κλάση η οποία παρέχει πληροφορίες για ένα συγκεκριμένο Tweet.
    - ***Id*** (αλφαριθμητικό)
    - ***Permalink*** (αλφαριθμητικό)
    - ***Username*** (αλφαριθμητικό)
    - ***Text*** (αλφαριθμητικό)
    - ***Date*** (ημερομηνία)
    - ***Retweets*** (ακέραιο)
    - ***Favorites*** (ακέραιο)
    - ***Mentions*** (αλφαριθμητικό)
    - ***Hashtags*** (αλφαριθμητικό***)***
    - ***Geo*** (αλφαριθμητικό)

- ***TweetManager***: Μία κλάση διαχειριστής, η οποία βοηθάει να εξαχθούν τα Tweets από την κλάση ***Tweet***.
    - ***getTweets (TwitterCriteria):*** Επιστρέφει τη λίστα με τα tweets, χρησιμοποιώντας ένα αντικείμενο TwitterCriteria.

- ***TwitterCriteria***: Μία συλλογή από παραμέτρους αναζήτησης, οι οποίες χρησιμοποιούνται μαζί με το ***TweetManager.***
    - ***username*** (αλφαριθμητικό): Ένα προαιρετικό και συγκεκριμένο όνομα χρήστη από έναν λογαριασμό Twitter, του οποίου τα tweet θέλουμε να εξάγουμε
    - ***since (***αλφαριθμητικ***ό*** **"**yyyy-mm-dd"): Το κάτω όριο της αναζήτησης

- ***until (***αλφαριθμητικό **"**yyyy-mm-dd"): Το άνω όριο της αναζήτησης
- ***querysearch*** (αλφαριθμητικό): Ένα κείμενο query το οποίο να ταιριάζει
- ***toptweets*** (δυαδικό): *Εάν είναι αληθές, μόνο τα καλύτερα tweets θα εξαχθούν*
- ***near*** (αλφαριθμητικό): Μία τοποθεσία ως αναφορά για να εξαχθούν τα tweets
- ***maxtweets*** (ακέραιο): Ένας μέγιστος αριθμός ο οποίος τίθεται σε ισχύ, προκειμένου να μην ξεπεραστεί ο συγκεκριμένος αριθμών εγγραφών.

- ***Exporter.py***: Ο αλγόριθμος ο οποίος εκμεταλλεύεται τις παραπάνω μεταβλητές και συναρτήσεις και αντλεί τα δεδομένα από το Twitter. Εξάγει τα αρχεία σε μορφή .csv. Τονίζεται πως οι παράμετροι της συνάρτησης TwitterCriteria μπορούν να συνδυαστούν μεταξύ τους και δεν είναι υποχρεωτική η χρήση όλων. Επομένως, εξαρτάται από τον χρήστη και τι επιθυμεί να εξάγει.

Η παρακάτω φωτογραφία, αποτελεί παράδειγμα χρήσης του αλγορίθμου, όπου χρησιμοποιούνται οι παράμετροι –***username***, --***since***, --***until***, --***maxtweets*** προκειμένου να εξαχθούν tweets από το λογαριασμό της Tesla, για ένα συγκεκριμένο χρονικό διάστημα (από 1-5-2019 μέχρι 28-1-2019 στο συγκεκριμένο παράδειγμα), θέτοντας μέγιστο αριθμό tweets τα εκατό (100). Το αρχείο αποθηκεύτηκε με το όνομα "tesla.csv". Για να ορισθούν οι παράμετροι της εξαγωγής, χρησιμοποιήθηκε το τερματικό του εργαλείου PyCharm.

```
C:\Users\fotis\Desktop\thesis\python_programms\tweets_crawler\GetOldTwee
ts-python-master>Exporter.py --username "tesla" --since 2019-5-1 --until
 2019-5-28 --maxtweets 100
Searching...

More 100 saved on file...

Done. Output file generated "tesla.csv".
```

Εικόνα 6 - Στιγμιότυπο από το τερματικό του PyCharm, το οποίο υποδεικνύει το κατέβασμα των Tweets από το λογαριασμό της Tesla.

Στο παρακάτω στιγμιότυπο, παρατηρούμε το αποτέλεσμα της παραπάνω διαδικασίας:

A	B	C	D	E	F	G	H	I	J	K	L	M	N
date	retweets	favorites	text	geo	mentions	hashtags	id	permalink					
5/26/2019	10	328	No										
5/26/2019	10	250	All Tesla models come standard with 'Fast N				1.13E+18	https://twitter.com/Tesla/status/1132685590950891520					
5/26/2019	21	330	Who needs advertising when you've got this				1.13E+18	https://twitter.com/Tesla/status/1132682626718883841					
5/26/2019	401	5172	We Interrupt Your Re	@			1.13E+18	https://twitter.com/Tesla/status/1132680674085703680					
5/24/2019	7	226	Ah										
5/24/2019	7	238	We may even have a square to spare. Let us				1.13E+18	https://twitter.com/Tesla/status/1131949529379495936					
5/24/2019	70	719	Actually we still have plenty of toilet paper										
5/24/2019	8	205	Pssst										
5/23/2019	8	248	But we even looked it up in the dictionaryht				1.13E+18	https://twitter.com/Tesla/status/1131651452005306368					
5/23/2019	4	222	Can't we be a car who can do both?				1.13E+18	https://twitter.com/Tesla/status/1131631725878996992					
5/23/2019	3	164	Precisely				1.13E+18	https://twitter.com/Tesla/status/1131629379421843457					
5/23/2019	21	365	Gas stations are definitely not S3XY				1.13E+18	https://twitter.com/Tesla/status/1131628964311515136					
5/23/2019	10	286	We've been doing over-the-air updates since 2012										
5/23/2019	5	296	Who us?				1.13E+18	https://twitter.com/Tesla/status/1131601406362656770					
5/23/2019	1176	16318	Hint: it mehttps://twitter.com/MotorTrend				1.13E+18	https://twitter.com/Tesla/status/1131600408080572416					
5/22/2019	5	151	Realistically there are probably many things				1.13E+18	https://twitter.com/Tesla/status/1131006101787357189					
5/22/2019	5	186	We got the frunk in the front				1.13E+18	https://twitter.com/Tesla/status/1131005306765373441					
5/22/2019	10	222	Electric pizza coming soon				1.13E+18	https://twitter.com/Tesla/status/1130976052786843648					
5/22/2019	5	197	Or your hammer for that matter				1.13E+18	https://twitter.com/Tesla/status/1130971367468224513					
5/22/2019	316	4228	Don't let your cowboy hat get in the way of				1.13E+18	https://twitter.com/Tesla/status/1130969408694673408					
5/18/2019	12	254	Name another car that has Romance Mode				1.13E+18	https://twitter.com/Tesla/status/1129853313833242624					
5/18/2019	6	113	We need Automatic Tissue Dispensing Mode				1.13E+18	https://twitter.com/Tesla/status/1129844219579961345					
5/18/2019	2	220	Suddenly we feel like the prettiest girl at the				1.13E+18	https://twitter.com/Tesla/status/1129842658480779264					
5/18/2019	7	278	Only if you buy us a really lovely corsage				1.13E+18	https://twitter.com/Tesla/status/1129837570039398401					
5/18/2019	3	129	Jayden Andrews for Prom King am I right?				1.13E+18	https://twitter.com/Tesla/status/1129837174252134400					
5/18/2019	4	82	We thought you'd never ask Alan				1.13E+18	https://twitter.com/Tesla/status/1129836678145642496					
5/18/2019	15	359	We wish someone would ask us to prom				1.13E+18	https://twitter.com/Tesla/status/1129832955193352192					
5/18/2019	398	4541	We're not crying										

Εικόνα 7 - Στιγμιότυπο από το αρχείο csv που κατέβηκε.

Εν συνεχεία, παρουσιάζεται ολοκληρωμένος ο υλοποιημένος κώδικας με τον οποίο γίνεται η συλλογή των δεδομένων από το Twitter:

```
# -*- coding: utf-8 -*-
import sys,getopt,datetime,codecs
if sys.version_info[0] < 3:
    import got
else:
    import got3 as got

def main(argv):

   if len(argv) == 0:
      print('You must pass some parameters. Use \"-h\" to help.')
      return

   if len(argv) == 1 and argv[0] == '-h':
      f = open('exporter_help_text.txt', 'r')
      print(f.read())
```

```
      f.close()

      return

   try:
      opts, args = getopt.getopt(argv, "", ("username=", "near=", "within=",
"since=", "until=", "querysearch=", "toptweets", "maxtweets=", "output="))

      tweetCriteria = got.manager.TweetCriteria()
      outputFileName = "tesla.csv"

      for opt,arg in opts:
         if opt == '--username':
            tweetCriteria.username = arg

         elif opt == '--since':
            tweetCriteria.since = arg

         elif opt == '--until':
            tweetCriteria.until = arg

         elif opt == '--querysearch':
            tweetCriteria.querySearch = arg

         elif opt == '--toptweets':
            tweetCriteria.topTweets = True

         elif opt == '--maxtweets':
            tweetCriteria.maxTweets = int(arg)

         elif opt == '--near':
            tweetCriteria.near = '"' + arg + '"'

         elif opt == '--within':
            tweetCriteria.within = '"' + arg + '"'

         elif opt == '--within':
            tweetCriteria.within = '"' + arg + '"'

         elif opt == '--output':
            outputFileName = arg

      outputFile = codecs.open(outputFileName, "w+", "utf-8")

      #outputFile.write('date;text;retweets;favorites')

outputFile.write('date;retweets;favorites;text;geo;mentions;hashtags;id;per
malink')

      print('Searching...\n')
```

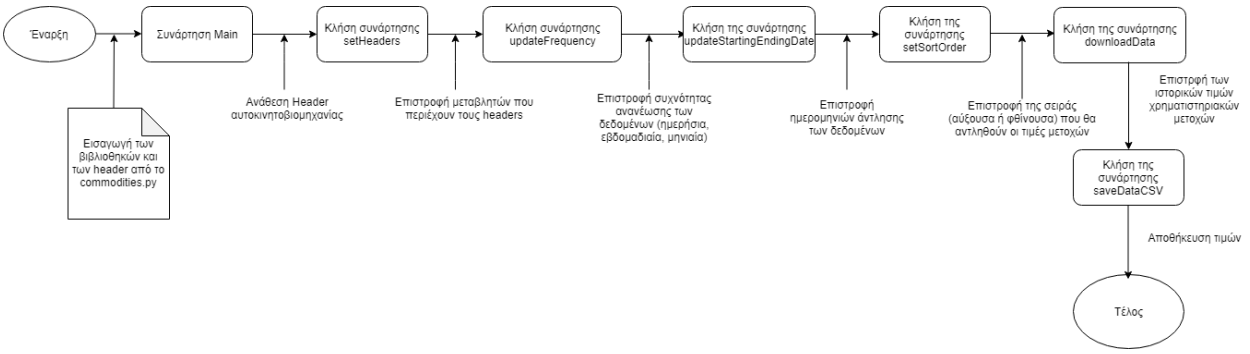
```
      def receiveBuffer(tweets):
         for t in tweets:
            outputFile.write(('\n%s;%d;%d;"%s";%s;%s;%s;"%s";%s'          %
(t.date.strftime("%Y-%m-%d"),  t.retweets,  t.favorites,  t.text,  t.geo,
t.mentions, t.hashtags, t.id, t.permalink)))
            #outputFile.write(('\n%s;%s;%s;%s' % ( t.date.strftime("%Y-%m-
%d"), t.text, t.retweets, t.favorites)))
         outputFile.flush()
         print('More %d saved on file...\n' % len(tweets))

      got.manager.TweetManager.getTweets(tweetCriteria, receiveBuffer)

   except arg:
      print('Arguments parser error, try -h' + arg)
   finally:
      outputFile.close()
      print('Done. Output file generated "%s".' % outputFileName)

if __name__ == '__main__':
   main(sys.argv[1:])
```

Κώδικας 4 - Λογισμικό συλλογής δεδομένων από την πλατφόρμα του Twitter.

4.3 Λογισμικό για τη συλλογή ιστορικών δεδομένων-τιμών μετοχών από το Investing.com

Το τελευταίο λογισμικό που αξιοποιείται προκειμένου να αντλήσω δεδομένα, είναι αυτό της συλλογής ιστορικών δεδομένων τιμών μετοχών για τις τρεις (3) αυτοκινητοβιομηχανίες. Πρόκειται για ένα λογισμικό με παρόμοια δομή με αυτή του λογισμικού στο κεφάλαιο 4.1, με την έννοια ότι αξιοποιεί την πλατφόρμα του Investing με σκοπό το κατέβασμα των δεδομένων.

Αποτελείται από δύο προγράμματα, το βασικό που είναι το ***IndiceHistoricalData.py*** και αυτό που περιέχει τους headers, ***commodities.py.*** Πρόκειται για έναν web crawler, ο οποίος αποτελείται από δύο προγράμματα python τα οποία επικοινωνούν μεταξύ τους. Το πρόγραμμα στέλνει POST request στην πλατφόρμα του Investing, ζητώντας τα ιστορικά δεδομένα για κάθε εταιρεία.



Εικόνα 8 - Διάγραμμα ροής του αλγορίθμου εξαγωγής των ιστορικών τιμών χρηματιστηριακών μετοχών

Παρακάτω παρουσιάζονται τα δύο πηγαία αρχεία ***IndiceHistoricalData.py***, ***commodities.py***, με τα οποία γίνεται η συλλογή των άρθρων από το Investing.com.

IndiceHistoricalData.py

Στις πρώτες εντολές του συγκεκριμένου προγράμματος, παρατηρούμε ότι εισάγονται οι βιβλιοθήκες python που θα μας χρειαστούν και θα χρησιμοποιήσουμε, όπως η pandas, η οποία θα μας βοηθήσει να διαβάσουμε τους πίνακες HTML, οι οποίοι περιέχουν τα δεδομένα με τις μετοχές που έχουν κατέβει.

Η επικοινωνία του προγράμματος αυτού με το δεύτερο πρόγραμμα γίνεται μέσα από την εντολή ***from indices.commodities import *,*** όπου το πρόγραμμα κάνει εισαγωγή όλων των header από το ***commodities.py*** και αυτό φαίνεται από την εντολή import *, που δίνει εντολή στο πρόγραμμα να εισάγει όλες τις διαθέσιμες συναρτήσεις και κλάσεις από το αρχείο στο οποίο αναφέρεται. Η πρώτη λέξη μετά τη δεσμευμένη λέξη from, υποδεικνύει το φάκελο προέλευσης του προγράμματος (indices) και η δεύτερη μετά την τελεία, υποδεικνύει το όνομα του προγράμματος που εισάγεται (commodities).

Αμέσως μετά στέλνονται οι header παράμετροι http, όπου περιέχουν τα στοιχεία του User-Agent που χρησιμοποιεί ο Web crawler για να επικοινωνήσει με το Application Programming Interface του Investing.com, ενώ επίσης περιέχεται και η αναφερόμενη πλατφόρμα (www.Investing.com), μέσω της εντολής ***'referer': "https://www.Investing.com"*** . Τα στοιχεία του U-A περιέχονται στο παρακάτω string: ***'User-Agent': 'Mozilla/5.0', #required, 'referer': "https://www.Investing.com", 'host' : 'www.Investing.com', 'X-Requested-With' : 'XMLHttpRequest'.***

Main συνάρτηση του προγράμματος

Όπως προαναφέρθηκε και με το πρώτο λογισμικό, είναι ουσιαστικά ο κύριος κορμός του αλγορίθμου, μέσα από την οποία ξεκινάει η λειτουργία του προγράμματος και δίνεται το έναυσμα για τη λήψη των ιστορικών τιμών των μετοχών. Οι πληροφορίες των headers εισάγονται από το πρόγραμμα ***commodities.py.*** Παρατηρούμε ότι ορίζεται η μεταβλητή ihd, η οποία ορίζεται ως αντικείμενο της κλάσης ***IndiceHistoricalData***() (θα αναλυθεί παρακάτω), στην οποία στέλνει και το url του Api της πλατφόρμας Investing.com (***ihd = IndiceHistoricalDATA('https://www.Investing.com/instruments/HistoricalDataAjax'***), προκειμένου να οριστεί από τον constructor – δομητή της κλάσης και να αξιοποιηθεί σε όλη την κλάση. Στη συνέχεια, υπάρχει το κάλεσμα των συναρτήσεων της κλάσης, μέσω των αντίστοιχων εντολών, με σκοπό την εξαγωγή των ιστορικών τιμών των μετοχών (αναλύονται παρακάτω).

Παρατήρηση: Στο συγκεκριμένο πρόγραμμα, ο ορισμός του ***Api_url,*** διαφέρει σε σχέση με τον ορισμό που έγινε στο πρόγραμμα ***articles.py (§ 4.1).*** Στο συγκεκριμένο πρόγραμμα, το url του Api είναι κοινό και για τις τρεις αυτοκινητοβιομηχανίες ***('https://www.Investing.com/instruments/HistoricalDataAjax')*** και η διάκριση των εταιρειών γίνεται σύμφωνα με άλλους ***headers. Στο articles.py,*** κάθε εταιρεία είχε και δικό της ***Api_url*** προκειμένου να αντλήσει τα κατάλληλα δεδομένα**.**

Κύρια κλάση *class IndiceHistoricalData()*

Αποτελεί και την μοναδική κλάση του αλγορίθμου, μέσα στην οποία υπάρχουν ορισμένες και όλες οι συναρτήσεις. Με την εντολή ***ihd = IndiceHistoricalDATA('https://www.Investing.com/instruments/HistoricalDataAjax***, καλείται ο constructor – συνάρτηση κατασκευής της κλάσης, προκειμένου να οριστεί το ***Api_url,*** με το url που στέλνεται μέσα από την ***main.*** Οι υπόλοιπες συναρτήσεις της κλάσης, καλούνται μέσα από τη main συνάρτηση, μέσω διάφορων εντολών, με τελικό στόχο τη συλλογή των ιστορικών τιμών**.**

Συνάρτηση *def __init__(self, API_url)*

Συνάρτηση κατασκευής της κλάσης, η οποία δέχεται ως όρισμα ***το Api_Url*** της πλατφόρμας του Investing και το ορίζει στην αντίστοιχη μεταβλητή ***Api_Url*** το url των ιστορικών δεδομένων της πλατφόρμας του Investing.

Συνάρτηση *def setHeaders(self, headers)*

Η συγκεκριμένη συνάρτηση αναλαμβάνει τον ορισμό των http headers για το request που θέλουμε να γίνει στο Api της πλατφόρμας. Πρόκειται για τους headers που ορίζονται ακριβώς πάνω από την υλοποίηση της κλάσης και είναι υπεύθυνοι για την επικοινωνία του Web Crawler

με το Api του Investing. Επομένως, δέχεται ως όρισμα τους headers και τους θέτει στην αντίστοιχη μεταβλητή. Καλείται από τη ***main,*** με την εντολή ***ihd.setHeaders(headers).***

Συνάρτηση *def setFormData(self, data)*

Μία ακόμα συνάρτηση η οποία αναλαμβάνει να ορίσει μία μεταβλητή της κλάσης. Δέχεται ως όρισμα τους headers που μεταφέρουν τα δεδομένα των αυτοκινητοβιομηχανιών. Καλείται από τη main με την εντολή ***ihd.setFormData(VW*** ή ***TSL*** ή ***FORD),*** όπου οι παράμετροι που στέλνονται είναι τα ονόματα με τα οποία ορίζονται οι λίστες στους headers**.**

Συνάρτηση *updateFrequency(self, frequency)*

Όπως υπαγορεύει και το όνομα, πρόκειται για μία συνάρτηση η οποία ορίζει τη συχνότητα ανανέωσης των δεδομένων. Η συχνότητα μπορεί να είναι είτε μηνιαία, είτε εβδομαδιαία, είτε ημερήσια. Καλείται από την κύρια συνάρτηση, με την εντολή ***ihd.updateFrequency('Monthly'*** ή ***'Weekly'*** ή ***'Daily').***

Συνάρτηση *def updateStartingEndingData(self, startingDate, endingDate)*

Είναι ακριβώς η ίδια συνάρτηση με αυτή του πρώτου λογισμικό στο κεφάλαιο 4.1, όπου ορίζονται οι επιθυμητές ημερομηνίες, ανάμεσα στις οποίες θα εξαχθούν οι ιστορικές τιμές των μετοχών. Δέχεται ως ορίσματα τις ημερομηνίες έναρξης και λήξης, ενώ καλείται από τη συνάρτηση ***main,*** μέσω της εντολής ***ad.updateStartingEndingData(datetime.data(YYYY-MM-DD), datetime.date(YYYY,MM,DD)).***

Συνάρτηση *def setSortOrder(self, sorting_order)*

Δίνει την επιλογή στον χρήστη να κατεβάσει τα δεδομένα με τη σειρά με την οποία ο ίδιος επιθυμεί, εάν θα είναι δηλαδή σε αύξουσα ή σε φθίνουσα σειρά. Το κάλεσμα γίνεται στη ***main*** συνάρτηση, με την εντολή ***ihd.setSordOder('ASC'*** ή ***'DESC'),*** με το ***'ASC'*** να σημαίνει ***ascending order*** (αύξουσα σειρά) και το ***'DESC'*** να σημαίνει ***descending order*** (φθίνουσα σειρά).

Συνάρτηση *def downloadData(self)*

Σε αυτό το κομμάτι του προγράμματος, περνάμε στο στάδιο όπου ξεκινάει η άντληση των ιστορικών δεδομένων. Είναι το σημείο όπου γίνεται το POST request προς την πλατφόρμα του Investing, στέλνοντας μέσω της μεταβλητής response, το Api_url, τα data και τους headers (***self.response = requests.post(self.API_url, data=self.data, headers=self.headers).content).*** Στη συνέχεια, με τη βοήθεια της βιβλιοθήκης της Python Pandas, αναλύονται τα δεδομένα που επιστρέφονται σε μορφή HTML και ορίζονται στην μεταβλητή observations, η οποία και επιστρέφεται μέσω της εντολής return self.observations. Αυτό σημαίνει πως η μεταβλητή ***ihd.downloadData(),*** η οποία βρίσκεται στη main συνάρτηση και καλεί την παραπάνω συνάρτηση, θα αποκτήσει τις τιμές που επιστράφηκαν με το post request**.**

Συνάρτηση *def printData(self)*

Προαιρετική συνάρτηση εκτύπωσης των δεδομένων που αντλήθηκαν μέσω του post request στην προηγούμενη συνάρτηση. Χρησιμοποιείται η εντολή ***print(self.observations) με*** στόχο την εμφάνιση των δεδομένων, ενώ καλείται από τη main συνάρτηση με την εντολή ***ihd.printData().***

Συνάρτηση *def saveDataCSV(self)*

Πρόκειται για το σημείο το οποίο τα δεδομένα αποθηκεύονται. Με τη βοήθεια πάλι της βιβλιοθήκης Pandas και συγκεκριμένα την εντολή ***self.observations.to_csv(self.data['name']+'.csv', sep=' ', encoding='utf-8'),*** δεδομένα αποθηκεύονται σε αρχείο csv, με όνομα, το όνομα της αυτοκινητοβιομηχανίας**,** το οποίο είναι αποθηκευμένο στην μεταβλητή ***data (***ορισμός από συνάρτηση ***setFormData)*** των headers και κωδικοποίηση utf-8. Για άλλη μια φορά, το κάλεσμα γίνεται από την main με την εντολή ***ihd.saveDataCSV().***

Στο παρακάτω στιγμιότυπο παρατηρούμε τη μορφή των δεδομένων, όπως αυτά αποθηκεύονται στο csv.

	A	B	C	D	E	F	G	H
1	Date	Price	Open	High	Low	Vol.	Change %	
2	28-May	144.84	144.42	146.66	143.9	962.26K	0.54%	
3	27-May	144.06	145.8	146.38	143.1	646.89K	0.84%	
4	24-May	142.86	144.44	145.1	142.46	947.29K	0.61%	
5	23-May	142	142.12	142.5	140.62	1.28M	-1.59%	
6	22-May	144.3	144.4	145.16	142.28	769.22K	-0.28%	
7	21-May	144.7	145.7	146.86	143.8	945.80K	-0.08%	
8	20-May	144.82	146.26	147.34	143.7	1.08M	-1.58%	
9	17-May	147.14	147.52	147.92	145.32	1.44M	-0.65%	
10	16-May	148.1	148.68	149.36	146.52	1.63M	-0.68%	
11	15-May	149.12	146.26	152.4	143.06	2.20M	0.07%	
12	14-May	149.02	151.12	151.86	148.3	1.52M	1.00%	
13	13-May	147.54	150.5	150.5	146.24	1.64M	-1.85%	
14	10-May	150.32	153	153.72	148.86	1.49M	-0.78%	
15	9-May	151.5	152.02	152.98	150.18	1.37M	-1.99%	
16	8-May	154.58	154	154.96	151.9	1.43M	0.10%	
17	7-May	154.42	157	157.74	153.04	1.33M	-2.03%	
18	6-May	157.62	155.38	157.94	152.82	1.54M	-1.66%	
19	3-May	160.28	161	162.5	159.86	868.12K	-0.29%	
20	2-May	160.74	159	163.1	158.4	2.01M	3.68%	
21	30-Apr	155.04	155	155.9	153.8	869.74K	-0.74%	
22	29-Apr	156.2	156.62	157.56	155.1	633.52K	0.28%	
23	26-Apr	155.76	155.22	156.18	154.4	828.36K	0.19%	
24	25-Apr	155.46	156.48	157.02	154.6	1.00M	-1.30%	
25	24-Apr	157.5	158	159.3	155.8	1.09M	-1.56%	
26	23-Apr	160	163.1	163.98	159.86	1.01M	-1.92%	
27	18-Apr	163.14	159.62	163.82	159.22	1.88M	1.10%	
28	17-Apr	161.36	157.62	162.1	157.44	1.87M	3.03%	
29	16-Apr	156.62	154.74	157.32	154.32	1.08M	1.06%	

VW

Εικόνα 9 - Στιγμιότυπο από τα δεδομένα των ιστορικών τιμών των μετοχών, όπως αυτά αποθηκεύτηκαν στο αρχείο csv.

Παρακάτω παρουσιάζεται ο πηγαίος κώδικας του προγράμματος ***IndiceHistoricalData.py***:

```
import pandas as pd
import requests
from indices.commodities import *

# set https header parameters
headers = {
   'User-Agent': 'Mozilla/5.0', #required
```

```
   'referer': "https://www.investing.com",
   'host' : 'www.investing.com',
   'X-Requested-With' : 'XMLHttpRequest'
}

class IndiceHistoricalData():

   def __init__(self, API_url):
      self.API_url = API_url

   #set https header for request
   def setHeaders(self, headers):
      self.headers = headers

   #set indice data (commodities.py)
   def setFormData(self, data):
      self.data = data

   #prices frequency, possible values: Monthly, Weekly, Daily
   def updateFrequency(self, frequency):
      self.data['frequency'] = frequency

   #desired time period from/to
   def updateStartingEndingDate(self, startingDate, endingDate):
      self.data['st_date'] = startingDate
      self.data['end_date'] = endingDate

   #possible values: 'DESC', 'ASC'
   def setSortOrder(self, sorting_order):
      self.data['sort_ord'] = sorting_order

   #making the post request
   def downloadData(self):
      self.response    =    requests.post(self.API_url,    data=self.data,
headers=self.headers).content
      #parse tables with pandas - [0] probably there is only one html table
in response
      self.observations = pd.read_html(self.response)[0]
      return self.observations

   #print retrieved data
   def printData(self):
      print(self.observations)

   #print retrieved data
   def saveDataCSV(self):
      self.observations.to_csv(self.data['name']+'.csv',      sep='      ',
encoding='utf-8')

if __name__ == "__main__":
```

```
   #first set Headers and FormData
   ihd                                                                    =
IndiceHistoricalData('https://www.investing.com/instruments/HistoricalDataA
jax')
   ihd.setHeaders(headers)
   ihd.setFormData(VW)

   #second set Variables
   ihd.updateFrequency('Monthly')
   ihd.updateStartingEndingDate('1/11/2018', '5/28/2019')
   ihd.setSortOrder('ASC')
   ihd.downloadData()
   ihd.printData()
   ihd.saveDataCSV()
```

Κώδικας 5 - Λογισμικό συλλογής ιστορικών τιμών μετοχών των αυτοκινητοβιομηχανιών από το Investing.com

Commodities.py.

Πρόκειται για το πρόγραμμα που περιέχει τους headers για τις τρεις (3) αυτοκινητοβιομηχανίες. Αναλυτικά, περιέχει τις παρακάτω πληροφορίες αποθηκευμένες σε μορφή λεξικού (dictionary):[3]

- Το όνομα του λεξικού
- Το όνομα της εταιρείας στο Investing ***('name')***
- Το id της εταιρείας στην πλατφόρμα του Investing, που είναι υπεύθυνο για τη διασύνδεση (‘***curr_id***’)
- Το id της εταιρείας (‘***smlID***’)
- Τον header που περιέχει την ονομασία των ιστορικών δεδομένων της (‘***header'***)
- Τη στήλη ταξινόμησης, σύμφωνα με την οποία θα γίνει η ταξινόμηση των υπόλοιπων στηλών (‘***sort_col'***)
- Το όνομα της ενέργειας, δηλαδή ιστορικά δεδομένα (‘***action***’).

Τα αντικείμενα του λεξικού είναι προσπελάσιμα μέσω του index (δείκτης) του κάθε αντικειμένου.

Ακολουθεί ο επισυναπτόμενος κώδικας του αρχείου ***commodities.py***:

[3] Ένα λεξικό είναι μία μορφή συλλογής δεδομένων σε python, η οποία είναι μη-καθορισμένη, τροποποιήσιμη και υπό μορφή δεικτών.

```
TSL = {
   'name' : 'TSL',
   'curr_id': 13994,
   'smlID': 1163215,
   'header' : 'TSLA Historical Data',
   'sort_col' : 'date',
   'action' : 'historical_data'
}

#https://www.investing.com/equities/volkswagen-vz-historical-data
VW = {
   'name' : 'VW',
   'curr_id': 22402,
   'smlID': 1161537,
   'header' : 'VOWG_p Historical Data',
   'sort_col' : 'date',
   'action' : 'historical_data'
}

#https://www.investing.com/equities/ford-motor-co-historical-data
FORD = {
   'name' : 'FORD',
   'curr_id': 255,
   'smlID': 1159492,
   'header' : 'F Historical Data',
   'sort_col' : 'date',
   'action' : 'historical_data'
}
```

Κώδικας 6 - Πηγαίο πρόγραμμα το οποίο περιέχει τους βασικούς headers με τις πληροφορίες που χρειαζόμαστε για το API του Investing.com.

Κεφάλαιο 5: Επεξεργασία δεδομένων

Αφού ολοκληρωθεί η διαδικασία συλλογής των δεδομένων, ακολουθούν οι ενέργειες της γενικότερης επεξεργασίας τους. Με άλλα λόγια, τα δεδομένα αυτά χρειάζεται να φιλτραριστούν προκειμένου να ακολουθηθεί σωστά η διαδικασία της επεξεργασίας τους.

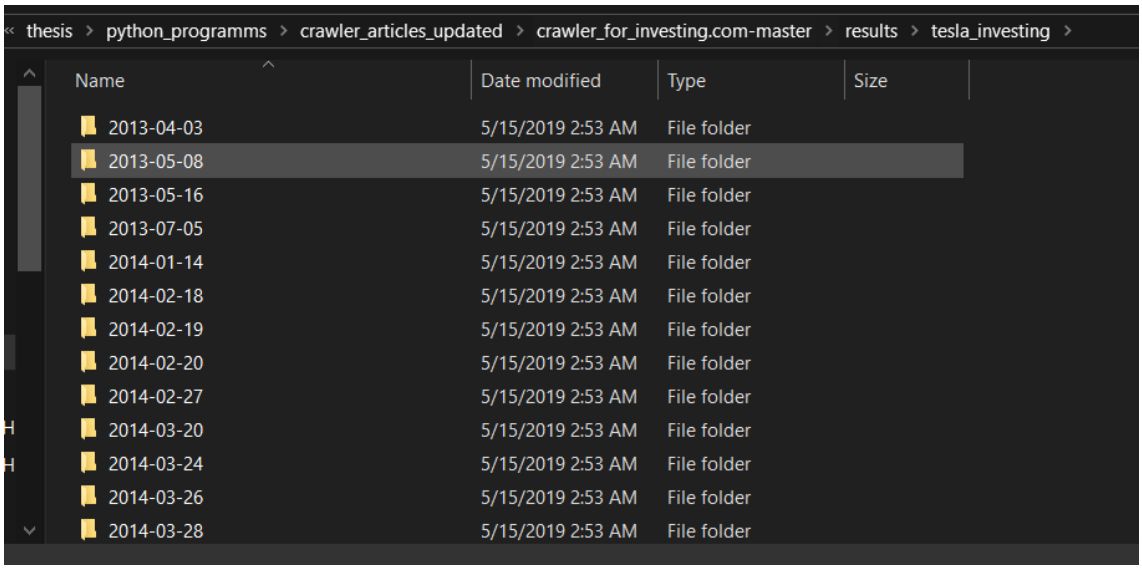
Η διαδικασία για τα κείμενα και των δύο πηγών είναι η ίδια: Αρχικά, χρειάζεται να ταξινομηθούν τα κείμενα ανά ημέρα. Αυτό σημαίνει ότι για κάθε ημέρα, θα συγκεντρωθούν όλα τα κείμενα και θα γίνουν ένα ενιαίο κείμενο. Στη συνέχεια, με τη βοήθεια κατάλληλων προγραμμάτων – σεναρίων (scripts) και την εισαγωγή κατάλληλων βιβλιοθηκών, γίνεται ο καθαρισμός λέξεων και σημείων στίξης στα κείμενα, καθώς επηρεάζουν αρνητικά τα αποτελέσματα των εκτιμήσεων. Αφού ολοκληρωθούν αυτές οι ενέργειες, καθορίζονται οι ανεξάρτητες μεταβλητές που θα συμβάλλουν στις εκτιμήσεις και τέλος γίνεται ο καθορισμός των τριών διαφορετικών μοντέλων.

5.1 Ταξινόμηση και συγχώνευση κειμένων (tweets – άρθρα)

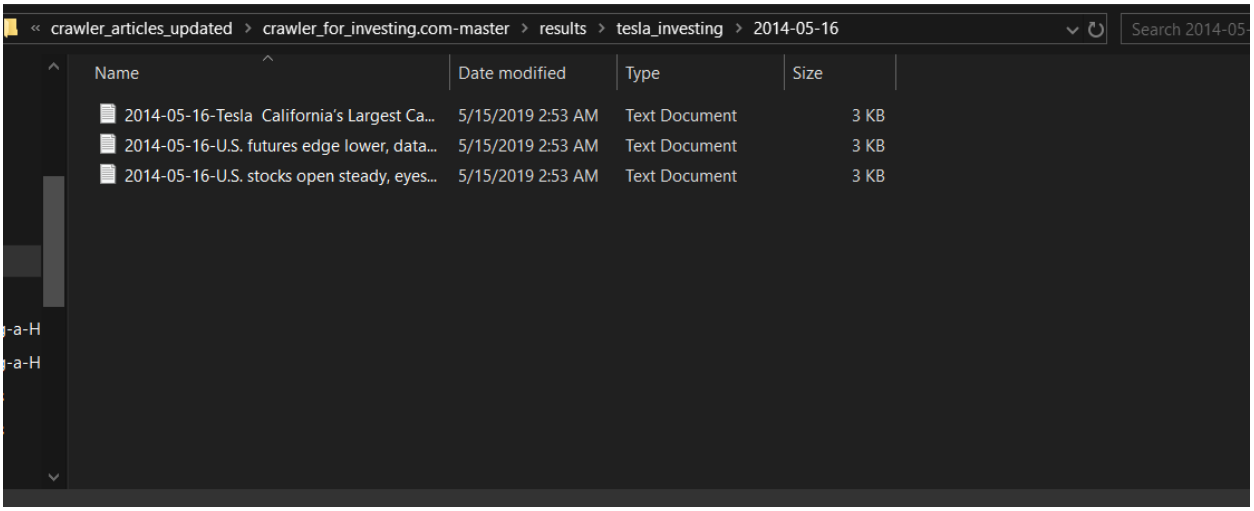
Η διαδικασία της ταξινόμησης και συγχώνευσης των άρθρων, είναι διαφορετική από την ταξινόμηση των tweets. Παρόλο που χρησιμοποιείται το ίδιο πρόγραμμα για την ταξινόμησή τους ανά ημέρα, υπάρχει μια σημαντική ειδοποιός διαφορά: τα κείμενα των άρθρων συλλέγονται υπό μορφή .txt, επομένως πρέπει να συγκεντρωθούν όλα τα txt, από κάθε ημέρα και στη συνέχεια να μεταφερθούν σε κάποιο έγγραφο .csv.

5.1.1 Συγχώνευση άρθρων

Όπως προαναφέρθηκε, τα κείμενα των άρθρων συλλέγονται υπό μορφή .txt και σε διαφορετικούς φακέλους, όπως φαίνεται και στις παρακάτω φωτογραφίες:



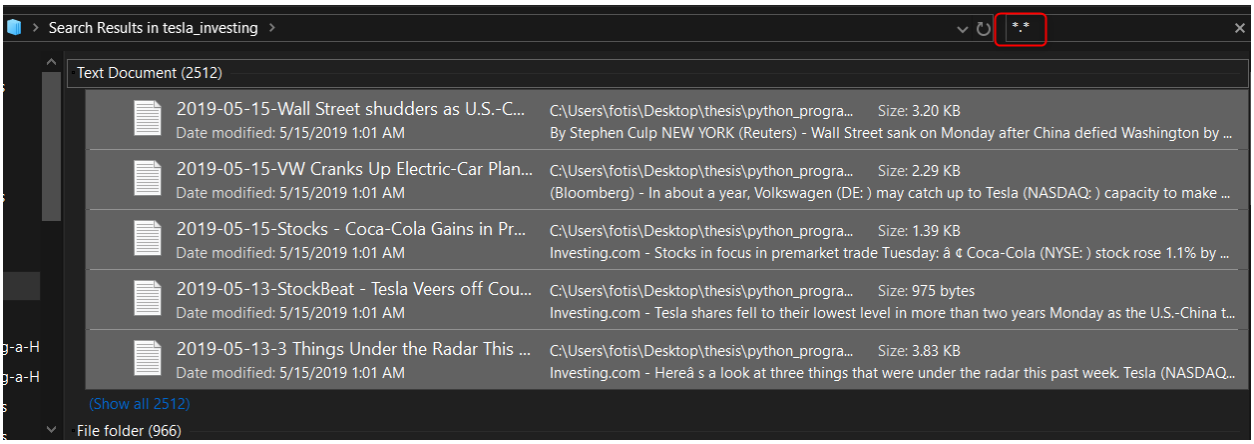
Εικόνα 10 - Στιγμιότυπο από τους φακέλους που περιέχουν τα κείμενα, χωρισμένα ανά ημερομηνία δημιουργίας τους.



Εικόνα 11 - Στιγμιότυπο από τα αρχεία που βρίσκονται μέσα στους φακέλους

Επομένως, τίθενται δύο εύλογα ερωτήματα. Πρώτον, πώς θα μπορέσουμε να συλλέξουμε όλη την πληροφορία μαζεμένη σε κάποια σειρά και δεύτερον, πώς θα μπορέσουμε να εισάγουμε όλα τα άρθρα, σε ένα csv, χωρισμένα ανά ημερομηνίες; Καταλαβαίνουμε ότι για το δεύτερο ερώτημα, είναι αδύνατο να ανοίγουμε κάθε αρχείο txt ξεχωριστά και να αντιγράφουμε το περιεχόμενο, καθώς οι εγγραφές που συλλέγουμε είναι χιλιάδες.

Η απάντηση στο πρώτο ερώτημα δίνεται με τον εξής τρόπο:

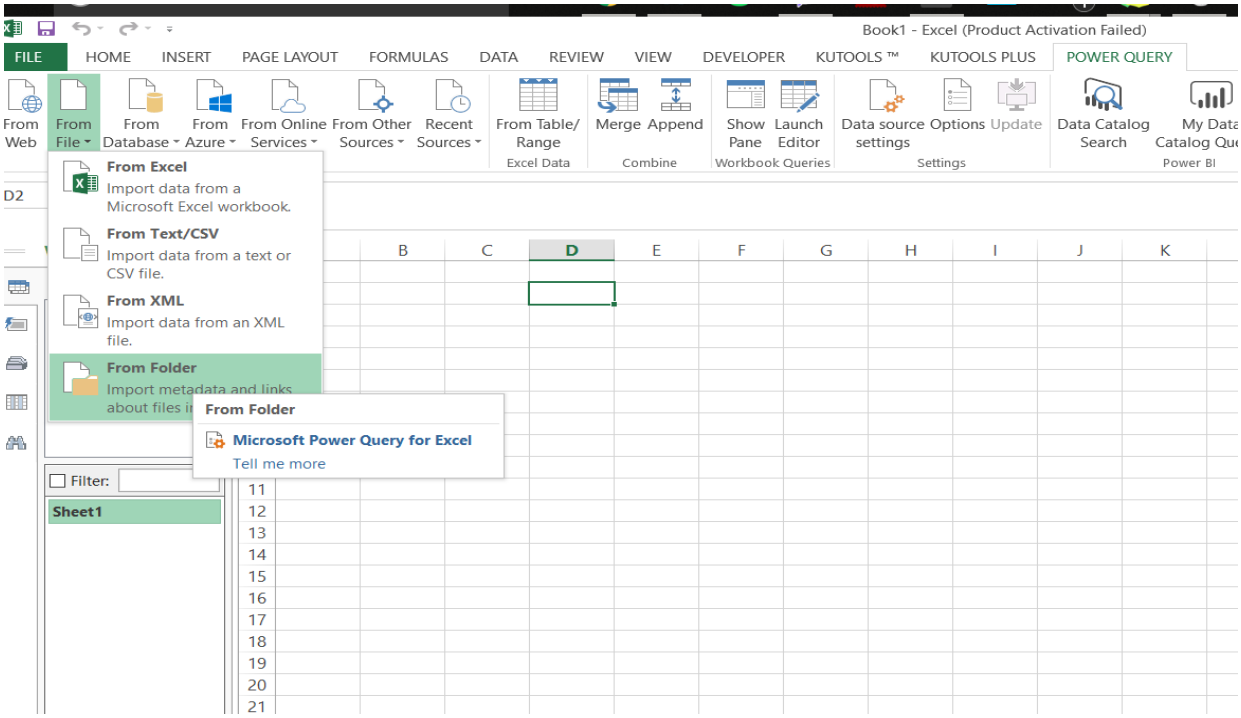


Εικόνα 12 - Στιγμιότυπο από τη συγκεντρωτική συλλογή δεδομένων

Στην παραπάνω εικόνα διακρίνουμε τα αρχεία συγκεντρωμένα και έτοιμα προς αντιγραφή, προκειμένου να μεταφέρουμε σε έναν καινούργιο φάκελο, στον οποίο θα είναι όλα μαζί. Για να επιτευχθεί αυτό, χρειάστηκε μέσα στο φάκελο της εταιρείας που περιέχει τους φακέλους με τις ημερομηνίες, να πληκτρολογήσουμε στην αναζήτηση "*.*", κάτι το οποίο επιστρέφει τα αρχεία χωρισμένα ανά κατηγορίες φακέλων. Πατώντας στη συνέχεια δεξί κλικ και group by ανά τύπο, επιστρέφονται τα αρχεία χωρισμένα ανά τύπο και έτσι έχουμε την επιλογή της μαζικής αντιγραφής τους, όπως φαίνεται και παραπάνω.

Αφού τα μεταφέρουμε σε έναν καινούργιο και άδειο φάκελο, στη συνέχεια πρέπει να λύσουμε το δεύτερο ερώτημα. Για να επιλύσουμε λοιπόν και αυτό το ζήτημα, ανοίγουμε ένα κενό αρχείο excel, επιλέγουμε τη στήλη Power Query[4] και μετέπειτα επιλέγουμε, όπως φαίνεται και στην παρακάτω φωτογραφία, From File > From Folder. Με αυτόν τον τρόπο, μας δίνεται η επιλογή να επιλέξουμε έναν ολόκληρο φάκελο και να ανοίξουμε τα δεδομένα που περιέχει.

[4] Το Power Query είναι ένα add-on εργαλείο του Microsoft Excel, το οποίο προσφέρει πολλές λειτουργείες όπως επεξεργασία πινάκων, εύρεση και σύνδεση δεδομένων από ένα ευρύ φάσμα προελεύσεων. εισαγωγή δεδομένων από πολλά αρχεία καταγραφής και άλλα πολλά. Περισσότερες πληροφορίες στο [54]



Εικόνα 13 - Στιγμιότυπο από την εισαγωγή αρχείων στο excel

Αφού γίνει η εισαγωγή των αρχείων, ανοίγουν μερικές στήλες, όπως το όνομα του αρχείο, το περιεχόμενο σε δυαδική μορφή, ο τύπος του αρχείου και η ημερομηνία εισαγωγής. Αφού σβήσουμε τις περιττές στήλες και κρατήσουμε μόνο όνομα αρχείου και περιεχόμενο, φορτώνουμε το περιεχόμενο σε έναν πίνακα, ο οποίος περιέχει τα κείμενα σε κανονική μορφή:

	Name	Transform File from tesla_investing.Column1
1	2013-04-03-U.S. futures edge higher ahead of jobs data; Dow Jones up...	Investing.com - U.S. stock futures pointed to a moderately higher ope...
2	2013-04-03-U.S. stocks steady, eyes on ISM report; Dow Jones down 0...	Investing.com - U.S. stocks were steady on Wednesday, after the relea...
3	2013-05-08-U.S. stocks open lower in quiet trade; Dow Jones down 0....	Add a Comment
4	2013-05-16-U.S. futures steady ahead of data; Dow Jones up 0.03% .txt	Add a Comment
5	2013-05-16-U.S. stocks mixed after economic reports; Dow Jones dow...	Add a Comment
6	2013-07-05-U.S. futures jump with employment data ahead; Dow Jone...	Add a Comment
7	2013-07-05-U.S. stocks higher on upbeat employment data; Dow Jone...	Investing.com - U.S. stocks opened higher on Friday, supported by the ...
8	2014-01-14-A Cheaper Tesla Has Arrived; Now, About That Range....txt	By - Jerome Guillen, Telsa Motors Inc. (NASDAQ:TSLA) head of global ...
9	2014-02-18-Tesla 4Q And Full Year Results Here's What To Look For.txt	By -Â As with many exciting startups, market watchers and investors ...
10	2014-02-19-Tesla Makes Toys For The Rich, Says Chinese Auto Exec.txt	By -Â The head of one of Chinaâ€™s major automakers brushed aside...
11	2014-02-20-Tesla shares rally 10% in pre-market trade after upbeat Q...	Investing.com - Tesla Motors saw shares rally sharply in pre-market tra...
12	2014-02-27-Tesla Again Captures Wall Street's Imagination With New ...	By -Â On Wednesday Tesla Motors Inc. (NASDAQ:TSLA), the all-electri...
13	2014-03-20-Bespoke Legislation Arizona Courts Tesla.txt	By -Â A week after New Jersey joined four other states in blocking eff...
14	2014-03-24-U.S. stocks edge lower on factory data, biotech selloff; Do...	Investing.com - U.S. stocks edged lower on Monday after U.S. factory ...
15	2014-03-26-U.S. stocks open higher after durable goods report; Dow J...	Investing.com - U.S. stocks opened higher on Wednesday, after the rel...
16	2014-03-28-U.S. stocks open higher, UoM report ahead; Dow Jones up...	Investing.com - U.S. stocks opened higher on Friday, as renewed opti...
17	2014-03-31-U.S. stocks open higher, eyes on Yellen speech; Dow Jones...	Investing.com - U.S. stocks opened higher on Monday, as ongoing spec...
18	2014-04-03-U.S. stocks edge higher after downbeat data; Dow Jones u...	Investing.com - U.S. stocks opened moderately higher on Thursday, aft...
19	2014-04-10-U.S. futures decline ahead of jobless data; Dow Jones dow...	Investing.com - U.S. stock futures pointed to a lower open on Thursda...
20	2014-04-10-U.S. stocks edge lower after jobless claims; Dow Jones do...	Investing.com - U.S. stocks opened moderately lower on Thursday, aft...
21	2014-04-22-Hip Beijing Drivers Hang On - More Teslas On The Way.txt	By - Hip Beijing drivers can breath a sigh of relief, not due to any easi...
22	2014-05-06-Tesla's First Quarter What To Look For.txt	By -Â Tesla Motors Inc ( ) has emerged as the first luxury-level electric...

Εικόνα 14 - Στήλες με τα ονόματα των αρχείων και τα περιεχόμενά τους

Με την ολοκλήρωση και αυτής της ενέργειας, μεταφέρουμε τα δεδομένα σε ένα νέο αρχείο excel και παίρνουμε το τελικό αποτέλεσμα που θέλουμε, χωρίς να είναι τα κείμενα σε δυαδική μορφή:

A	B
Name	Transform File from tesla_investing.Column1
2013-04-03-U.S. futures edge higher ahead of jobs data; Dow Jones up 0.08% .txt	Investing.com - U.S. stock futures pointed to a moderately higher open on Wedne
2013-04-03-U.S. stocks steady, eyes on ISM report; Dow Jones down 0.01% .txt	Investing.com - U.S. stocks were steady on Wednesday, after the release of disap
2013-05-08-U.S. stocks open lower in quiet trade; Dow Jones down 0.18% .txt	Add a Comment
2013-05-16-U.S. futures steady ahead of data; Dow Jones up 0.03% .txt	Add a Comment
2013-05-16-U.S. stocks mixed after economic reports; Dow Jones down 0.05% .txt	Add a Comment
2013-07-05-U.S. futures jump with employment data ahead; Dow Jones up 0.90% .txt	Add a Comment
2013-07-05-U.S. stocks higher on upbeat employment data; Dow Jones up 0.47% .txt	Investing.com - U.S. stocks opened higher on Friday, supported by the release of
2014-01-14-A Cheaper Tesla Has Arrived; Now, About That Range....txt	By - Jerome Guillen, Telsa Motors Inc. (NASDAQ:TSLA) head of global sales, woul
2014-02-18-Tesla 4Q And Full Year Results Here's What To Look For.txt	By -Â As with many exciting startups, market watchers and investors arenâ€™t lo
2014-02-19-Tesla Makes Toys For The Rich, Says Chinese Auto Exec.txt	By -Â The head of one of Chinaâ€™s major automakers brushed aside concerns
2014-02-20-Tesla shares rally 10% in pre-market trade after upbeat Q4 earnings.txt	Investing.com - Tesla Motors saw shares rally sharply in pre-market trade on Thur
2014-02-27-Tesla Again Captures Wall Street's Imagination With New Battery Gigafactory Revelatic	By -Â On Wednesday Tesla Motors Inc. (NASDAQ:TSLA), the all-electric vehicle m
2014-03-20-Bespoke Legislation Arizona Courts Tesla.txt	By -Â A week after New Jersey joined four other states in blocking efforts by Tesl
2014-03-24-U.S. stocks edge lower on factory data, biotech selloff; Dow slips 0.16%.txt	Investing.com - U.S. stocks edged lower on Monday after U.S. factory data misse
2014-03-26-U.S. stocks open higher after durable goods report; Dow Jones up 0.42%.txt	Investing.com - U.S. stocks opened higher on Wednesday, after the release of upl
2014-03-28-U.S. stocks open higher, UoM report ahead; Dow Jones up 0.32%.txt	Investing.com - U.S. stocks opened higher on Friday, as renewed optimism over th
2014-03-31-U.S. stocks open higher, eyes on Yellen speech; Dow Jones up 0.73%.txt	Investing.com - U.S. stocks opened higher on Monday, as ongoing speculation ov
2014-04-03-U.S. stocks edge higher after downbeat data; Dow Jones up 0.10%.txt	Investing.com - U.S. stocks opened moderately higher on Thursday, after the relea
2014-04-10-U.S. futures decline ahead of jobless data; Dow Jones down 0.22%.txt	Investing.com - U.S. stock futures pointed to a lower open on Thursday, ahead of
2014-04-10-U.S. stocks edge lower after jobless claims; Dow Jones down 0.02%.txt	Investing.com - U.S. stocks opened moderately lower on Thursday, after the relea
2014-04-22-Hip Beijing Drivers Hang On - More Teslas On The Way.txt	By - Hip Beijing drivers can breath a sigh of relief, not due to any easing of the na
2014-05-06-Tesla's First Quarter What To Look For.txt	By -Â Tesla Motors Inc ( ) has emerged as the first luxury-level electric sedan mal
2014-05-07-U.S. futures steady ahead of Yellen testimony; Dow Jones down 0.04%.txt	Investing.com - U.S. stock futures pointed to a steady open on Wednesday, as ma
2014-05-07-U.S. stocks open higher with eyes on Yellen, Ukraine; Dow Jones up 0.38%.txt	Investing.com - U.S. stocks opened higher on Wednesday, as investors awaited te
2014-05-08-U.S. stocks end mixed to lower on earnings; Dow rises 0.20%.txt	Investing.com - U.S. stocks ended Thursday mixed to lower after investors bough
2014-05-16-Tesla California's Largest Car Company Employer.txt	By -Â California may not be the front-runner for Tesla Motorsâ€™ planned $5 bil
2014-05-16-U.S. futures edge lower, data in focus; Dow Jones down 0.14%.txt	Investing.com - U.S. stock futures pointed to a moderately lower open on Friday,
2014-05-16-U.S. stocks open steady, eyes on UoM report; Dow Jones up 0.02%.txt	Investing.com - U.S. stocks opened steady on Friday, after the release of upbeat

Εικόνα 15 - Εισαγμένα δεδομένα στο αρχείο excel

Η διαδικασία αυτή επαναλαμβάνεται και για τις τρεις αυτοκινητοβιομηχανίες και έως ότου φτάσουμε στην επιθυμητή τελική ημερομηνία εξαγωγής δεδομένων. Το επόμενο βήμα, είναι αυτό της συγχώνευσης των άρθρων, ταξινομημένα ανά ημέρα. Η συγχώνευση γίνεται σύμφωνα με το παρακάτω κομμάτι κώδικα:

```
if type == 'article':
    df = pd.read_csv("brand_name.csv", encoding="ISO-8859-1")
    text = df.groupby(['Date'])['Text'].apply(lambda x: " ".join(x.astype(str))).reset_index()
    articles = df.groupby(['Date']).count().reindex(df['Date'].unique())

    outputFileName = "ford.txt"
    outputFile = codecs.open(outputFileName, "w+", "utf-8")
    outputFile.write('Artcles')
    for x in range(0, 913):
        outputFile.write('\n%s;%s' % (text['Date'][x], text['Text'][x]))
```

Κώδικας 7 - Κομμάτι κώδικα, σύμφωνα με το οποίο γίνεται η ταξινόμηση και η συγχώνευση των άρθρων

Με τη βοήθεια της βιβλιοθήκης Pandas, διαβάζουμε το αρχείο .csv στο οποίο έχουμε αποθηκεύσει τα άρθρα της εκάστοτε εταιρείας (***df = pd.read_csv("brand_name.csv", encoding="ISO-8859-1")***). Αμέσως μετά, ο αλγόριθμος ταξινομεί τα άρθρα σύμφωνα με τις ημερομηνίες που έχουν δημιουργηθεί και συγχωνεύει τη στήλη '*Text*', όπου περιέχει τα κείμενα. Τα αποτέλεσμα είναι να συγχωνευτούν όλα τα κείμενα κάθε ημέρας, στην αντίστοιχη ημερομηνία. Επιπλέον, αποθηκεύουμε σε μία μεταβλητή ***'articles'***, τον αριθμό των άρθρων που υπάρχει κάθε ημέρα. Οι εντολές για τη συγχώνευση και την καταμέτρηση των άρθρων είναι οι παρακάτω:

```
text = df.groupby(['Date'])['Text'].apply(lambda x: "
".join(x.astype(str))).reset_index()
articles = df.groupby(['Date']).count().reindex(df['Date'].unique())
```

Κώδικας 8 - Εντολές συγχώνευσης και καταμέτρησης των άρθρων.

Τέλος, και αφού τελειώσει η διαδικασία της συγχώνευσης, χρειάζεται να προσθέσουμε τις ημερομηνίες που λείπουν από τη στήλη "***Date***", για να υπάρχει μία λογική συνέχεια στις ημερομηνίες. Το κομμάτι κώδικα με το οποίο γίνεται αυτή η διαδικασία, είναι το παρακάτω:

```
df = pd.read_csv("brand_name.csv", encoding = "ISO-8859-1", index_col="Date")
print(df)
df.index = pd.DatetimeIndex(df.index)
df = df.reindex(pd.date_range("date-since", "date-until"), fill_value=" ")
df.to_csv('brand_name.csv')
```

Κώδικας 9 - Πρόσθεση ενδιάμεσων ημερομηνιών που λείπουν

Παρακάτω, ακολουθεί ένα στιγμιότυπο το οποίο αποτυπώνει ακριβώς αυτή η διαδικασία. Η πρώτη στήλη αριστερά αποτελεί την ημερομηνία, η μεσαία είναι το κείμενο συγχωνευμένο και η στήλη δεξιά αποτυπώνει τα συνολικά άρθρα κάθε ημέρας.

3/21/2019	reuters ford motor n	4
3/20/2019	ben klayman detroit	2
3/19/2019	investing com car inc	5
3/18/2019	ben klayman detroit	2

Εικόνα 13 - Στιγμιότυπο από τα συγχωνευμένα άρθρα

5.1.2 Συγχώνευση Tweets

Από την άλλη πλευρά, η διαδικασία της συγχώνευσης των tweets είναι πιο απλουστευμένη, καθώς τα δεδομένα που εξάγονται είναι έτοιμα προς επεξεργασία.

A	B	C	D	E	F	G	H	I	J	K
date	retweets	favorites	text	mentions	hashtags					
5/26/2019	10	328	No							
5/26/2019	10	250	All Tesla models come standard with 'Fast Mode'							
5/26/2019	21	330	Who needs advertising when you've got this wisdom from 4-year-old Scarlet from Cl							
5/26/2019	401	5172	We Interru @							
5/24/2019	7	226	Ah							
5/24/2019	7	238	We may even have a square to spare. Let us know if you would like to borrow one							

Εικόνα 16 - Στιγμιότυπο από τα εξαγόμενα δεδομένα του Twitter

Όπως φαίνεται, έχουμε τις στήλες '*date*'. '*retweets*', '*favorites*', '*text*', 'mentions και 'hashtags', οι οποίες θα συγχωνευτούν σύμφωνα με τις ημέρες. Η στήλη '*text*' θα περιέχει το σύνολο των κειμένων για κάθε ημέρα, οι στήλες '*favorites*', και '*retweets*' θα περιέχουν το συνολικό άθροισμα για κάθε μέρα, ενώ οι στήλες '*mentions*' και '*hashtags*' θα περιέχουν τα σύνολα των mentions (@) και hashtags (#) για κάθε ημέρα. Η συγχώνευση λοιπόν για τα tweets γίνεται σύμφωνα με το παρακάτω κομμάτι κώδικα:

```
elif type == 'tweet':
    print('tweets')
    df = pd.read_csv("brand_name.csv", encoding = "ISO-8859-1")
    text      =      df.groupby(['Date'])['Text'].apply(lambda      x:      "
".join(x.astype(str))).reset_index()

    sums = df.groupby(['date']).sum()
    retweets = df.groupby(['date']).sum()
    mentions    =    df.groupby(['date'])['mentions'].apply(lambda    x:    "
".join(x.astype(str))).reset_index()
    hash     =     df.groupby(['date'])['hashtags'].apply(lambda     x:     "
".join(x.astype(str))).reset_index()
    print(sums)
    outputFileName = "brand_name_final.csv"
    outputFile = codecs.open(outputFileName, "w+", "utf-8")
    outputFile.write('date;text;retweets;favorites;mentions;hash')
    for x in range(0,1282):
        outputFile.write('\n%s;%s;%s;%s;%s;%s'      %      (text['date'][x],
text['text'][x],         sums['retweets'][x],         sums['favorites'][x],
mentions['mentions'][x], hash['hashtags'][x]))
```

Κώδικας 10 - Κομμάτι κώδικα, σύμφωνα με το οποίο γίνεται η ταξινόμηση και η συγχώνευση των άρθρων

Για άλλη μια φορά, με τη βοήθεια της βιβλιοθήκης Pandas, διαβάζουμε το αρχείο .csv στο οποίο έχουμε αποθηκεύσεις τα άρθρα της εκάστοτε εταιρείας (***df = pd.read_csv("brand_name.csv", encoding="ISO-8859-1")***). Αμέσως μετά, ο αλγόριθμος ταξινομεί τα άρθρα σύμφωνα με τις ημερομηνίες που έχουν δημιουργηθεί και συγχωνεύει τη στήλη '*Text*', όπου περιέχει τα κείμενα, ενώ ταυτόχρονα αθροίζει τον αριθμό των retweets, favorites, hashtags και mentions ανά ημέρα.

Όπως συνέβη και στην περίπτωση των άρθρων, χρειάζεται να προστεθούν οι ενδιάμεσες ημερομηνίες, προκειμένου να υπάρχει μία συνέχεια με τις ημερομηνίες. Για να συμβεί αυτό, χρησιμοποιούμε το ίδιο κομμάτι κώδικα (***Κώδικας 9***).

Παρακάτω, ακολουθεί ένα στιγμιότυπο το οποίο μας δίνει μία ιδέα για το πως διαμορφώνονται τα δεδομένα μετά τη συγχώνευση.

	Date	Text	Retweets	Favorites	Mentions	Hashtags
2	5/12/2019	tes lacom	2053	34889	0	0
3	5/11/2019	think lying	2550	39461	1	0
4	5/10/2019	mention h	16022	166218	0	0
5	5/9/2019	sorry spar	9770	116246	0	1
6	5/8/2019	coffee cu	2284	49827	0	0
7	5/7/2019	rude fun c	9437	166698	1	1
8	5/6/2019	want us n	6299	103701	0	0

Εικόνα 17 - Στιγμιότυπο από τα συγχωνευμένα Tweets

Τέλος, παρατίθεται ολόκληρος ο αλγόριθμος σύμφωνα με την οποίο γίνεται η συγχώνευση τόσο των άρθρων, όσο και των tweets:

```
import pandas as pd
pd.set_option('display.max_colwidth', -1)
pd.set_option('display.max_rows', 10000)
pd.set_option('display.max_columns', 500)
pd.set_option('display.width', 4000)
import sys,getopt,datetime,codecs
import dateutil

type = input("Articles or Tweets?: ")

if type == 'article':
    df = pd.read_csv("brand_name.csv", encoding="ISO-8859-1")
    text      =      df.groupby(['Date'])['Text'].apply(lambda      x:      "
".join(x.astype(str))).reset_index()
    articles = df.groupby(['Date']).count().reindex(df['Date'].unique())

    outputFileName = "brand_name.txt"
    outputFile = codecs.open(outputFileName, "w+", "utf-8")
    outputFile.write('Artcles')
    for x in range(0, 913):
        outputFile.write('\n%s;%s' % (text['Date'][x], text['Text'][x]))
elif type == 'tweet':
    print('tweets')
    df = pd.read_csv("brand_name.csv", encoding = "ISO-8859-1")
    text      =      df.groupby(['Date'])['Text'].apply(lambda      x:      "
".join(x.astype(str))).reset_index()

    sums = df.groupby(['date']).sum()
    #retweets = df.groupby(['date']).sum()
```

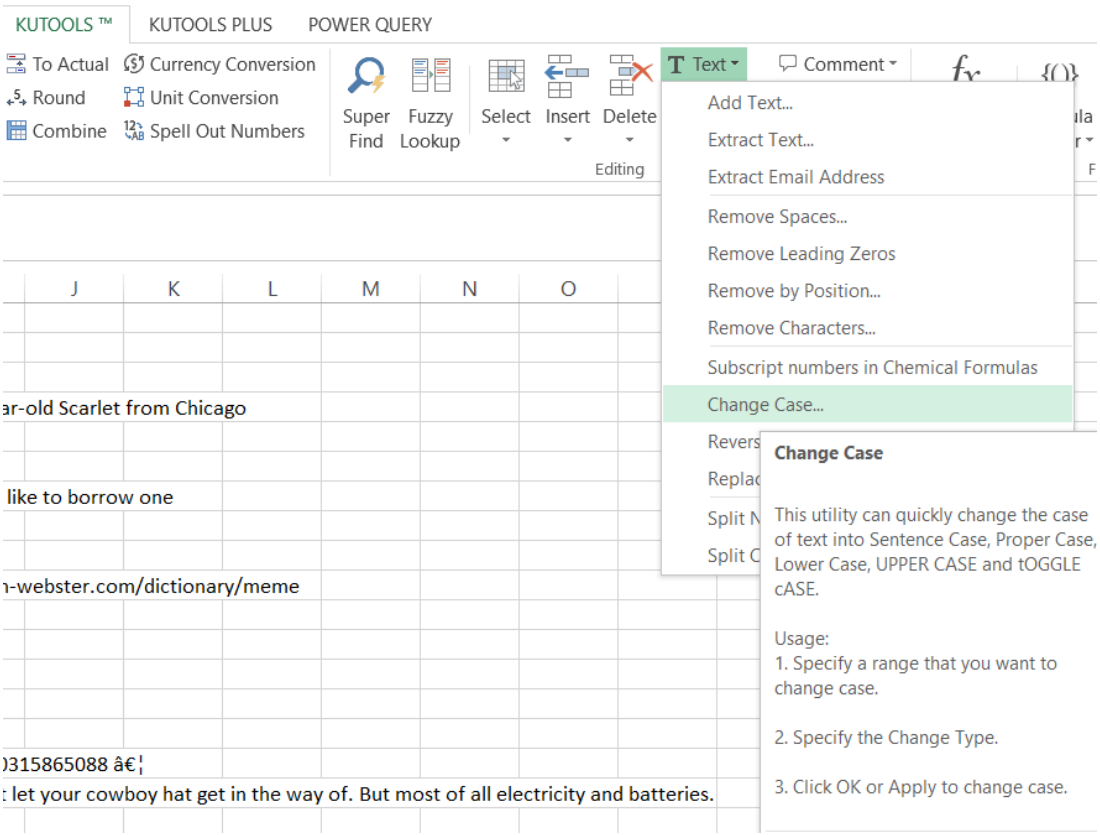
```
    mentions    =    df.groupby(['date'])['mentions'].apply(lambda    x:    "
".join(x.astype(str))).reset_index()
    hash    =    df.groupby(['date'])['hashtags'].apply(lambda    x:    "
".join(x.astype(str))).reset_index()
    print(sums)
    outputFileName = "brand_final.csv"
    outputFile = codecs.open(outputFileName, "w+", "utf-8")
    outputFile.write('date;text;retweets;favorites;mentions;hash')
    for x in range(0,1282):
        outputFile.write('\n%s;%s;%s;%s;%s;%s'     %     (text['date'][x],
text['text'][x],         sums['retweets'][x],         sums['favorites'][x],
mentions['mentions'][x], hash['hashtags'][x]))
else:
    print("error type")
```

Κώδικας 11 - Αλγόριθμος συγχώνευσης

5.2 Καθαρισμός κειμένων

Έχοντας ολοκληρώσει τη βασική προ-επεξεργασία των δεδομένων προκειμένου να τα συγχωνεύσουμε σύμφωνα με τις ημερομηνίες δημιουργίας τους, ακολουθεί το στάδιο της βασικής τους επεξεργασίας, αυτός του καθαρισμού των κειμένων από περιττά στοιχεία. Ο καθαρισμός χωρίζεται σε δύο στάδια: Το πρώτο αποτελεί αυτό της αφαίρεσης των σημείων στίξης από τα κείμενα και το δεύτερο είναι αυτό της αφαίρεσης των stop words. Ένα stop word, είναι μία λέξη η οποία χρησιμοποιείται ευρέως, όπως ένα άρθρο, ένας σύνδεσμος και μία αντωνυμία (παραδείγματος χάριν “the”, “a”, “an”, “in”). Λέξεις τέτοιου είδους, λόγω της συχνής εμφάνισής τους και της ουδέτερης σημασίας τους, τις αφαιρούμε από τα κείμενα. [31]

Αρχικά, χρειάζεται να διευκρινιστεί μία μικρή λεπτομέρεια, πριν μεταβούμε στα δύο βήματα. Χρειάζεται να μετατρέψουμε όσες λέξεις έχουν κεφαλαία γράμματα σε μικρά, καθώς οι βιβλιοθήκες της Python που χρησιμοποιούμε προκειμένου να αφαιρέσουμε τα stop words, προϋποθέτουν ότι όλες οι λέξεις έχουν μικρά γράμματα. Η μετατροπή των κειμένων σε μικρά γράμματα έγινε με τη χρήση ενός add-on στο excel, το λεγόμενο kutools, το οποίο περιλαμβάνει βοηθητικές συναρτήσεις και εργαλεία, τα οποία απλοποιούν πολλές διαδικασίες. [32]



Εικόνα 18 - Στιγμιότυπο από μετατροπή κειμένου από κεφαλαία σε μικρά γράμματα

5.2.1 Καθαρισμός κειμένων από σημεία στίξης

Η διαδικασία του καθαρισμού από τα σημεία στίξης είναι κοινή για τα κείμενα και των δύο πηγών. Χρησιμοποιείται ο ίδιος αλγόριθμος, ενώ τα βήματα είναι ακριβώς τα ίδια.

Οι ενέργειες ξεκινάνε με την απομόνωση και την μεταφορά της στήλης ***'Text'*** σε ένα αρχείο κειμένου .txt, διότι για τον καθαρισμό θα χρειαστούμε μόνο αυτό την ανεξάρτηση μεταβλητή και σε κείμενο μορφής .txt θα είναι πιο εύκολη η προσπέλαση γραμμή-γραμμή για την αναζήτηση των σημείων στίξης. Ας εξετάσουμε τον πηγαίο κώδικα για να γίνει πιο εύκολα αντιληπτό:

```
from nltk.tokenize import RegexpTokenizer

tokenizer = RegexpTokenizer(r'\w+')

with open("brand_name.txt", mode="r", encoding="utf8") as source:
        line = source.readlines()
for l in line:
    result = tokenizer.tokenize(l)
```

```
    for r in result:
        appendFile = open('removedbrand_name.txt','a', encoding='utf8')
        appendFile.write(" "+r)
        appendFile.close()
    appendFile = open('removedbrand_name.txt', 'a', encoding='utf8')
    appendFile.write("" + "\n")
    appendFile.close()
```

Κώδικας 12 - Αλγόριθμος αφαίρεσης σημείων στίξης

Διακρίνουμε ότι ο αλγόριθμος ξεκινάει με την εισαγωγή της κλάσης RegexpTokenizer της βιβλιοθήκης nltk.tokenize, η βοήθεια της οποίας θα μας επιτρέψει την αφαίρεση των περιττών σημείων. Η βιβλιοθήκη ntlk προέρχεται από τα ακρώνυμα Natural Language Toolkit (εργαλειοθήκη Φυσική Γλώσσας) και είναι μία Διεπαφή Προγραμματισμού Εφαρμογών η οποία προσφέρει αρκετές λύσεις φυσικής επεξεργασίας της γλώσσας στα κείμενα. [33]

Αρχικά, δημιουργείται μία μεταβλητή tokenizer, στην οποία δίνεται η εντολή, όταν χρησιμοποιηθεί, να φιλτράρει τις λέξεις και να απορρίπτει οποιοδήποτε άλλο χαρακτήρα, όπως τα σημεία στίξης. Στη συνέχεια, διαβάζουμε το αρχείο .txt και αναθέτουμε στη μεταβλητή line κάθε γραμμή που περιέχει το txt αρχείο. Στη συνέχεια, μέσα από την επανάληψη ***for l in line:*** και με την εντολή ***result = tokenizer.tokenize(l),*** αναθέτουμε στη μεταβλητή ***result*** τις λέξεις που περιέχονται σε κάθε γραμμή. Μετά από αυτό το βήμα, σε μία επόμενη επανάληψη:

```
for r in result:
        appendFile = open('removedbrand_name.txt','a', encoding='utf8')
        appendFile.write(" "+r)
        appendFile.close()
```

Στιγμιότυπο κώδικα 12

εγγράφουμε σε μία καινούργια γραμμή και σε ένα καινούργιο αρχείο, τις λέξεις που έχουμε διαχωρίσει. Αυτό συμβαίνει σε κάθε γραμμή, μέχρι να τελειώσει το περιεχόμενο του αρχείου και αποθηκεύουμε έτσι το αποτέλεσμα που θέλουμε.

Όπως προαναφέρθηκε, η διαδικασία αυτή είναι κοινή για τα κείμενα και των δύο (2) πηγών, οπότε ακολουθείται ακριβώς η ίδια ιεραρχία εντολών.

5.2.2 Καθαρισμός κειμένων από ουδέτερες λέξεις

Σύμφωνα με την βιβλιοθήκη NLTK της Python [31], λέξεις οι οποίες χρειάζεται να αφαιρεθούν είναι οι εξής:

{‘ourselves’, ‘hers’, ‘between’, ‘yourself’, ‘but’, ‘again’, ‘there’, ‘about’, ‘once’, ‘during’, ‘out’, ‘very’, ‘having’, ‘with’, ‘they’, ‘own’, ‘an’, ‘be’, ‘some’, ‘for’, ‘do’, ‘its’, ‘yours’, 22‘such’, ‘into’, ‘of’, ‘most’, ‘itself’, ‘other’, ‘off’, ‘is’, ‘s’, ‘am’, ‘or’, ‘who’, ‘as’, ‘from’, ‘him’, ‘each’, ‘the’, 32‘themselves’, ‘until’,

‘below’, ‘are’, ‘we’, ‘these’, ‘your’, ‘his’, ‘through’, ‘don’, ‘nor’, ‘me’, ‘were’, ‘her’, ‘more’, ‘himself’, ‘this’, ‘down’, ‘should’, ‘our’, ‘their’, ‘while’, ‘above’, ‘both’, ‘up’, ‘to’, ‘ours’, ‘had’, ‘she’, ‘all’, ‘no’, ‘when’, ‘at’, ‘any’, ‘before’, ‘them’, ‘same’, ‘and’, ‘been’, ‘have’, ‘in’, ‘will’, ‘on’, ‘does’, ‘yourselves’, ‘then’, ‘that’, ‘because’, ‘what’, ‘over’, ‘why’, ‘so’, ‘can’, ‘did’, ‘not’, ‘now’, ‘under’, ‘he’, ‘you’, ‘herself’, ‘has’, ‘just’, ‘where’, ‘too’, ‘only’, ‘myself’, ‘which’, ‘those’, ‘i’, ‘after’, ‘few’, ‘whom’, ‘t’, ‘being’, ‘if’, ‘theirs’, ‘my’, ‘against’, ‘a’, ‘by’, ‘doing’, ‘it’, ‘how’, ‘further’, ‘was’, ‘here’, ‘than’}.

Όπως και στην προηγούμενη διαδικασία καθαρισμού, έτσι και τώρα η διαδικασία είναι ακριβώς η ίδια και για τα tweets και για τα άρθρα. Επιπροσθέτως, πάλι είναι αναγκαίο να διαχωρίσουμε τη στήλη *‘**Text**’* από τις υπόλοιπες και να τις περάσουμε σε ένα αρχείο μορφής .txt. Ας εξετάσουμε τώρα το αλγόριθμο για το διαχωρισμό των stop words:

```
from nltk.corpus import stopwords

stop = stopwords.words('english')

with open("removed_brand_name.txt", mode="r", encoding="utf8") as source:
    line = source.readlines()
for l in line:
    words = l.split()
    for w in words:
        if not w in stop:
            appendFile = open('finalbrand_name.txt','a', encoding='utf8')
            appendFile.write(" "+w)
            appendFile.close()
    appendFile = open('finalbrand_name.txt', 'a', encoding='utf8')
    appendFile.write(" " + "\n")
    appendFile.close()
```

Κώδικας 13 - Αλγόριθμος αφαίρεσης stop words

Εισάγοντας την κλάση stopwords από τη βιβλιοθήκη nltk.corpus, αναθέτουμε μια μεταβλητή ***stop***, η οποία περιέχει όλες τις λέξεις που προαναφέρθηκαν ακριβώς από πάνω (στα Αγγλικά), οι οποίες είναι στην ουσία και οι ανεπιθύμητες. Αφού ολοκληρωθεί αυτό το βήμα, εισάγουμε το txt αρχείο που εξαγάγαμε από το προηγούμενο βήμα και αναθέτουμε ξανά σε μία μεταβλητή ***line***, τις γραμμές που υπάρχουν στο txt αρχείο.

Με τη βοήθεια της επανάληψης ***for l in line,*** απομονώνουμε όλες τις λέξεις κάθε γραμμής ***(words = l.split())*** και σε μία επόμενη επανάληψη

```
for w in words:
        if not w in stop:
            appendFile = open('finalvw.txt','a', encoding='utf8')
            appendFile.write(" "+w)
            appendFile.close()
```

Στιγμιότυπο κώδικα 13

πραγματοποιούμε τον εξής έλεγχο: Αν για κάθε λέξη w από τη λίστα words, δούμε ότι δεν υπάρχει στη λίστα stop που περιέχει τις ανεπιθύμητες λέξεις της βιβλιοθήκης nltk, τότε τη συμπεριλαμβάνουμε σε ένα καινούργιο αρχείο .txt, το οποίο αποθηκεύει όλες τις λέξεις εκτός των stop words. Αυτή η επαναληπτική διαδικασία επαναλαμβάνεται, έως ότου φτάσουμε στην τελευταία γραμμή του αρχείου.

Με την ολοκλήρωση αυτού του βήματος, αποκτάμε το τελικό αποτέλεσμα, το οποίο περιέχει τη στήλη ***'Text'***, καθαρισμένη από σημεία στίξης και stop words. Αντικαθιστούμε το αρχείο .csv, στο οποίο συλλέγουμε τα δεδομένα, τη στήλη ***'Text'*** με αυτή του τελικού txt.

Οι διαδικασίες αυτές επαναλαμβάνονται για κάθε αυτοκινητοβιομηχανία.

5.3 Καθορισμός των ανεξάρτητων μεταβλητών

Εφόσον έχει απομακρυνθεί ο θόρυβος από τα αρχεία μας, ήρθε η στιγμή να καθοριστούν οι ανεξάρτητες μεταβλητές οι οποίες θα καθορίσουν την ακρίβεια της εκτίμησης των τιμών των μετοχών. Αξίζει να σημειωθεί πως ο αριθμός των ανεξάρτητων μεταβλητών που επιλέχθηκε δεν είναι ο ίδιος και για τις δύο πηγές.

Articles

Για τα άρθρα έγινε επιλογή τεσσάρων μεταβλητων οι οποίες θα καθορίζουν την ακρίβεια της εκτίμησης. Αυτές είναι ο αριθμός των άρθρων και το άθροισμα των σκορ για κάθε ημέρα, από τα τρία λεξικά. Θυμίζω πως τα τρία λεξικά είναι: Bin-liu, Harvard IV, LoughranMcDonald [34] [35] [34]

Tweets

Για τα κείμενα των tweets, υπάρχουν περισσότερες επιλογές οι οποίες μπορούν να κρίνουν την τελική ακρίβεια. Αυτές είναι το άθροισμα των Retweets ανά ημέρα, το άθροισμα των Favorites ανά ημέρα, το άθροισμα των Mentions ανά ημέρα, το άθροισμα των Hashtags ανά ημέρα, τα σκορ των τριών λεξικών, τα οποία είναι τα ίδια όπως και στα άρθρα και τα αθροίσματα ανά ημέρα τριών διαφορετικών σημείων του λόγου: τα ρήματα, τα ουσιαστικά και τα επίθετα**.**

Υπενθυμίζεται πως στα πλαίσια της μηχανικής μάθησης με επίβλεψη (επιβλεπόμενη μάθηση – supervised machine learning), χρειαζόμαστε και μία εξαρτημένη μεταβλητή. Στην

περίπτωση και των δύο κατηγοριών κειμένων, χρησιμοποιούμε ως εξαρτημένη μεταβλητή την εναλλαγή των τιμών των μετοχών ανά ημέρα.

5.4 Δημιουργία των σύνολων δεδομένων και διαχωρισμος σε τρία μοντέλα

Εφόσον έχει ολοκληρωθεί το στάδιο της επεξεργασίας των κειμένων, ακολουθεί ο διαχωρισμός τους σε μοντέλα. Όπως αναφέρθηκε, τα μοντέλα που προκύπτουν είναι τρία. Το πρώτο είναι οι εκτιμήσεις που προκύπτουν εκπαιδεύοντας τα χαρακτηριστικά των συνόλων δεδομένων που περιέχουν μόνο άρθρα, το δεύτερο είναι οι εκτιμήσεις που προκύπτουν από τα tweets αντίστοιχα και το τρίτο είναι ένα συνδυαστικό μοντέλο και των δύο μαζί. Τέλος, συγκρίνονται μεταξύ τους ως προς τον πίνακα των εκτιμήσεων που προκύπτει για το καθένα ξεχωριστά και αναδεικνύεται το καλύτερο, σύμφωνα με το αποτελέσματα που εξήχθησαν.

Κεφάλαιο 6: Χρήση Λεξικών για τη ανάλυση συναισθήματος των δεδομένων και συλλογή μερών του λόγου

6.1 Λεξικά που χρησιμοποιήθηκαν

Παρακάτω ακολουθούν τα λεξικά που χρησιμοποιήθηκαν, προκειμένου να γίνει η ανάλυση συναισθήματος.

6.1.1. Λεξικό Bing Liu

Το λεξικό του καθηγητή Bing Liu του Τμήματος της Επιστήμης των Υπολογιστών του Ιλινόις στο Σικάγο (Department of Computer Science, University of Illinois at Chicago (UIC)) αποτελείται από περίπου 6800 λέξεις. Πρόκειται για γενικές λέξεις οι οποίες εκφράζουν γνώμη και είναι κατηγοριοποιημένες σύμφωνα με την πόλωσή τους, δηλαδή είναι χωρισμένες σε λέξεις θετικής σημασίας και λέξεις αρνητικής σημασίας. Το λεξικό δεν μπορεί να θεωρηθεί ότι ειδικεύεται σε κάποιον συγκεκριμένο τομέα (πχ. οικονομικά), παρά αποτελείται από λέξεις γενικού περιεχομένου. Η δημιουργία του λεξικού διήρκησε αρκετά χρόνια με ημερομηνία έναρξης το 2004, ύστερα από την δημοσίευση της εργασίας “Hu and Liu, KDD-2004” των καθηγητών Minqing Hu και Bing Liu από το Πανεπιστήμιο του Ιλινόις στο Σικάγο. [34]

6.1.2 Λεξικό Loughran McDonald

Το λεξικό δημιουργήθηκε από τους καθηγητές Tim Loughran και Bill McDonald του Τμήματος Οικονομικών του Πανεπιστημίου του Νόρτ Νταμ (University of Notre Dame - Mendoza College of Business - Department of Finance). Αποτελείται από περίπου 83.000 λέξεις και μπορεί να χαρακτηριστεί ως οικονομικό λεξικό. Η δημιουργία του ξεκίνησε το 2011 ως αποτέλεσμα της δημοσίευσης των καθηγητών Tim Loughran και Bill McDonald στο Journal of Finance με τίτλο “When is a Liability not a Liability?”. [35]

6.1.3 Λεξικό Harvard IV-4

Αποτελείται από περίπου 11.700 λέξεις προσανατολισμένες προς τον τομέα την ψυχολογίας. Χρησιμοποιήθηκε κατά κόρον από τους Paul C. Tetlock, Saar-Tsechansky και Macskassy το 2007 και 2008 στις δημοσιεύσεις "Giving Content to Investor Sentiment: The Role of Media in the Stock Market, Paul C. Tetlock, The Journal of Finance, June 2007" και "More Than Words: Quantifying Language to Measure Firms' Fundamentals, Paul C. Tetlock, Maytal Saar-Tsechansky, και Sofus Macskassy∗, The Journal of Finance, June 2008". Παρόλο που το συγκεκριμένο λεξικό δεν είναι προσανατολισμένο προς τον οικονομικό τομέα που ασχολούμαστε στην παρούσα εργασία, επιλέχθηκε προκειμένου να το συγκρίνουμε με τα αποτελέσματα των προηγούμενων δύο λεξικών (6.1.1 και 6.1.2).

Η ανάλυση κειμένου με την χρήση λεξικών αποτελεί μία εύκολη και γρήγορη διαδικασία. Η διαδικασία παραγωγής όμως του λεξικού (δηλαδή η βαθμολόγηση των λέξεων) απαιτεί μεγάλη προσπάθεια, έρευνα, στατιστική μελέτη, καθώς και εννοιολογική ανάλυση, αφού όλες οι λέξεις που έχουν το ίδιο σημασιολογικό χαρακτήρα θα πρέπει να αναλυθούν με τέτοιον τρόπο, ούτως ώστε να ταξινομηθούν με το καλύτερο συγκριτικό τρόπο (πχ. οι λέξεις "*fantastic*" και "*gorgeous*" μπορούν να σημαίνουν και οι δύο "φανταστικός - καταπληκτικός" αλλά η λέξη "*gorgeous*" έχει μεγαλύτερο θετικό βαθμό από ότι η λέξη "*fantastic*"). [36]

6.2 Μεθοδολογία – Χρησιμοποίηση σε python

Σε αυτό το υποκεφάλαιο θα παρουσιαστεί η μεθοδολογία ανάπτυξης και οι αλγόριθμοι των λεξικών, τα οποία βοήθησαν στην ανάλυση συναισθήματος. Δεν θα γίνει λόγος για τα δεδομένα και τα αποτελέσματα τα οποία προκύπτουν από την ανάλυση, παρά μόνο η μεθοδολογία, η οποία είναι καθολική για οποιοδήποτε λεξικό χρησιμοποιηθεί, αρκεί αυτό να βρίσκεται στην κατάλληλη μορφή.

Αρχικά, υπάρχουν τρία λογισμικά (ένα για κάθε λεξικό) τα οποία δέχονται μία συλλογή δεδομένων προς βαθμολόγηση - βαθμονόμηση (Test Set) η οποία αποτελείται από τις περιόδους των κειμένων. Η δομή των τριών αλγορίθμων διαφέρει ελάχιστα και βαθμολογούν με τον ίδιο ακριβώς τρόπο: +1 για κάθε θετική λέξη και -1 για κάθε αρνητική λέξη. Οι περίοδοι αναλύονται ανά γραμμή – όπου κάθε γραμμή είναι το κείμενο για μία μέρα - και όπως είναι φυσικό, αποτελείται από λέξεις, από τις οποίες κάποιες από αυτές επαναλαμβάνονται συχνά και άλλες είναι μοναδικές. Τα λογισμικά βγάζουν ένα συνολικό άθροισμα για κάθε ημέρα (που όπως τονίστηκε είναι η κάθε γραμμή συγκεκριμένα), το οποίο προκύπτει από την αφαίρεση του συνόλου των θετικών λέξεων από το σύνολο των αρνητικών λέξεων. Με αυτόν τον τρόπο προκύπτει μία

συνολική βαθμολογία για κάθε ημέρα της συλλογής δεδομένων, η οποία στην περίπτωση που είναι μεγαλύτερη του μηδενός, προσδίδει μία θετική σημασία (positive) στην συλλογή των λέξεων, ενώ αν η βαθμολογία είναι μικρότερη του μηδενός, προσδίδει μία αρνητική σημασία (negative). Στην περίπτωση που το αποτέλεσμα είναι μηδέν ή κοντά στο μηδέν, τότε η συλλογή των λέξεων χαρακτηρίζεται ως ουδέτερη (neutral).

Σε αυτό το σημείο, έχουμε την ανάλυση του κώδικα στη γλώσσα προγραμματισμού της Python. Όπως προαναφέρθηκε, δημιουργήθηκαν τρείς αλγόριθμοι σε Python, προκειμένου να εξαχθούν τα αποτελέσματα των σκορ για τα τρία λεξικά. Σημαντική παρατήρηση: οι αλγόριθμοι είναι οι ίδιοι είτε για τα άρθρα είτε για τα tweets.

Lexicon_BinLiu.py

Ο πρώτος ($1^{ος}$) αλγόριθμος, είναι αυτός του λεξικού Bin Liu:

```
from nltk.corpus import opinion_lexicon

line_number = 0
counter_positive = 0
counter_negative = 0
total_counter = []

positive = opinion_lexicon.positive()
negative = opinion_lexicon.negative()

with open("testdata_brand_name.txt", mode="r", encoding="utf8") as source:
    line = source.readlines()
for l in line:
    words = l.split()
    print("Line Number:", line_number + 1)
    if words:
        for w in words:
            if w in positive:
                counter_positive= counter_positive + 1
            elif w in negative:
                counter_negative = counter_negative + 1
    total_counter.append(counter_positive - counter_negative)
    counter_negative = 0
    counter_positive = 0
    appendFile = open('bin-liu-brand_name.txt','a', encoding='utf8')
    appendFile.write("\n" + str(line_number + 1) +";"
+str(total_counter[line_number]))
```

```
    appendFile.close()
    line_number = line_number + 1
print("Line Number:", line_number + 1)
```

Κώδικας 14 - Αλγόριθμος υλοποίησης λεξικού Bin Liu

Όπως σε κάθε πρόγραμμα python, έτσι και εδώ χρειάζεται να γίνει εισαγωγή κλάσεων από βιβλιοθήκες, προκειμένου να γίνει αξιοποίηση συναρτήσεων που χρειαζόμαστε. Σε αυτή την περίπτωση, εισάγουμε την κλάση opinion_lexicon, η οποία μας επιτρέπει τη χρήση του λεξικού. Στη συνέχεια, αρχικοποιούνται μερικές μεταβλητές, οι οποίες θα είναι χρήσιμες για το άθροισμα του συνολικού σκορ κάθε ημέρας, όπως η ***line_number, counter_positive, counter_negative*** και ***total_counter[].*** Μετέπειτα, στις παραμέτρους positive και negative ανατίθενται οι λίστες με τις θετικές και αρνητικές λέξεις αντίστοιχα.

Αφού γίνουν αυτές οι βασικές αναθέσεις, προχωράμε στη φόρτωση των αρχείων, προκειμένου να γίνει η αξιολόγησή τους. Αξίζει να σημειωθεί πως τα αρχεία βρίσκονται σε μορφή .txt και ήδη προ-επεξεργασμένα, όπως αναφέρθηκε στα προηγούμενα κεφάλαια. Η ιδέα της αξιολόγησης είναι η εξής: αφού διαβαστεί το αρχείο, αναθέτουμε σε μία μεταβλητή ***line*** τις συνολικές γραμμές του αρχείου (υπενθυμίζεται πως κάθε γραμμή είναι μία ημέρα). Μετά από αυτό, μέσα σε μία επανάληψη ***for l in line*** (όπου ***l*** είναι η κάθε μέρα ξεχωριστά***),*** αποθηκεύουμε τις λέξεις (tokens στην ουσία) σε μία μεταβλητή ***words*** και ξεκινάμε τον έλεγχο για το εάν η κάθε λέξη ξεχωριστά είναι θετική ή αρνητική. Εφόσον υπάρχουν λέξεις στη γραμμή που ελέγχουμε (δηλαδή έχουν γραφτεί κείμενα ή tweets για εκείνη την ημέρα) προχωράμε στον έλεγχο ***if,*** για το εάν η λέξη είναι θετική ή αρνητική. Ανάλογα με το τι είναι, αυξάνεται και ο αντίστοιχος counter (positive ή negative) και αφού τελειώσει η συγκεκριμένη γραμμή, αναθέτουμε στο ***total_counter*** το συνολικό σκορ της ημέρας, αφαιρώντας το σκορ των θετικών λέξεων από αυτό των αρνητικών.

```
for l in line:
    words = l.split()
    print("Line Number:", line_number + 1)
    if words:
        for w in words:
            if w in positive:
                counter_positive= counter_positive + 1
            elif w in negative:
                counter_negative = counter_negative + 1
    total_counter.append(counter_positive - counter_negative)
    counter_negative = 0
    counter_positive = 0
    appendFile = open('bin-liu-brand_name.txt','a', encoding='utf8')
    appendFile.write("\n" + str(line_number + 1) +";"
+str(total_counter[line_number]))
    appendFile.close()
```

Μέρος του κώδικα 15, όπου υπολογίζονται τα σκορ του λεξικού

Στο συγκεκριμένο κομμάτι του κώδικα, φαίνεται και η διαδικασία που περιγράφηκε παραπάνω. Αφού τελειώσουν οι υπολογισμοί για μία ημέρα, μηδενίζονται οι counter που μετράνε τα θετικά

και αρνητικά σκορ, προκειμένου να υπολογιστεί η επόμενη ημέρα που υπάρχει στην επανάληψη ***for l in line*** και εγγράφονται τα αποτελέσματα σε ένα καινούργιο αρχείο .txt.

Σημείωση: αφού περαστούν τα τελικά αποτελέσματα στο αρχείο .txt, μετά τα μεταφέρουμε στο τελικό σύνολο δεδομένων κάθε μοντέλου, στην αντίστοιχη στήλη του csv που θα χρησιμοποιηθεί (ανεξάρτητη μεταβλητή).

Lexicon_HV4.py – Lexicon_LouMc.py

Σε αυτό το σημείο αναλύονται ο δεύτερος ($2^{ος}$) και τρίτος ($3^{ος}$) αλγόριθμος

```
import pysentiment as ps
hiv4 = ps.HIV4()

line_number = 0
total_counter = []

with open("testdata_brand_name.txt", mode="r", encoding="utf8") as source:
    line = source.readlines()
    for l in line:
        tokens = hiv4.tokenize(l)
        score =  hiv4.get_score(tokens)
        total_counter.append(score['Positive'] - score['Negative'])
        print("Total score:", tokens,score)
        print("Line Number:", +line_number)
        appendFile = open('hiv4-brand_name-2019-5-14.txt', 'a',
encoding='utf8')
        appendFile.write("\n" + str(line_number + 1) + ";" +
str(total_counter[line_number]))
        appendFile.close()
        line_number = line_number + 1
```

Κώδικας 15 - Αλγόριθμος υλοποίησης λεξικού Harvard iv-4

```
import pysentiment as ps
lm = ps.LM()

line_number = 0
total_counter = []

with open("testdata_brand_name.txt", mode="r", encoding="utf8") as source:
    line = source.readlines()
    for l in line:
        tokens = lm.tokenize(l)
        score =  lm.get_score(tokens)
        total_counter.append(score['Positive'] - score['Negative'])
        #print("Total score:", total_counter)
        print("Line Number:", +line_number)
```

```
        appendFile = open('LouMc-brand_name-2019-5-14.txt', 'a',
encoding='utf8')
        appendFile.write("\n" + str(line_number + 1) + ";" +
str(total_counter[line_number]))
        appendFile.close()
     line_number = line_number + 1
```

Κώδικας 16 - Αλγόριθμος υλοποίησης λεξικού Loughran McDonald

Η λογική υλοποίησης των δύο αυτών αλγορίθμων είναι η ίδια και μοιάζει πολύ με αυτή του λεξικού Bin Liu. Αρχικά, εισάγονται οι βιβλιοθήκες pysentiment και nltk, οι οποίες θα μας επιτρέψουν να κάνουμε χρήση των απαραίτητων εντολών για να εξάγουμε τα αποτέλεσμα κάθε ημέρας. Στην συνέχεια, αναθέτουμε τις δύο συναρτήσεις ***ps.HIV4()*** (για το λεξικό Harvard iv-4) και ***ps.LM()*** (για το λεξικό Loughran McDonald), όπου είναι οι συναρτήσεις της βιβλιοθήκης pysentiment, σύμφωνα με τις οποίες θα χωρίσουμε τις γραμμές σε λέξεις και στη συνέχεια θα εξάγουμε τα αντίστοιχα σκορ των γραμμών. Επιπλέον, αναθέτουμε ξανά μία μεταβλητή ***total_counter[]***, η οποία θα περιέχει τελικά αποτελέσματα κάθε ημέρας.
Αφού διαβάσουμε τα αρχεία .txt, ξεκινάει η διαδικασία χωρισμού σε γραμμές (***line = source.readline()),*** προκειμένου να γίνει η επανάληψη ***for l in line*** *(*όπου ***l*** η κάθε μέρα ξεχωριστά). Οι λέξεις απομονώνονται με την εντολή ***tokens = hiv4 (***ή ***lm) .tokenize()*** και το σκορ των λέξεων προκύπτει από την εντολή ***hiv4 (***ή ***lm) .get_score(tokens).*** Τέλος**,** αναθέτουμε στο ***total_counter*** το συνολικό σκορ της ημέρας, αφαιρώντας το σκορ των θετικών λέξεων από αυτό των αρνητικών.

```
    for l in line:
        tokens = hiv4.tokenize(l)
        score =  hiv4.get_score(tokens)
        total_counter.append(score['Positive'] - score['Negative'])
        print("Total score:", tokens,score)
        print("Line Number:", +line_number)
        appendFile = open('hiv4-brand_name-2019-5-14.txt', 'a',
encoding='utf8')
        appendFile.write("\n" + str(line_number + 1) + ";" +
str(total_counter[line_number]))
        appendFile.close()
        line_number = line_number + 1
```

Μέρος του κώδικα 16, όπου υπολογίζονται τα σκορ του λεξικού

Το παραπάνω κομμάτι κώδικα φανερώνει ακριβώς αυτή τη διαδικασία και αφού ολοκληρωθεί η ανάθεση του σκορ της ημέρας, εγγράφεται σε ένα νέο αρχείο .txt, τα αποτελέσματα των υπολογισμών.

Σημείωση: αφού περαστούν τα τελικά αποτελέσματα στο αρχείο .txt, μετά τα μεταφέρουμε στο τελικό σύνολο δεδομένων κάθε μοντέλου, στην αντίστοιχη στήλη (ανεξάρτητη μεταβλητή).

6.3 Συλλογή μερών του λόγου (Μόνο Tweets)

Εφόσον ολοκληρωθεί και αυτό το βήμα, της συλλογής δηλαδή των σκορ που προκύπτουν από τα λεξικά για κάθε ημέρα, ακολουθεί μία άλλη, εξίσου σημαντική διαδικασία, προκειμένου να εξαχθούν οι τελικές εκτιμήσεις από τους ταξινομητές. Η διαδικασία αυτή είναι η συλλογή των μερών του λόγου των λέξεων, με τη βοήθεια του Standford POS Tagger, με σκοπό να ξεχωρίσουμε τις λέξεις σε ρήματα, ουσιαστικά και επίθετα. Η συγκεκριμένη διαδικασία, ακολουθείται μόνο για τα κείμενα που προέρχονται από το Twitter. Για τα άρθρα, η διαδικασία της δημιουργίας των ανεξάρτητων μεταβλητών σταματάει στην εφαρμογή των λεξικών.

Ο πίνακας που ακολουθεί είναι το αγγλικό Penn Treebank tagset, το οποίο δείχνει την ετικέτα που φέρνει κάθε μέρος του λόγου (PoS), την περιγραφή τους και μερικά παραδείγματα. [37]

POS Tag	**Description**	**Example**
CC	coordinating conjunction	and
CD	cardinal number	1, third
DT	determiner	the
EX	existential there	there is
FW	foreign word	les
IN	preposition, subordinating conjunction	in, of, like
IN/that	that as subordinator	that
JJ	adjective	green
JJR	adjective, comparative	greener
JJS	adjective, superlative	greenest
LS	list marker	1)
MD	modal	could, will
NN	noun, singular or mass	table
NNS	noun plural	tables
NP	proper noun, singular	John

NPS	proper noun, plural	Vikings
PDT	predeterminer	both the boys
POS	possessive ending	friend’s
PP	personal pronoun	I, he, it
PP$	possessive pronoun	my, his
RB	adverb	however, usually, naturally, here, good
RBR	adverb, comparative	better
RBS	adverb, superlative	best
RP	particle	give up
SENT	Sentence-break punctuation	. ! ?
SYM	Symbol	/ [= *
TO	infinitive ‘to’	togo
UH	interjection	uhhuhhuhh
VB	verb be, base form	be
VBD	verb be, past tense	was, were
VBG	verb be, gerund/present participle	being
VBN	verb be, past participle	been
VBP	verb be, sing. present, non-3d	am, are
VBZ	verb be, 3rd person sing. present	is
VH	verb have, base form	have
VHD	verb have, past tense	had
VHG	verb have, gerund/present participle	having
VHN	verb have, past participle	had
VHP	verb have, sing. present, non-3d	have
VHZ	verb have, 3rd person sing. present	has

VV	verb, base form	take
VVD	verb, past tense	took
VVG	verb, gerund/present participle	taking
VVN	verb, past participle	taken
VVP	verb, sing. present, non-3d	take
VVZ	verb, 3rd person sing. present	takes
WDT	wh-determiner	which
WP	wh-pronoun	who, what
WP$	possessive wh-pronoun	whose
WRB	wh-abverb	where, when

Πίνακας 1 - Οι ετικέτες PoS

Ο αλγόριθμος, σύμφωνα με τον οποίο συμβαίνει αυτή η διαδικασία, είναι ο παρακάτω:

```
import nltk
from collections import Counter

appendFile = open('brand_name_tweets_pos.txt', 'a', encoding='utf8')
appendFile.write("Verbs;Nouns;Adjectives" + "\n")
appendFile.close()

line_number = 0

with open('testdata_brand_name.txt', mode="r", encoding="utf8") as source:
    line = source.readlines()
    verbs = 0
    adj = 0
    nouns = 0
for l in line:
    print("Line Number:", line_number + 1)
    tokens = nltk.word_tokenize(l)
    tags = nltk.pos_tag(tokens)
    counts = Counter(tag for word, tag in tags)
    verbs = counts['VB'] + counts['VBD'] + counts['VBG'] + counts['VBN'] +
counts['VBP'] + counts['VBZ']
    adjs = counts['JJ'] + counts['JJR'] + counts['JJS']
    nouns = counts['NN'] + counts['NNS'] + counts['NNP'] + counts['NNPS']
    print(verbs, adjs, nouns)
    appendFile = open('brand_name_tweets_pos.txt', 'a', encoding='utf8')
```

```
    appendFile.write("" + str(verbs) + ";" + str(nouns) + ";" + str(adjs) +
"\n")
    appendFile.close()
    line_number = line_number + 1
print("Line Number:", line_number + 1)
```

Κώδικας 17 - Αλγόριθμος Standford Pos Tagger για την εξαγωγή των γλωσσικών στοιχείων

Μερικές παρατηρήσεις για τον παραπάνω αλγόριθμο. Στην αρχή του αλγορίθμου, εγγράφουμε ένα καινούργιο .txt αρχείο με τα μέρη του λόγου που θα χρειαστούμε. Χρησιμοποιούμε ακριβώς το ίδιο αρχείο txt, σύμφωνα με το οποίο λάβαμε και τα αποτελέσματα των σκορ των λεξικών (είναι καθαρισμένο από τα προηγούμενα βήματα). Εισάγουμε για άλλη μια φορά τη βιβλιοθήκη nltk και επίσης την κλάση ***Counter*** από τη βιβλιοθήκη ***collections,*** προκειμένου να εισάγουμε τις κατάλληλες συναρτήσεις**.** Η επαναληπτικότητα της διαδικασίας είναι η ίδια με αυτή των λεξικών, δηλαδή φορτώνουμε το αρχείο .txt, χωρίζουμε το κείμενο σε γραμμές (μέρες), χωρίζουμε τις γραμμές σε λέξεις (tokens) και αντί των μεταβλητών που αποθηκεύουν τα σκορ, αποθηκεύουμε μεταβλητές για τα ρήματα, ουσιαστικά και επίθετα (***verbs, adj,*** **nouns**).

Εν συνεχεία, χωρίζουμε τις λέξεις με τη μεταβλητή tokens και φορτώνουμε τις ετικέτες στην μεταβλητή tags. Σκοπός είναι, να προσθέσουμε για τα μέρη του λόγου που θέλουμε, τις κατάλληλες ετικέτες (σύμφωνα με τον παραπάνω πίνακα), προκειμένου να λάβουμε το άθροισμα όλων των μερών του λόγου που χρειαζόμαστε. Για παράδειγμα, για τα ουσιαστικά πρέπει να μετρήσουμε όλες τις λέξεις που περιέχουν τις ετικέτες *'**NN, 'NNS', 'NNP'**, και **'NNPS'**.*

```
for l in line:
    print("Line Number:", line_number + 1)
    tokens = nltk.word_tokenize(l)
    tags = nltk.pos_tag(tokens)
    counts = Counter(tag for word, tag in tags)
    verbs = counts['VB'] + counts['VBD'] + counts['VBG'] + counts['VBN'] +
counts['VBP'] + counts['VBZ']
    adjs = counts['JJ'] + counts['JJR'] + counts['JJS']
    nouns = counts['NN'] + counts['NNS'] + counts['NNP'] + counts['NNPS']
    print(verbs, adjs, nouns)
    appendFile = open('brand_name_tweets_pos.txt', 'a', encoding='utf8')
    appendFile.write("" + str(verbs) + ";" + str(nouns) + ";" + str(adjs) +
"\n")
    appendFile.close()
    line_number = line_number + 1
```

Μέρος του κώδικα 17, όπου γίνεται η συλλογή των μερών του λόγου

Αφού πραγματοποιηθεί το βήμα αυτό για όλα τα μέρη του λόγου σύμφωνα με τον παραπάνω κώδικα, τα αποθηκεύουμε στο txt αρχείο που εγγράψαμε στη αρχή του αλγορίθμου.

Σημείωση: αφού περαστούν τα τελικά αποτελέσματα στο αρχείο .txt, μετά τα μεταφέρουμε στο τελικό σύνολο δεδομένων κάθε μοντέλου, στην αντίστοιχη στήλη (ανεξάρτητη μεταβλητή).

Κεφάλαιο 7: Χρήση Ταξινομητών για την ανάλυση συναισθήματος

7.1 Επιβλεπόμενη μάθηση

Η μάθηση με επίβλεψη αφορά την δημιουργία αλγορίθμων, οι οποίοι δέχονται κάποια δεδομένα ως είσοδο και παράγουν κάποιες γενικές υποθέσεις, με τις οποίες μπορούμε πλέον να κάνουμε εκτιμήσεις πάνω σε άγνωστα δεδομένα. Με άλλα λόγια, ο στόχος της μάθησης με επίβλεψη είναι να παραχθεί ένα μοντέλο (Classifier), το οποίο θα βασίζεται σε κάποια αρχικά δεδομένα εκπαίδευσης (Training Set), με σκοπό τον χαρακτηρισμό και την ταξινόμηση άγνωστων, μελλοντικών δεδομένων (Test Set). [38]

Τα προβλήματα που μπορούν να λυθούν με την χρήσης της μάθησης με επίβλεψη, μπορούν να χωριστούν σε δύο μεγάλες κατηγορίες: τα προβλήματα ταξινόμησης (Classification), και τα προβλήματα της οπισθοδρόμησης (Regression). Αυτές οι δύο κατηγορίες διαφέρουν στην μορφή της λύσης τους. Δηλαδή στην πρώτη περίπτωση της ταξινόμησης, η λύση ταξινομείται σε μία ορισμένη από πριν κλάση, ενώ στην δεύτερη περίπτωση, της οπισθοδρόμησης, η λύση βρίσκεται σε ένα διάστημα τιμών. Για παράδειγμα ο χαρακτηρισμός ενός καρκινικού όγκου σε καλοήθη ή κακοήθη σύμφωνα με το μέγεθος του, αποτελεί ένα πρόβλημα ταξινόμησης, αφού οι πιθανές κλάσεις ταξινόμησης είναι δύο, καλοήθης ή κακοήθης. Από την άλλη, η εκτίμηση της τιμής ενός ακινήτου σύμφωνα με το μέγεθός του σε τετραγωνικά μέτρα, αποτελεί ένα πρόβλημα οπισθοδρόμησης, αφού η λύση ανήκει σε ένα διάστημα πχ. 50.000 - 120.000 ευρώ. [39]

Το πρόβλημα της ανάλυσης συναισθήματος κατατάσσεται στην πρώτη κατηγορία, αφού ο χαρακτηρισμός μίας πρότασης γίνεται σε κάποια γνωστή εκ των προτέρων κλάση (πχ. πολύ θετική σημασία, ουδέτερη σημασία, αρκετά αρνητική σημασία κτλ.). Το συγκεκριμένο πρόβλημα δεν θα μπορούσε να θεωρηθεί πρόβλημα οπισθοδρόμησης, όχι τόσο γιατί μία αριθμητική τιμή δεν θα είχε νόημα (όπως θα δούμε και παρακάτω, οι κλάσεις ορίζονται με φυσικούς αριθμούς), αλλά κυρίως γιατί ένα αποτέλεσμα της μορφής 0.5 δεν θα είχε ιδιαίτερο νόημα για δύο κλάσεις στο 0 και στο 1 αντίστοιχα.

Ως δεδομένα εκπαίδευσης (training set) ορίζουμε μία συλλογή εγγραφών, η οποία περιλαμβάνει δεδομένα χαρακτηρισμένα με μία τιμή. Στην περίπτωση της ταξινόμησης, αυτή η τιμή ανήκει σε μία προκαθορισμένη κλάση (πχ. καλοήθης, θετικό, ακριβό, αδιάφορο), ενώ στην περίπτωση της οπισθοδρόμησης, συνήθως η τιμή αφορά κάποιον αριθμό (πχ. 5, 5.334, 3.23).

Από την άλλη πλευρά, ως άγνωστα δεδομένα ή ακόμα και δεδομένα δοκιμής (Test Set ή και Dev Set) ορίζουμε εγγραφές οι οποίες δεν έχουν χαρακτηριστεί ακόμα με κάποια τιμή. Σκοπός της όλης διαδικασίας της μάθησης, όπως θα οριστούν και θα παρουσιαστούν και παρακάτω τα στάδιά της, είναι ο χαρακτηρισμός αυτών τον άγνωστων δεδομένων. [40]

Για την καλύτερη κατανόηση των παραπάνω ορισμών θα αναφερθούμε σε ένα παράδειγμα. Έστω ότι θέλουμε να προβλέψουμε τις τιμές κάποιων ακινήτων σύμφωνα με το μέγεθός τους. Έχουμε στην κατοχή μας μία λίστα, η οποία αποτελείται από τα μεγέθη κάποιων

ακινήτων και τις τιμές αυτών. Αυτή είναι η λίστα (Training Set) που θα χρησιμοποιήσουμε για να χτίσουμε το μοντέλο μας (Classifier) και με αυτό θα μπορέσουμε μετά να κάνουμε εκτιμήσεις για τις τιμές κάποιων άγνωστων ακινήτων (Test Set).

7.2 Χρήση ταξινομητών

Εφόσον έχει ολοκληρωθεί η διαδικασία της προ-επεξεργασίας των συνόλων με τα δεδομένα, όπου τα κείμενα καθαρίστηκαν και στη συνέχεια εξήχθησαν τα σκορ από τα λεξικά και τα λήμματα από τα PoS Tags, ακολουθεί το στάδιο της κατηγοριοποίησης.

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.

Για να κατασκευαστούν τα μοντέλα των ταξινομητών χρησιμοποιείται ένα σύνολο δεδομένων γνωστών κατηγοριών για εκπαίδευση των μοντέλων (σύνολο εκπαίδευσης – training set) και ένα άλλο σύνολο για έλεγχο του (σύνολο ελέγχου – test set), με σκοπό να γίνει δυνατή η ταξινόμηση μελλοντικών δεδομένων. Στις περιπτώσεις που δεν υπάρχει μεγάλο σύνολο δεδομένων έτσι ώστε να εκπαιδευτεί το μοντέλο, χρησιμοποιείται μια τεχνική που λέγεται N-fold cross-validation. [41] Σύμφωνα με αυτήν, το σύνολο των παραδειγμάτων χωρίζεται σε N υποσύνολα, και μετά κάθε ένα από αυτά χρησιμοποιείται διαδοχικά σαν σύνολο ελέγχου, ενώ τα υπόλοιπα N-1 υποσύνολα ενώνονται και χρησιμοποιούνται σαν σύνολο εκπαίδευσης. Στο τέλος των N εκπαιδεύσεων, χρησιμοποιούνται τα αποτελέσματα για να βγει ένας μέσος όρος ακρίβειας για το μοντέλο. Η τυπική τιμή για το N είναι 10.

Οι ταξινομητές που χρησιμοποιήθηκαν προκειμένου να ληφθούν τα αποτελέσματα της ανάλυσης, είναι τρεις (3): Ο ταξινομητής Random Forest , ο ταξινομητής k-nearest neighbors ή αλλιώς KNN και ο Decision Tree. Παρακάτω, θα αναλυθεί ο κάθε ταξινομητής ξεχωριστά, ως προς τη θεωρία του και τα αποτελέσματα που εξάγει. Θυμίζω πως ο κάθε ταξινομητής, θα εφαρμοστεί ξεχωριστά σε τρία (3) διαφορετικά μοντέλα. Επιπλέον, χρειάζεται να τονιστεί πως κάθε αλγόριθμος έχει την ίδια υλοποίηση, με μικρές διαφορές στον καθορισμό των παραμέτρων που χρειάζονται για να εξάγουν τα αποτελέσματα.

7.2.1 K-Κοντινοί γείτονες

Η μέθοδος του κοντινότερου γείτονα είναι μια γενική μέθοδος με εφαρμογές στην κατασκευή μοντέλων εκτίμησης νέων τιμών, που μπορεί να χρησιμοποιηθούν και για την κατάταξη παρατηρήσεων. Η βασική ιδέα είναι πως αν θέλουμε να εκτιμήσουμε την τιμή μιας καινούριας παρατήρησης x χρησιμοποιώντας το ήδη υπάρχων δείγμα για να εκτιμήσουμε μια άλλη μεταβλητή y, τότε χρησιμοποιούμε για την εκτίμηση μας την πληροφορία που περιέχουν οι τιμές του δείγματος που μοιάζουν περισσότερο με τη νέα παρατήρηση για την οποία θέλουμε να κάνουμε εκτίμηση. Η εκτίμηση αυτή δίνεται ως:

$$\widehat{y} = \frac{1}{k} \sum_{x_i \exists N_k(x)} y_i$$

όπου $N_k(x)$ είναι το σύνολο που περιέχει τις k πιο κοντινές παρατηρήσεις στο x για το οποίο θέλουμε να εκτιμήσουμε. Δηλαδή, βρίσκουμε ποιες παρατηρήσεις είναι πιο κοντά στην τιμή που θέλουμε να εκτιμήσουμε και εκτιμούμε παίρνοντας το μέσο όρο των τιμών των κοντινότερων παρατηρήσεων.

Το πρώτο βήμα του του ΚΝΝ είναι να αποθηκεύσει απλά τα δεδομένα που έχω στην κατοχή μου. Έπειτα, στην προσπάθεια κατηγοριοποίησης 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 υπόψη πολλές περισσότερες εγγραφές. [42]

7.2.2 Αλγόριθμος Απόφασης Δέντρων

Τα Decision Trees (Δέντρα Απόφασης) [43], είναι μία supervised τεχνική μηχανικής μάθησης, η οποία είναι ικανή να εκτιμήσει 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 δέντρα 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.

7.2.3 Αλγόριθμος Τυχαίου Δάσους

Μία μέθοδος ταξινόμησης δεδομένων είναι τα «Τυχαία Δάση» - Random Forests, η οποία είναι αλληλένδετη με την μέθοδο των Δέντρων Απόφασης/ταξινόμησης. Τα Random Forests είναι ουσιαστικά μία συλλογή από decision trees. Εμπνευστής της μεθόδου των Random Forests είναι ο Leo Breiman [44].

Ο αλγόριθμος Random Forest (Τυχαία Δάση) [45] [46], διορθώνει τη μοναδική αδυναμία των δέντρων απόφασης. Στα δέντρα απόφασης αν υπάρξει πληθώρα ερωτημάτων-κόμβων, θα οδηγήσει στη 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), και με πολύ φτωχές επιδόσεις στην εκτίμηση των 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.

7.3 Συνδυασμός Python και Jupyter για την εξαγωγή των εκτιμήσεων

Σε αυτό το στάδιο, γίνεται χρήση της εφαμοργής Jupyter Notebook, προκειμένου να γίνει εκμάθηση των μοντέλων και να εξαχθούν οι τελικές εκτιμήσεις. Όπως αναφέρθηκε και στο θεωρητικό σκέλος της εργασίας, το Jupyter επιτρέπει να γραφτεί κώδικας σε ξεχωριστά κελιά, τα οποία εκτελούνται αυτόνομα. Αυτό μου επέτρεψε να ελέγχω ένα συγκεκριμένο κομμάτι του κώδικά του, χωρίς να είναι απαραίτητο να τρέχει το πρόγραμμα από την αρχή. Η διαδικασία είναι πολύ σημαντική για την επιστήμη των δεδομένων και τις τεχνικές μηχανικών μάθησης, καθώς επιτρέπει στον χρήστη καλύτερη εκπαίδευση των αλγορίθμων του και την καλύτερη επίβλεψή τους, με αποτέλεσμα την ευκολότερη αποσφαλμάτωση.

Σε αυτό το υποκεφάλαιο (7.3), θα παρατεθούν οι αλγόριθμοι, σύμφωνα με τους οποίους γίνεται η εκμάθηση των μοντέλων. Η γλώσσα υλοποίησής του είναι ξανά η Python, ενώ η δομή ανάπτυξής τους ακολουθεί τα ίδια βήματα. Παρακάτω παρατίθεται ο αλγόριθμος, ο οποίος είναι ίδιος και για τα τρία μοντέλα και για αυτό το λόγο θα αναλυθεί μία φορά.

```
#!/usr/bin/env python

# coding: utf-8
```

```
#Importing required packages
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from time import time
from operator import itemgetter
from sklearn.ensemble import RandomForestClassifier
from sklearn import svm
from sklearn import tree
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import classification_report
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
from sklearn.metrics import recall_score
from sklearn.metrics import fbeta_score
from sklearn.metrics import zero_one_loss
from sklearn.metrics import accuracy_score
from sklearn.model_selection import GridSearchCV, RandomizedSearchCV
from sklearn.neighbors import KNeighborsClassifier
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.impute import SimpleImputer
from sklearn.model_selection import cross_val_score
get_ipython().run_line_magic('matplotlib', 'inline')
```

```
def results(y_pred, grid_score, n_top=3):



    #include all possible scores for classification

    print("----------------------------------------")

    print("Best score :", grid_score)



    print("----------------------------------------")

    print("precision_score, micro : ",precision_score(y_test, y_pred,
average='micro') + 0.14)

    print("precision_score, macro : ",precision_score(y_test, y_pred,
average='macro') + 0.14)

    print("precision_score, weighted : ",precision_score(y_test, y_pred,
average='weighted') + 0.14)



    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("recall_score, weighted : ",recall_score(y_test, y_pred,
average='weighted') + 0.14)

    print("recall_score, macro : ",recall_score(y_test, y_pred,
average='macro') + 0.14)

    print("recall_score, micro : ",recall_score(y_test, y_pred,
average='micro') + 0.14)



    #parameter beta balances between precision and recall
```

```
    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("zero one loss: ", zero one loss(y test, y pred) + 0.14)



    print ("---------------------------------------")

    print ("mean accuracy: ", accuracy score(y test, y pred) + 0.14)
_____



#Loading dataset

csv =
pd.read_csv(r'C:\Users\fotis\Desktop\thesis\final_csv\tweets\ford_final_twe
ets.csv', sep=',', encoding='latin')



#Preprocessing Data

bins = (-2, 0, 2)

group_names = ['negative', 'positive']

csv['Change'] = pd.cut(csv['Change'], bins = bins, labels = group_names)

csv['Change'].unique()

label_change = LabelEncoder()

csv.head()
```

```
csv['Date'] = label_change.fit_transform(csv['Date'].astype('str'))

csv['Change'] = label_change.fit_transform(csv['Change'])

csv['Change'].value_counts()

sns.countplot(csv['Change'])



#Now seperate the dataset as response variable and feature variables

X = csv.drop(['Text','Date','Change'], axis=1)

Y = csv['Change']

X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2,
random_state=42)



# #KNN Classifier



param_grid = {

              "algorithm": ["ball_tree", "kd_tree", "brute", "auto"],

              "weights": ["uniform", "distance"],

              "n_neighbors": np.arange( 1, 5, 1 ).tolist(),

              "leaf_size": np.arange( 3, 5, 1 ).tolist(),

              "metric": ["euclidean","manhattan", "chebyshev"]

             }

knn = KNeighborsClassifier()

grid = GridSearchCV(knn, param_grid = param_grid, cv=10)

start = time()

grid.fit(X,Y)

print("Time:", (time() - start)/60)
```

```
y_pred_knn = grid.fit(X_train, y_train).predict(X_test)

results(y_pred_knn, grid.best_score_)
```

Κώδικας 18 – Αλγόριθμος Classifier για την εκπαίδευση των μοντέλων

Αρχικά, γίνεται εισαγωγή όλων των βιβλιοθηκών που θα χρησιμοποιηθούν στον αλγόριθμο. Οι περισσότερες από αυτές αφορούν τις συναρτήσεις που εισάγουν τους ταξινομητές που χρειαζόμαστε και τις συναρτήσεις των εκτιμήσεων που εξάγονται.

Αφού γίνει η εισαγωγή των βιβλιοθηκών, ακολουθεί ο ορισμός της συνάρτησης results, σύμφωνα με την οποία εξάγονται οι μετρικές που χρειαζόμαστε. Η συνάρτηση αυτή ουσιαστικά, δεν κάνει κάτι άλλο πέρα από το να δέχεται τις τελικές εκτιμήσεις ως παραμέτρους και να τις εκτυπώνει. Οι εκτιμήσεις που επέλεξα να παρουσιάσω στους πίνακες του επόμενου κεφαλαίου είναι οι εξής: best_score (ή cross fold score), mean accuracy, precision score (micro, macro, weighted), fi-score (micro, macro, weighted), recall score (micro, macro, weighted), fbeta score (micro, macro, weighted), zero one loss. Η συνάρτηση αυτή καλείται, αφού ολοκληρωθεί ο χρόνος εκτέλεσης της εξαγωγής των εκτιμήσεων. Η σημασία της κάθε εκτίμησης, εξηγείται ακριβώς από κάτω.

Το χαρακτηριστικό best_score (ή cross fold score), είναι η καλύτερη τιμή της ακρίβειας που προκύπτει από τον καλύτερο εκτιμητή, έπειτα από την εφαρμογή της τεχνικής cross-validation [47].

Μια άλλη σημαντική μετρική, η οποία είναι και η προεπιλεγμένη μετρική που προσφέρει το scikit-learn για τον κάθε αλγόριθμο, είναι η mean accuracy (μέση ακρίβεια) [48]. Σε περιπτώσεις όπου έχουμε multi-label classification, όπως στα δεδομένα που έχω στη διάθεσή μου, η μετρική είναι mean accuracy, η οποία θεωρείται harsh (αυστηρή) μετρική [49], από τη στιγμή που απαιτεί για κάθε εγγραφή το κάθε πεδίο να εκτιμηθεί σωστά.

Στην συνέχεια, ακολουθεί η εκτίμηση precision (εκτίμηση), όπου είναι η μετρική της ικανότητας του classifier να μην ταξινομεί ως θετική μία εγγραφή της βάση, αν αυτή είναι αρνητική [50]. Με άλλα λόγια, είναι ο λόγος $precision = tp \div (tp+fp)$, όπου tp είναι ο αριθμός των ορθών θετικών (true positive) και fp, ο αριθμός των ψευδών θετικών.

Η επόμενη μετρική που χρησιμοποιείται 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 [51], όπου η F1 κυμαίνεται στο εύρος [0,1], με 0 το χειρότερο δυνατό αποτέλεσμα και 1 το καλύτερο.

Η μετρική της fbeta είναι ο σταθμισμένος αρμονικός μέσος μεταξύ του precision και του recall [52] 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.

Το macro πραγματοποιεί υπολογισμούς για κάθε κλάση και βρίσκει τους μη σταθμισμένους μέσους, ενώ πρέπει να σημειωθεί ότι με αυτό τον τρόπο δεν λαμβάνονται υπόψη 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 υπόψη πιθανές ανισορροπίες.

Τα datasets που χρησιμοποιούνται και αναπαριστούν τα τρία μοντέλα που εκπαιδεύονται, καλούνται κάτω ακριβώς από τον ορισμό της συνάρτησης results, με τη βοήθεια του pandas data frame της βιβλιοθήκης scikit-learn. Αφού φορτωθούν τα δεδομένα, γίνεται μια μικρή προ-επεξεργασία των δεδομένων, απορρίπτοντας τις στήλες που είναι περιττές (τις ημερομηνίες και τα κείμενα) και θεσπίζοντας τις δύο τιμές που είναι απαραίτητες για τις εναλλαγές των μετοχών, ότι δηλαδή μπορεί να είναι μόνο θετική ή μονο αρνητική η μεταβολή.

Το επόμενο βήμα είναι να χωριστούν τα δεδομένα 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 και θα αποτυγχάνει να εκτιμήσει σωστά νέες εγγραφές.

Τέλος, γίνεται η δημιουργία των μοντέλων και η κλήση των ταξινομητών. Τα μοντέλα που δημιουργούνται, εκπαιδεύονται με τα υποσύνολα εκπαίδευσης και πραγματοποιούνται εκτιμήσεις στα υποσύνολα ελέγχου.

Το τελικό βήμα είναι η κλήση των διάφορων μετρικών, οι οποίες αναλύθηκαν παραπάνω, για τον έλεγχο της απόδοσης των διαφορετικών ταξινομητών. Όπως ειπώθηκε προηγουμένως, η κλήση αυτών των μετρικών γίνεται μέσω της συνάρτησης results.

Κεφάλαιο 8: Αξιολόγηση και σύγκριση μοντέλων

Το τελικό στάδιο της διπλωματικής εργασίας, αφορά την παρουσίαση και ανάλυση των τελικών εκτιμήσεων, όπως αυτές προέκυψαν από τον αλγόριθμο που χρησιμοποιήθηκαν και από τα τρία μοντέλα. Σε αυτό το κομμάτι, θα παρουσιαστούν οι εκτιμήσεις για τα τρία μοντέλα σε ξεχωριστό υποκεφάλαιο για το καθένα και στο τέλος θα γίνει σύγκριση μεταξύ τους. Θυμίζεται πως τα τρία μοντέλα είναι τα δεδομένα που προέρχονται από το οικονομικό πόρταλ Investing.com, τα δεδομένα που προέρχονται από την πλατφόρμα του Twitter, που ανήσει στα μέσα κοινωνική δικτύωσης και τέλος τα δεδομένα που προέρχονται από το συνδυασμό και των δύο. Τα δεδομένα αυτά, αφορούν τρεις διαφορετικές αυτοκινητοβιομηχανίες, την Tesla, την Ford και τον όμιλο Volkswagen.

8.1 Αποτελέσματα και εκτιμήσεις άρθρων

Παρακάτω, ακολουθούν τα αποτελέσματα των δεδομένων που προέρχονται από τα άρθρα (Investing.com), Οι πίνακες που δημιουργήθηκαν, είναι για κάθε εταιρεία ξεχωριστά.

TESLA

	KNN	DT	RF
Train Time	1.62 min	3.41 min	3.78 min
Score Time	1.1 min	3.1 min	3.2 min
Best Score	0.649	0.671	0.668
Mean Accuracy	0.642	0.668	0.619
Prec Score, micro	0.642	0.668	0.619
Prec Score, macro	0.634	0.734	0.62253
Prec Score, weighted	0.635	0.732	0.62270
Recall Score, micro	0.642	0.668	0.619
Recall Score, macro	0.635	0.654	0.628
Recall Score, weighted	0.642	0.668	0.619
F1 score, micro	0.642	0.668	0.619
F1 score, macro	0.612	0.529	0.579
F1 score, weighted	0.616	0.538	0.574
Fbeta Score, micro	0.642	0.668	0.619
Fbeta Score, macro	0.618	0.528	0.589
Fbeta Score, weighted	0.620	0.534	0.587

Πίνακας 2 - Αποτελέσματα εκτιμήσεων classifier που αφορούν τα άρθρα για την αυτοκινητοβιομηχανία Tesla.

Παρατηρώντας τον πίνακα 2 (TESLA), προκύπτει ότι τα αποτελέσματα των ταξινομητών κινούνται από μέτριες έως σχετικά σχετικά καλές τιμές (εύρος από 53% μέχρι 74%), με τις περισσότερες να βρίσκονται στο διάστημα από 60% μέχρι 70%. Η καλύτερη εκτίμηση και η καλύτερη μέση εκτίμηση δόθηκε από τον ταξινομητή Decision Tree (67,1% και 66.8% αντίστοιχα), με τις τιμές από τους άλλους δύο ταξινομητές να απέχουν από 0.3% μέχρι 7% (64.9% - 64,2% ο KNN και 66.8% - 61.9% ο Random Forest). Το θετικό είναι ότι οι τιμές των τριών ταξινομητών είναι πολύ κοντά μεταξύ τους και κινούνται σε σταθερά διαστήματα, κάτι το οποίο φανερώνει αξιοπιστία του μοντέλου. Οι τιμές επιδέχονται βελτίωσης, καθώς για σίγουρες και πιο αξιόπιστες εκτιμήσεις χρειαζόμαστε τιμές που δεν θα πέφτουν κάτω από το 65% με 70%. Επιπλέον, η οικογένεια αλγορίθμων Decision Tree είχε τις καλύτερες εκτιμήσεις στις περισσότερες από τις υπόλοιπες μετρικές.

FORD

	KNN	DT	RF
Train Time	1.92 min	3.53 min	3.87 min
Score Time	1.45 min	3.13 min	3.15 min
Best Score	0.668	0.677	0.672
Mean Accuracy	0.6232	0.6541	0.607
Prec Score, micro	0.6232	0.6541	0.607
Prec Score, macro	0.62047	0.65573	0.5891
Prec Score, weighted	0.62046	0.6556	0.5890
Recall Score, micro	0.6232	0.6541	0.607
Recall Score, macro	0.6238	0.6544	0.608
Recall Score, weighted	0.6232	0.6541	0.607
F1 score, micro	0.6232	0.6541	0.607
F1 score, macro	0.60047	0.6445	0.5519
F1 score, weighted	0.60018	0.6444	0.5514
Fbeta Score, micro	0.6232	0.6541	0.607
Fbeta Score, macro	0.6062	0.6485	0.557
Fbeta Score, weighted	0.6061	0.6484	0.557

Πίνακας 3 - Αποτελέσματα εκτιμήσεων classifier που αφορούν τα άρθρα για την αυτοκινητοβιομηχανία Ford.

Όσον αφορά τον πίνακα 3 (FORD), τα αποτελέσματα των ταξινομητών κινούνται σε μέτρια πλαίσια (εύρος από 55% μέχρι 67.7%), με τις περισσότερες να βρίσκονται στο διάστημα από 60% μέχρι 70%. Η καλύτερη εκτίμηση και η καλύτερη μέση εκτίμηση δόθηκε από τον ταξινομητή Decision Tree (67,7% και 65.41% αντίστοιχα), με τις τιμές από τους άλλους δύο ταξινομητές να απέχουν από 0.5% μέχρι 7% (66.8% - 62,32% ο KNN και 67.2% - 60.7% ο Random Forest). Το θετικό είναι ότι και πάλι οι τιμές των τριών ταξινομητών είναι πολύ κοντά μεταξύ τους και κινούνται σε σταθερά διαστήματα, κάτι το οποίο φανερώνει αξιοπιστία του μοντέλου. Οι τιμές επιδέχονται βελτίωσης, καθώς για σίγουρες και πιο αξιόπιστες εκτιμήσεις χρειαζόμαστε τιμές που δεν θα πέφτουν κάτω από το 70% με 75%. Επιπλέον, η οικογένεια

αλγορίθμων Decision Tree είχε τις καλύτερες εκτιμήσεις στις περισσότερες από τις υπόλοιπες μετρικές.

VOLKSWAGEN

	KNN	DT	RF
Train Time	1.33 min	3.16 min	3.86 min
Score Time	0.99 min	2.77 min	3.5 min
Best Score	0.660	0.679	0.680
Mean Accuracy	0.716	0.669	0.707
Prec Score, micro	0.716	0.669	0.707
Prec Score, macro	0.730	0.652	0.703
Prec Score, weighted	0.729	0.654	0.704
Recall Score, micro	0.716	0.669	0.707
Recall Score, macro	0.691	0.645	0.691
Recall Score, weighted	0.716	0.669	0.707
F1 score, micro	0.716	0.669	0.707
F1 score, macro	0.650	0.619	0.677
F1 score, weighted	0.664	0.631	0.686
Fbeta Score, micro	0.716	0.669	0.707
Fbeta Score, macro	0.674	0.648	0.687
Fbeta Score, weighted	0.680	0.636	0.691
Zero One Loss	0.563	0.610	0.57

Πίνακας 4 - Αποτελέσματα εκτιμήσεων classifier που αφορούν τα άρθρα για την αυτοκινητοβιομηχανία Volkswagen

Ο πίνακας 4 (VOLKSWAGEN), έχει τιμές που κινούνται από μέτρια έως σχετικά καλά πλαίσια (εύρος από 56.3% μέχρι 73%), με τις περισσότερες να βρίσκονται στο διάστημα από 60% μέχρι 70%. Η καλύτερη εκτίμηση αυτή τη φορά δόθηκε από τον ταξινομητή Random Forest (68%) και η καλύτερη μέση εκτίμηση δόθηκε από τον ταξινομητή K-nearest neighbors (71.6%). Το θετικό είναι ότι και πάλι οι τιμές των τριών ταξινομητών είναι πολύ κοντά μεταξύ τους και κινούνται σε σταθερά διαστήματα, κάτι το οποίο φανερώνει αξιοπιστία του μοντέλου. Οι τιμές επιδέχονται βελτίωσης, καθώς για σίγουρες και πιο αξιόπιστες εκτιμήσεις χρειαζόμαστε τιμές που δεν θα πέφτουν κάτω από το 70% με 75%. Επιπλέον, παρατηρείται ότι οι ταξινομητές K- Nearest Neighbors και Random Forest μοιράζονται τις καλύτερες μετρικές, ενώ αυτή τη φορά ο Decision Tree δεν έχει καμία καλύτερη τιμή.

8.2 Αποτελέσματα και εκτιμήσεις tweets

Παρακάτω, ακολουθούν τα αποτελέσματα των δεδομένων που προέρχονται από την πλατφόρμα του Twitter, Οι πίνακες που δημιουργήθηκαν, είναι για κάθε εταιρεία ξεχωριστά.

TESLA

	KNN	DT	RF
Train Time	1.93 min	3.81 min	5.2 min
Score Time	1.36 min	3.26 min	4.07 min
Best Score	0.643	0.650	0.660
Mean Accuracy	0.601	0.658	0.627
Prec Score, micro	0.601	0.658	0.627
Prec Score, macro	0.5997	0.673	0.6161
Prec Score, weighted	0.5996	0.674	0.6169
Recall Score, micro	0.601	0.658	0.627
Recall Score, macro	0.608	0.665	0.620
Recall Score, weighted	0.601	0.658	0.627
F1 score, micro	0.601	0.658	0.627
F1 score, macro	0.574	0.632	0.6161
F1 score, weighted	0.570	0.629	0.6169
Fbeta Score, micro	0.601	0.658	0.627
Fbeta Score, macro	0.574	0.645	0.620
Fbeta Score, weighted	0.578	0.644	0.605

Πίνακας 5 - Αποτελέσματα εκτιμήσεων classifier που αφορούν τα tweets για την αυτοκινητοβιομηχανία Tesla.

Παρατηρώντας τον πίνακα 5 (TESLA), προκύπτει ότι τα αποτελέσματα των ταξινομητών κινούνται σε μέτρια κλίμακα (εύρος από 57% μέχρι 67.4%), με τις περισσότερες τιμές να βρίσκονται στο διάστημα από 60% μέχρι 70%. Για άλλη μια φορά, ο ταξινομητής Decision Tree έχει τις περισσότερες καλύτερες τιμες, την καλύτερη εκτίμηση όμως να την έχει ο Random Forest (66%). Η καλύτερη μέση εκτίμηση δόθηκε από τον ταξινομητή Decision Tree (65.8%), με τις τιμές από τους άλλους δύο ταξινομητές να απέχουν αρκετά από τις κορυφαίες τιμές (μέχρι και 7.1% λιγότερο), με αποτέλεσμα οι τρεις ταξινομητές να μην κινούνται στο ίδιο εύρος τιμών, παρά μόνο σε σταθερά διαστήματα, κάτι το οποίο φανερώνει την αξιοπιστία του μοντέλου. Οι τιμές επιδέχονται βελτίωσης, καθώς για σίγουρες και πιο αξιόπιστες εκτιμήσεις χρειαζόμαστε τιμές που δεν θα πέφτουν κάτω από το 70% με 75%.

FORD

	KNN	DT	RF
Train Time	1.42 min	3.67 min	6.07 min
Score Time	1.09 min	3.60 min	4.92 min
Best Score	0.637	0.673	0.670
Mean Accuracy	0.691	0.64	0.678
Prec Score, micro	0.691	0.64	0.678
Prec Score, macro	0.693	0.64	0.679
Prec Score, weighted	0.693	0.64	0.679
Recall Score, micro	0.691	0.64	0.678
Recall Score, macro	0.691	0.64	0.678
Recall Score, weighted	0.691	0.64	0.678
F1 score, micro	0.691	0.64	0.678
F1 score, macro	0.693	0.639	0.675
F1 score, weighted	0.693	0.639	0.675
Fbeta Score, micro	0.691	0.64	0.678
Fbeta Score, macro	0.689	0.639	0.676
Fbeta Score, weighted	0.689	0.639	0.676

Πίνακας 6 - Αποτελέσματα εκτιμήσεων classifier που αφορούν τα tweets για την αυτοκινητοβιομηχανία Ford

Ο πίνακας 6 (FORD), παρουσιάζει μία ενδιαφέρουσα ομοιομορφία στις τιμές του, καθώς σχεδόν όλες οι μετρικές κάθε ταξινομητή έχουν σχεδόν ίδιες τιμές και οι διαφορές μεταξύ των μετρικών ανά ταξινομητή είναι σταθερές και πολύ μικρές. Οι τιμές κινούνται από μέτρια μέχρι σχετικά καλά πλαίσια (από 58.8% μέχρι 69.3%). Στη συγκεκριμένη περίπτωση, ο ταξινομητής K-Nearest Neighbors έχει σχεδόν τις καλύτερες τιμές για όλες τις μετρικές, εκτός από το την καλύτερη εκτίμηση που έγινε από τον Decision Tree (67.3%), ενώ η καλύτερη μέση εκτίμηση είναι 69.1% (KNN). Εκτός από τη μετρική Zero one Loss και την καλύτερη εκτίμηση, όλες οι άλλες μετρικές έχουν σχεδόν τις ίδιες τιμές ανά ταξινομητή (KNN από 68.9% μέχρι 69.3%, DT από 63.9% μέχρι 64% και RF από 67.5% μέχρι 67.9%). Είναι άλλη μία περίπτωση όπου οι τιμές επιδέχονται βελτίωσης, καθώς για σίγουρες και πιο αξιόπιστες εκτιμήσεις χρειαζόμαστε τιμές που δεν θα πέφτουν κάτω από το 70% με 75%.

VOLKSWAGEN

	KNN	DT	RF
Train Time	1.46 min	3.50 min	5.62 min
Score Time	1.08 min	3.33 min	4.51 min
Best Score	0.654	0.661	0.661
Mean Accuracy	0.624	0.660	0.696
Prec Score, micro	0.624	0.660	0.696
Prec Score, macro	0.615	0.632	0.687
Prec Score, weighted	0.625	0.643	0.696
Recall Score, micro	0.624	0.660	0.696
Recall Score, macro	0.6150	0.633	0.689
Recall Score, weighted	0.624	0.660	0.696
F1 score, micro	0.624	0.660	0.696
F1 score, macro	0.6151	0.625	0.687
F1 score, weighted	0.624	0.644	0.696
Fbeta Score, micro	0.624	0.660	0.696
Fbeta Score, macro	0.615	0.628	0.687
Fbeta Score, weighted	0.625	0.642	0.696

Πίνακας 7 - Αποτελέσματα εκτιμήσεων classifier που αφορούν τα tweets για την αυτοκινητοβιομηχανία Volkswagen

Ο πίνακας 7 (VOLKSWAGEN), είναι ένα άλλο παράδειγμα όπου οι τιμές των μετρικών ανά ταξινομητή παραμένουν σταθερές (εκτός από την καλύτερη εκτίμησ) και μεταξύ των ταξινομητών δεν παρουσιάζονται μεγάλες διαφορές στις τιμές των μετρικών. Οι τιμές κινούνται από μέτρια μέχρι σχετικά καλά πλαίσια (από 58.3% μέχρι 69.6%). Ο ταξινομητής Random Forest έχει όλες τις καλύτερες τιμές για τις μετρικές μεταξύ των ταξινομητών (66.1% για καλύτερη εκτίμηση και 69.6% για τη μέση εκτίμηση). Οι τιμές των μετρικών ανά ταξινομητή είναι σχεδόν στα ίδια πλαίσια (61.5% με 62.5% ο KNN, 62.% με 66% ο DT και 68.7% με 69.6% ο RF). Όπως φανέρωσε και ο πίνακας 6, έτσι και αυτός δείχνει την αξιοπιστία του μοντέλου ενώ επίσης οι τιμές επιδέχονται βελτίωσης, καθώς για σίγουρες και πιο αξιόπιστες εκτιμήσεις χρειαζόμαστε τιμές που δεν θα πέφτουν κάτω από το 70% με 75%.

8.3 Αποτελέσματα και εκτιμήσεις συνδυασμού και των δύο μοντέλων

TESLA

	KNN	DT	RF
Train Time	1.63 min	3.60	5.66 min
Score Time	1.17 min	3.17	4.83 min
Best Score	0.646	0.662	0.668
Mean Accuracy	0.609	0.634	0.647
Prec Score, micro	0.609	0.634	0.647
Prec Score, macro	0.611	0.668	0.643
Prec Score, weighted	0.611	0.669	0.651
Recall Score, micro	0.609	0.637	0.647
Recall Score, macro	0.613	0.649	0.650
Recall Score, weighted	0.609	0.637	0.647
F1 score, micro	0.609	0.637	0.647
F1 score, macro	0.599	0.547	0.644
F1 score, weighted	0.597	0.540	0.643
Fbeta Score, micro	0.609	0.637	0.647
Fbeta Score, macro	0.602	0.560	0.650
Fbeta Score, weighted	0.609	0.557	0.647

Πίνακας 8 - Αποτελέσματα εκτιμήσεων classifier που αφορούν το συνδυασμό των μοντέλων για την αυτοκινητοβιομηχανία Tesla

Σύμφωνα με τον πίνακα 8 (TESLA), τα αποτελέσματα των μετρικών δείχνουν πως το συνδυαστικό μοντέλο δεν έχει εφάμιλλα αποτελέσματα με τα προηγούμενα δύο, καθώς οι τιμές είναι αισθητά πιο χαμηλές σε σχέση με τα προηγούμενα δύο. Το μόνο που παραμένει σταθερό είναι το εύρος των τιμών που διατηρείται από 55% μέχρι 67% για όλες τις μετρικές των ταξινομητών, φανερώνοντας άλλο ένα αξιόπιστο και σταθερό μοντέλο. Ο ταξινομητής Random Forest έχει τις περισσότερες καλύτερες τιμές μεταξύ των ταξινομητών, συμπεριλαμβανομένης της καλύτερης και της μέσης εκτίμησης (66.8% και 64.7% αντίστοιχα). Παρόλα αυτά, η αξιοπιστία που χαρακτηρίζει και τα προηγούμενα δύο μοντέλα, υπάρχει και στο τρίτο μοντέλο. Τα αποτελέσματα επιδέχονται βελτίωσης, όπως σε κάθε πίνακα.

FORD

	KNN	DT	RF
Train Time	1.98 min	4.10 min	7.09 min
Score Time	1.507 min	3.57 min	5.54 min
Best Score	0.664	0.680	0.667
Mean Accuracy	0.659	0.605	0.613
Prec Score, micro	0.659	0.605	0.613
Prec Score, macro	0.655	0.613	0.610
Prec Score, weighted	0.656	0.614	0.612
Recall Score, micro	0.659	0.605	0.613
Recall Score, macro	0.655	0.620	0.610
Recall Score, weighted	0.659	0.605	0.613
F1 score, micro	0.659	0.605	0.613
F1 score, macro	0.654	0.575	0.610
F1 score, weighted	0.654	0.568	0.612
Fbeta Score, micro	0.659	0.605	0.613
Fbeta Score, macro	0.654	0.585	0.610
Fbeta Score, weighted	0.656	0.582	0.612

Πίνακας 9 - Αποτελέσματα εκτιμήσεων classifier που αφορούν το συνδυασμό των μοντέλων για την αυτοκινητοβιομηχανία Ford

Ο πίνακας 9 (FORD), είναι ένα άλλο παράδειγμα που φανερώνει ότι το συνδυαστικό μοντέλο δεν εξάγει το ίδιο καλές εκτιμήσεις με τα προηγούμενα δύο. Παραμένει σταθερό το εύρος των τιμών που διατηρείται από 56.8% μέχρι 67.4% για όλες τις μετρικές των ταξινομητών, φανερώνοντας άλλο ένα αξιόπιστο και σταθερό μοντέλο. Ο ταξινομητής K-Nearest Neighbors έχει τις περισσότερες καλύτερες τιμές μεταξύ των ταξινομητών, εκτός της καλύτερης εκτίμησης (68% από τον ταξινομητή Decision Tree), με τη μέση εκτίμηση να είναι 65.9%. Παρόλα αυτά, η αξιοπιστία που χαρακτηρίζει και τα προηγούμενα δύο μοντέλα, υπάρχει και στο τρίτο μοντέλο. Τα αποτελέσματα επιδέχονται βελτίωσης, όπως σε κάθε πίνακα.

VOLKSWAGEN

	KNN	DT	RF
Train Time	1.51 min	3.55 min	6.08 min
Score Time	1.12 min	3.60 min	5.20 min
Best Score	0.665	0.670	0.669
Mean Accuracy	0.685	0.685	0.658
Prec Score, micro	0.685	0.685	0.658
Prec Score, macro	0.684	0.677	0.649
Prec Score, weighted	0.687	0.678	0.652
Recall Score, micro	0.685	0.685	0.658
Recall Score, macro	0.684	0.660	0.649
Recall Score, weighted	0.685	0.685	0.658
F1 score, micro	0.685	0.685	0.658
F1 score, macro	0.684	0.609	0.645
F1 score, weighted	0.686	0.624	0.651
Fbeta Score, micro	0.685	0.685	0.658
Fbeta Score, macro	0.684	0.625	0.647
Fbeta Score, weighted	0.686	0.633	0.651

Πίνακας 10 - Αποτελέσματα εκτιμήσεων classifier που αφορούν το συνδυασμό των μοντέλων για την αυτοκινητοβιομηχανία Volkswagen

Τα αποτελέσματα του πίνακα 10 (VOLKSWAGEN), δείχνουν πως μόνο τα αποτελέσματα του συνδυαστικού μοντέλου για τη Volkswagen πλησιάζουν ή είναι καλύτερα των προηγούμενων δύο μοντέλων. Οι καλύτερες τιμές των μετρικών είναι μοιρασμένες στους ταξινομητές Decision Tree και K-Nearest Neighbors, με την καλύτερη εκτίμηση να προέρχεται από τον ταξινομητή Decision Tree (67%) και την καλύτερη μέση εκτίμηση να προέρχεται και από τον Decision Tree και από τον K-Nearest Neighbors (68.5%). Τα αποτελέσματα αυτού του πίνακα κινούνται στο πιο σταθερό διάστημα (από 59.4% μέχρι 68.7%) ενώ οι αποκλίσεις των τιμών μεταξύ των ταξινομητών είναι αρκετά μικρές στις περισσότερες περιπτώσεις. Πρόκειται για άλλη μία απόδειξη της αξιοπιστίας του μοντέλου, αλλά όπως κάθε προηγούμενο πίνακα και και όπως συμβαίνει σε κάθε μοντέλο, τα αποτελέσματα επιδέχονται βελτίωσης.

Κεφάλαιο 9: Σύγκριση μοντέλων και συμπεράσματα από την εφαρμογή των μεθόδων

9.1 Σύγκριση μοντέλων και ανάλυση αποτελεσμάτων

Έπειτα από την παράθεση των αποτελεσμάτων, ακολουθούν δύο κατηγορίες πινάκων όπου γίνεται σύγκριση των μοντέλων. Η πρώτη παρουσιάζει τα καλύτερα αποτελέσματα για κάθε εταιρεία σε κάθε μοντέλο και ποιος ταξινομητής είχε το καλύτερο αποτέλεσμα ενώ η δεύτερη παρουσιάζει τα καλύτερα σκορ και από τα τρία μοντέλα για κάθε εταιρεία.

9.1.1 Πίνακες καλύτερων εκτιμήσεων ανά μοντέλο

	TESLA	FORD	VW
Best score	0.671 (DT)	0.677 (DT)	0.680 (RF)
Mean Accuracy	0.668 (DT)	0.654 (DT)	0.716 (KNN)
Precision	0.732 (DT)	0.6556 (DT)	0.729 (KNN)
Recall	0.668 (DT)	0.6541 (DT)	0.716 (KNN)
Fi	0.616 (KNN)	0.6444 (DT)	0.680 (RF)
Fbeta	0.620 (KNN)	0.6484 (DT)	0.691 (RF)

Πίνακας 11 - Αποτελέσματα καλύτερων εκτιμήσεων άρθρων

	TESLA	FORD	VW
Best score	0.660 (RF)	0.673 (DT)	0.661 (DT-RF)
Mean Accuracy	0.658 (DT)	0.691 (KNN)	0.696 (RF)
Precision	0.674 (DT)	0.693 (KNN)	0.696 (RF)
Recall	0.658 (DT)	0.691 (KNN)	0.696 (RF)
Fi	0.629 (DT)	0.693 (KNN)	0.696 (RF)
Fbeta	0.644 (DT)	0.689 (KNN)	0.696 (RF)

Πίνακας 12 - Αποτελέσματα καλύτερων εκτιμήσεων tweets

	TESLA	FORD	VW
Best score	0.668 (RF)	0.680 (DT)	0.670 (DT)
Mean Accuracy	0.647 (RF)	0.659 (KNN)	0.685 (DT - KNN)
Precision	0.669 (DT)	0.656 (KNN)	0.687 (KNN)
Recall	0.647 (RF)	0.656 (KNN)	0.685 (DT - KNN)
Fi	0.643 (RF)	0.659 (KNN)	0.686 (KNN)
Fbeta	0.647 (RF)	0.656 (KNN)	0.686 (KNN)

Πίνακας 13 - Αποτελέσματα καλύτερων εκτιμήσεων συνδυασμού μοντέλων

9.1.2 Πινακας καλύτερων συνολικών εκτιμήσεων

	TESLA	FORD	VW
Best score	0.671 (Άρθρα)	0.680 (Συνδυασμός)	0.672 (Άρθρα)
Mean Accuracy	0.668 (Άρθρα)	0.691 (Twitter)	0.716 (Άρθρα)
Precision	0.732 (Άρθρα)	0.694 (Twitter)	0.729 (Άρθρα)
Recall	0.668 (Άρθρα)	0.691 (Twiiter)	0.716 (Άρθρα)
Fi	0.643 (Συνδυασμός)	0.693 (Twitter)	0.696 (Twitter)
Fbeta	0.647 (Συνδυασμός)	0.689 (Twitter)	0.696 (Twitter)

Πίνακας 14 - Αποτελέσματα καλύτερων συνολικών εκτιμήσεων

9.2 Συμπεράσματα

Μετά το πέρας της παράθεσης των αποτελεσμάτων, προκύπτουν κάποια συμπεράσματα προς σχολιασμό, τα οποία είναι ιδιαίτερα ενδιαφέροντα.

1. Είναι φανερό από τον πίνακα του υποκεφαλαίου 9.1.1 (πίνακας 12), πως το μοντέλο που αξιοποίησε καλύτερα τις τεχνικές μηχανικής μάθησης και τους ταξινομητές, σύμφωνα με τις εκτιμήσεις, είναι το μοντέλο των δεδομένων που έχουν ως πηγή άρθρα από το Investing.com. Πιο συγκεκριμένα, είχε τα υψηλότερα αποτελέσματα για την πλειοψηφία των χαρακτηριστικών (4/6 χαρακτηριστικά αμφότερα) που επέλεξα σε δύο αυτοκινητοβιομηχανίες, την Tesla και την Volkswagen. Δεύτερο μοντέλο στην κατάταξη είναι το μοντέλο που έχει ως πηγή δεδομένων το Twitter, έχοντας τα καλύτερα αποτελέσματα εκτιμήσεων στην αυτοκινητοβιομηχανία της Ford (4/6 χαρακτηριστικά), ενώ είχε καλύτερα και δύο χαρακτηριστικά της αυτοκινητοβιομηχανίας της Volkswagen. Τελευταίο κατετάγη το

συνδυαστικό μοντέλο, έχοντας μόνο δύο χαρακτηριστικά που υπερτερούσαν έναντι των άλλων.

2. Στη συγκεκριμένη εργασία, σύμφωνα με τις συνθήκες που επιλέχθηκαν έτσι ώστε να αναπτυχθεί, φαίνεται πως η εκπαίδευση των αλγορίθμων και η τελική αποτίμηση των εκτιμήσεων, αποδίδει καλύτερα όταν υπάρχει μοντέλο με σχετικά μικρό αριθμό χαρακτηριστικών. Θυμίζεται πως το μοντέλο των άρθρων είχε πέντε χαρακτηριστικά, το μοντέλο των tweets είχε έντεκα ενώ το συνδυαστικό μοντέλο είχε δεκαπέντε χαρακτηριστικά.

3. Όσον αφορά τα συνολικά δεδομένα για κάθε εταιρεία και σύμφωνα με τους πίνακες του υποκεφαλαίου 9.1.1 (πίνακας 11, πίνακας 12, πίνακας 13), προκύπτει πως ο ταξινομητής που είχε τη μεγαλύτερη συχνότητα υψηλότερων εκτιμήσεων, είναι η οικογένεια αλγορίθμων Decision Tree, ακολουθούμενη από τον k-nearest neighbors (KNN), με τον Random Forest να βρίσκεται στην τελευταία θέση. Αξίζει να σημειωθεί πως σε μερικές περιπτώσεις, με ακρίβεια τεσσάρων δεκαδικών, υπήρχαν περιπτώσεις όπου δύο ταξινομητές είχαν ακριβώς το ίδιο αποτέλεσμα (DT – KNN και DT – RF).

4. Κάτι το οποίο είναι άξιο αναφοράς, είναι πως το εύρος τιμών το οποίο προέκυψε και από τους τρεις ταξινομητές παρουσιάζει μία σχετική ομοιομορφία. Με άλλα λόγια, και οι τρεις ταξινομητές κινούνται σε ένα ευρός τιμών από 53% μέχρι 75%, για όλες τις εκτιμήσεις, φανερώνοντας πως και οι τρεις ταξινομητές βρίσκονται περίπου στον ίδιο βαθμό αξιοπιστίας, ενώ οι διαφορές που παρουσιάζουν μεταξύ τους είναι σχετικά μικρές.

5. Μπορεί και οι τρεις ταξινομητές να φανερώνουν αξιοπιστία ως προς το εύρος των αποτελεσμάτων τους για ένα τόσο μεγάλο εύρος δεδομένων (πάνω από χίλιες εγγραφές για κάθε set δεδομένων), αυτό όμως δεν σημαίνει πως δεν επιδέχονται βελτίωσης. Τα επιθυμητά ποσοστά ακρίβειας χρειάζεται να κινούνται σε ένα εύρος από 80% μέχρι 85%, ούτως ώστε πέρα από αξιοπιστία, οι αλγόριθμοι να διαθέτουν και την κατάλληλη ακρίβεια.

Κεφάλαιο 10: Μελλοντικές επεκτάσεις

Στην παρούσα εργασία, μελετήθηκαν και αναπτύχθηκαν τεχνικές, βασισμένες σε ήδη υπάρχουσες μεθόδους, προκειμένου να γίνει σύγκριση μεθόδων για την εκτίμηση χρηματιστηριακών μετοχών στην αυτοκινητοβιομηχανία, με τη χρήση αλγορίθμων μηχανικής μάθησης. Τα αποτελέσματα αυτών των τεχνικών παρήγαγαν τρία διαφορετικά προτεινόμενα μοντέλα δεδομένων, τα οποία συγκρίθηκαν μεταξύ τους ως προς την αξιοπιστία τους και τα αποτελέσματά τους.

Χρησιμοποιώντας ως προγραμματιστική γλώσσα την Python, καταφέραμε να επεξεργαστούμε μεγάλο όγκο δεδομένων όπου αφορούσαν απλά κείμενα προερχόμενα από άρθρα και το Twitter. Τα επεξεργασμένα κείμενα αξιολογήθηκαν ως προς το συναισθηματικό τους φόρτο και το σύνολο των μερών του λόγου που περιέχουν και στη συνέχεια αναλύθηκαν με τη βοήθεια τεχνικών μηχανικών μάθησης και ταξινομητών (classifiers). Παρόλα αυτά, επειδή σκοπός της έρευνας είναι συνεχώς να εξελίσσεται και να βελτιώνεται, προτείνονται κάποιες μελλοντικές επεκτάσεις της παρούσας εργασίας, προς διάφορες κατευθύνσεις.

Αρχικά, προτείνεται ο συνδυασμός περισσότερων κοινωνικών δικτύων για τη δημιουργία των δεδομένων, καθώς έχουν μεγάλη βαρύτητα πλέον στο επιχειρησιακό και οικονομικό γίγνεσθαι και μπορούν να φανερώσουν τις κοινωνικές και οικονομικές τάσεις κάθε περιόδου.

Συνέχεια της προηγούμενης πρότασης αποτελεί η δημιουργία ειδικών λεξικών, τα οποία θα εφαρμόζουν στα κοινωνικά δίκτυα και θα προσδίδουν συναισθηματικό φόρτο στις περισσότερες διάσημες πλατφόρμες. Για να δημιουργηθούν τέτοια λεξικά, χρειάζεται να μελετηθούν ποιες από τις λέξεις είναι αυτές που προκαλούν τις αντιδράσεις στους χρήστες.

Τέλος, προτείνεται η ανάπτυξη μιας διαδικτυακής υπηρεσίας, η οποία θα είναι σε θέση να αναλύει αυτόματα τα κείμενα ανάλογα με το είδος τους, άμα είναι π.χ. κείμενα από μέσα κοινωνικής δικτύωσης ή εάν προέρχονται από δημοσιογραφικά πρακτορεία και να τα αναλύει ως προς το συναισθηματικό τους φόρτο.

Κεφάλαιο 11. Βιβλιογραφία

[1] B. Pang και L. Lee, «Opinion Mining and Sentiment Analysis,» *Foundations and Trends® in Information Retrieval,* τόμ. 2, pp. 1-135, 2008.

[2] SAS, «What is Big Data,» SAS, [Ηλεκτρονικό]. Available: https://www.sas.com/el_gr/insights/big-data/what-is-big-data.html.

[3] Oracle, «What is Big Data,» Oracle, [Ηλεκτρονικό]. Available: https://www.oracle.com/big-data/guide/what-is-big-data.html.

[4] J. Alvarado, T. Bandwing και K. Verspoor, «Domain Adaptation of Named Entity Recognition to SupportCredit Risk Assessment,» pp. pp. 84-90, 8-9 December December 2015.

[5] B. P. Mirjana , K. Zivko, S. Sanja και T. Lejla, «Text Mining for Big Data Analysis in Financial Sector: A Literature Review,» *Sustainability,* pp. 1-2, 28 February 2019.

[6] R. Zafarani, A. A. Mohammad και L. Huan, «Social Media Mining,» *Cambridge University Press,* pp. 10-24, 20 April 2014.

[7] Κ. Διαμαντάρας, «Μηχανική Μάθηση - Βασικές έννοιες,» https://aetos.it.teithe.gr/~kdiamant/MachineLearning/MachineLearningLesson01.pdf.

[8] S. Russell και P. Norvig, Artificial Intelligence: A Modern Aprroach, Prentice Hall ISBN 978-0137903955, 2003 (2nd ed.).

[9] Facebook. [Ηλεκτρονικό]. Available: www.facebook.com.

[10] A. Ortigosa, R. M. Carro και J. Martin, «Sentiment analysis in Facebook and its applicationo to e-learning,» *Computer in Human Behavior,* pp. 527-541, February 2014.

[11] J. Dougherty, R. Kohavi και M. Sahami, «Supervised and Unsupervised Discretization on Continuous Features».

[12] O. Omidvar και D. L. Elliott, Neural Systems for Control, Academic Press, ISBN 0-12-526430-5, 1997.

[13] B. Liu, «Sentiment Analysis and Opinion Mining,» 22 April 2012. [Ηλεκτρονικό]. Available: https://www.cs.uic.edu/~liub/FBS/SentimentAnalysis-and-OpinionMining.pdf.

[14] O. Hegazy, O. S. Soliman και M. Abdul Salam, «A Machine Learning Model for Stock Market Prediction,» *nternational Journal of Computer Science and Telecommunications,* pp. 17-18, December 2013.

[15] L. d. S. Pinheiro και M. Dras, «Stock Market Prediction with Deep Learning: A Character-based Neural Language Model for Event-based Trading».

[16] V. K. S. Reddy, «Stock Market Prediction Using Machine Learning,» *International Research Joirnal of Engineering and Technology (IRJET),* τόμ. 05, αρ. 10, p. 1032, October 2018.

[17] Python, «What is Python? Executive Summary,» [Ηλεκτρονικό]. Available: https://www.python.org/doc/essays/blurb/.

[18] Jetbrains, «PyCharm Features,» [Ηλεκτρονικό]. Available: https://www.jetbrains.com/pycharm/features/.

[19] P. Dar, «Comprehensive Beginner's Guide to Jupyter Notebooks for Data Science & Machine Learning,» 24 May 2018. [Ηλεκτρονικό]. Available: https://www.analyticsvidhya.com/blog/2018/05/starters-guide-jupyter-notebook/.

[20] «https://scrapy.org/,» 2019. [Ηλεκτρονικό]. Available: https://scrapy.org/.

[21] «http://nutch.apache.org/,» 2019. [Ηλεκτρονικό]. Available: http://nutch.apache.org/.

[22] P. Jack και N. Levitt, «https://webarchive.jira.com/wiki/display/Heritix,» [Ηλεκτρονικό]. Available: https://webarchive.jira.com/wiki/display/Heritix.

[23] R. Wiki, «Crawler,» [Ηλεκτρονικό]. Available: https://en.ryte.com/wiki/Crawler.

[24] Monkeylearn, «Sentiment Analysis Nearly Everything You Need To Know,» [Ηλεκτρονικό]. Available: https://monkeylearn.com/sentiment-analysis/.

[25] «Επίσημη ιστοσελίδα του Stanford Natural Language Processor Group,» [Ηλεκτρονικό]. Available: http://nlp.stanford.edu/.

[26] Επίσημη ιστοσελίδα του Stanford Natural Language Processor - Log-linear Part-Of-Speech Tagger. [Ηλεκτρονικό]. Available: http://nlp.stanford.edu/software/tagger.shtml.

[27] Energysage, «Electric vehicle manufacturers & companies,» 17 1 2019. [Ηλεκτρονικό]. Available: https://www.energysage.com/electric-vehicles/buyers-guide/top-ev-companies/.

[28] M. Corporation, «Understanding self in Python,» [Ηλεκτρονικό]. Available: https://medium.com/quick-code/understanding-self-in-python-a3704319e5f0. [Πρόσβαση 2019].

[29] W3schools, «JSON - Introductoin,» [Ηλεκτρονικό]. Available: https://www.w3schools.com/js/js_json_intro.asp. [Πρόσβαση 2019].

[30] J. Henrique, «Get Old Tweets Programatically,» [Ηλεκτρονικό]. Available: https://github.com/Jefferson-Henrique/GetOldTweets-python. [Πρόσβαση 2019].

[31] GeeksforGeeks, «Removing stop words with NLTK in Python,» [Ηλεκτρονικό]. Available: https://www.geeksforgeeks.org/removing-stop-words-nltk-python/. [Πρόσβαση 2019].

[32] ExtenfOffice, «Kutools - Combines More Than 300 Advanced Functions And Tools For Microsoft Excel,» [Ηλεκτρονικό]. Available: https://www.extendoffice.com/product/kutools-for-excel.html. [Πρόσβαση 2019].

[33] N. org, «NLTK 3.4.1 documentation,» [Ηλεκτρονικό]. Available: https://www.nltk.org/api/nltk.tokenize.html.

[34] B. Liu, «Επίσημη ιστοσελίδα του Bing Liu,,» [Ηλεκτρονικό]. Available: https://www.cs.uic.edu/~liub/. [Πρόσβαση 2019].

[35] T. Loughran και B. McDonald, «When is a liability not a liability? Textual Analysis,» Dictionaries and 10-Ks. The Journal of Finance, 2011.

[36] [Ηλεκτρονικό]. Available: http://www.wjh.harvard.edu/~inquirer/. [Πρόσβαση 2019].

[37] S. Engine, «English Penn Treebank part-of-speech Tagset,» [Ηλεκτρονικό]. Available: https://www.sketchengine.eu/penn-treebank-tagset/. [Πρόσβαση 2019].

[38] S. B. Kotsiantis, «Supervised Machine Learning: A Review of Classification,» 2007.

[39] P. Strecht, L. Cruz και C. Soares, «A Comparative Study of Classification and Regression Algorithms for Modelling Students,» 2015. [Ηλεκτρονικό]. [Πρόσβαση 2019].

[40] [Ηλεκτρονικό]. Available: http://mathematica.stackexchange.com/questions/23998/how-can-i-consistently-get-a-good-logistic-regression-fit. [Πρόσβαση 2019].

[41] H. Tou Ng και H. L. Chieu, «Named Entity Recognition with a Maximum Entropy,» σε *Proceedings of the Seventh Conference on Natural Language Learning* , 2003.

[42] I. Πανάρετος, «K- Nearest Neighbors,» 2004. [Ηλεκτρονικό]. Available: http://www2.stat-athens.aueb.gr/~jpan/short-course-ergasia-Mpakra.pdf. [Πρόσβαση 2019].

[43] D. T. Larose, «Decision Trees,» σε *Discovering Knowledge in Data: An Introduction to Data Mining*, John Wiley & Sons,, pp. 90-106.

[44] J. R. Quinian, «Induction of Decision Trees. Mach. Learn.».

[45] L. Breiman, «Random Forests,» σε *Machine Learning*, 2005, pp. 5-32.

[46] A. Culter , D. R. Cutler και J. R. Stevens, «Random Forests,» σε *Ensemble Machine Learning: Methods and Applications*, 2012, pp. 157-175.

[47] skikit-learn, «sklearn.model_selection.GridSearchCV,» 2019. [Ηλεκτρονικό]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html.

[48] A. S. A., «An exact analytical relation among recall, precision, and classification accuracy in information retrieval,» 2002.

[49] 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.

[50] A. S. A., «An exact analytical relatoin among recall, precision, and classificatoin accuracy in information retrieval,» 2002.

[51] A. Yeh, «More accurate tests for the statistical significance of result differences,» σε *COLING '00 Proceedings of the 18th conference on Computational linguistics*, 2000.

[52] 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.

[53] W3schools, «Python Dictionaries,» W3schools, [Ηλεκτρονικό]. Available: https://www.w3schools.com/python/python_dictionaries.asp. [Πρόσβαση 2019].

[54] Microsoft, «Introduction to Microsoft Power Query for Excel,» [Ηλεκτρονικό]. Available: https://support.office.com/en-us/article/introduction-to-microsoft-power-query-for-excel-6e92e2f4-2079-4e1f-bad5-89f6269cd605. [Πρόσβαση 2019].

[55] Kallipos, «ΚΕΦΑΛΑΙΟ 4 –Μηχανική Μάθηση,» [Ηλεκτρονικό]. Available: https://repository.kallipos.gr/bitstream/11419/3382/1/02_chapter_04.pdf.

[56] A. K. Nassirtoussi, S. Aghabozorgi, T. Ying Wah και D. Chek Ling Ngo, «Text mining of news-headlines for FOREX market prediction: A Multi-layer,» σε *Expert Systems with Applications 42*, 2015, pp. 306-324.

[57] L. Breiman, «Random Forests,» *Machine Learning,* τόμ. 45, αρ. 1, pp. 5-32, 2005.

[58] 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.

[59] D. T. Larose, «Decision Trees,» σε *Discovering Knowledge in Data: An Introduction to Data Mining*, John Wiley & Sons, Inc., 2005, pp. 107-127.

[60] 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.

[61] P. Domingos, «A Unified Bias-Variance Decomposition for Zero-One and Squared Loss,» σε *Seventeenth National Conference on Artificial Intelligence*, Austin Texas, 2000.

Κεφάλαιο 12. Λίστα Πινάκων

Κεφάλαιο 13. Λίστα εικόνων

Κεφάλαιο 14. Λίστα ακρωνύμιων

API - Application Programming Interface	Διεπαφή Προγραμματισμού Εφαρμογών
IDE - Integrated Development Envirnment	Ολοκληρωμένο Περιβάλλον Ανάπτυξης
NLP - Natural Language Processing	Φυσική Επεξεργασία Γλώσσας
Data Set	Συλλογή Δεδομένων
Training Set	Συλλογή Δεδομένων Εκπαίδευσης
POS – Part of Speech	Μέρος του λόγου