Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών

Τμήμα Πληροφορικής και Τηλεπικοινωνιών

Χειμερινό Εξάμηνο 2018-2019

Τεχνικές Ανάλυσης Δεδομένων Μεγάλης Κλίμακας(M118)

Σπυλόπουλος Σπυρίδων CS1180007

Τριανταφύλλου Κωνσταντίνος CS2180018

Αθήνα 2018

Πίνακας Περιεχομένων

1. Σκοπός της Εργασίας

Σκοπός της εργασίας είναι η εξοικείωσή μετα βασικά στάδια της διαδικασίας που ακολουθούνται για την εφαρμογή τεχνικών εξόρυξης δεδομένων, ήτοι: συλλογή, προ-επεξεργασία/καθαρισμός, μετατροπή, εφαρμογή τεχνικών εξόρυξης δεδομένων και αξιολόγηση. Η υλοποίηση έγινε στην γλώσσα προγραμματισμού Python με την χρήση του εργαλείου SciKit Learn και της βιβλιοθήκης gensim.

Πιο συγκεκριμένα η εργασία σχετίζεται με την κατηγορποίηση δεδομένων κειμένου από ειδησεογραφικά άρθρα. Τα Dataset είναι αρχεία CSV του οποίου τα πεδία είναι διαχωρισμένα με τον χαρακτήρα '\t'(TAB). Περέχονται δύο αρχεία:

1. train_set.csv: Το αρχείο αυτό χρησιμοποιήθηκε για την εκπαίδευση των αλγορίθμων και περέχει τα πεδία: Id, Title, Content, Category.

2. test_set.csv: Το αρχείο αυτό χρησιμοποιήθηκε για την πρόβλεψη της κατηγορίας κάθε άρθρου, που περιέχεται στο συγκεκριμένο αρχείο και περιέχει τα ίδια πεδία εκτός από το Category το οποίο κληθήκαμε να προβλέψουμε με τους αλγορίθμους κατηγοροποίησης που περιγράφονται σε επόμενο κεφάλαιο.

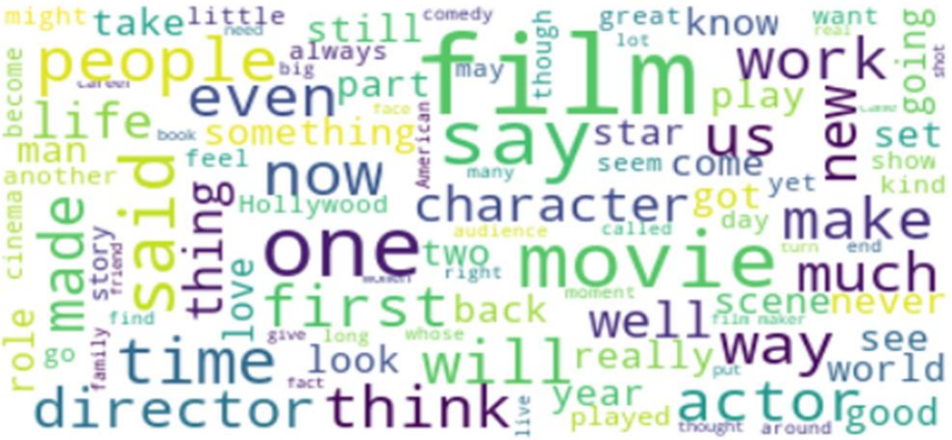
Οι κατηγορίες των άρθρων είναι πέντε: Business, Film, Football, Politics, Technology

2. Δημιουργία WordCloud

Στο σημείο αυτό έπρεπε να να δημιουργήσουμε ένα Wordcloud για κάθε κατηγορία, χρησιμοποιώντας όλα τα άρθρα της συγκεκριμενης κατηγορίας. Για την υλοποίηση του παραπάνω σεναρίου χρησιμοποιήθηκε η βιβλιοθήκη wordcloud που βρίσκεται στο παρακάτω λινκ: `https://github.com/amueller/word_cloud`.

Συγκεκριμένα διαβάστηκε το αρχείο train_set.csv με την βοήθεια της pandas. Δεν χρειάστηκε προ-επεξεργασία των δεδομένων καθώς δεν περιέχονται κενά πεδία. Στην συνέχεια με την βοήθεια του DataFrame της python βρέθηκαν οι μοναδικές κατηγορίες που περιέχονται και για κάθε μία δημιουργήθηκε ένα κείμενο με όλα τα Contents των άρθρων της συγκεκριμένης κατηγορίας. Δίνοντας σαν όρισμα το συγκεκριμένο κείμενο στο WordCloud δημιουργήθηκε το αντίστοιχο Wordcloud το οποίο με την βοήθεια της matplotlib.pyplot δημιουργήθηκε η αντίστοιχη εικόνα και αποθηκεύτηκε ώς αρχείο .png με όνομα το όνομα κάθε κατηγορίας. Στην συνέχεια παραθέτονται τα Wordclouds όλων των κατηγοριών τα οποία παράχθηκαν τρέχοντας το αρχείο wordCloud.py.

Εικόνα 1: Business.png



Εικόνα 2: Film.png

Εικόνα 3: Football.png

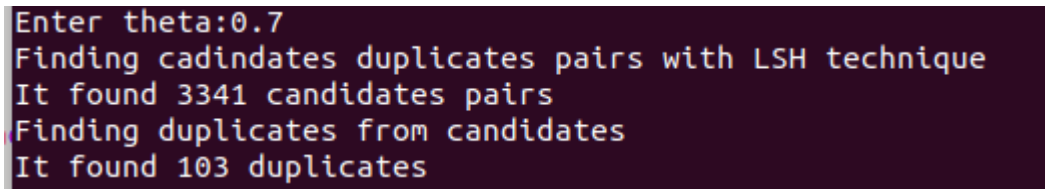
Εικόνα 4: Politics.png

Εικόνα 5: Technology.png

3. Εντοπισμός Duplicates

Στο συγκεκριμένο σημείο έπρεπε να εντοπιστούν παρόμοια κείμενα. Πιο συγκεκριμένα η ομοιότητα ανάμεσα σε δύο κείμενα μετρήθηκε με το cosine similarity μεταξύ των term vectors κάθε κειμένου. Το term vector κάθε κειμένου βρέθηκε με την μετατροπή του κειμένου σε vector με την βοήθεια του TfidfVectorizer ο οποίος αρχικά εκπαιδεύτηκε με όλα τα κείμενα. Για τον γρήγορο εντοπισμό των duplicates, ο οποίος προφανώς δεν είναι ο έλεγχος όλων των κειμένων μεταξύ τους, χρησιμοποιήθηκε η τεχνική LSH και συγκεκριμένα η βιβλιοθήκη LSH από το παρακάτω λινκ: https://github.com/mattilyra/lsh.

Η συγκεκριμένη τεχνική χρησιμοποιήθηκε με σκοπό των έλεγχο των ζευγαριών τα οποία είναι πιθανό να είναι παρόμοια και όχι όλων μεταξύ τους. Για τον σκοπό αυτό κάθε κείμενο έχινε hash με τον MinHasher της βιβλιοθήκης δημιουργώντας ένα “μοναδικό αποτύπωμα” και αποθηκεύεται στην μνήμη της lsh στο αντίστοιχο bin και στη συνέχεια στο αντίστοιχο bucket ανάλογα με το hashing του. Στη συνέχεια για όλα τα άρθρα του κάθε bucket επιστρέφονται όλοι οι συνδυασμοί μεταξύ τους σαν υποψήφια duplicates. Με την τεχνική αυτή μειώνεται κατα πολύ ο αριθμός των ελέγχων που πρέπει να γίνουν σε σχέση με την σειριακή αναζήτηση. Όσον αφορά τα κύρια ορίσματα που είναι απαραίτητα για την εκτέλεση της τεχνικής είναι δυό: το μήκος του αποτυπώματος(όσο μεγαλυτερος τόσο καλύτερη η κατανομή στα buckets με κόστος όμως αρκετή μνήμη) και ο αριθμός των ζώνων που στην μνήμη της lsh(όσο μεγαλύτερος τόσο πιό πολλα υποφήφια duplicates θα έχουμε). Πειραματιζόμενοι με τις δύο αυτές παραμέτρους παρατηρήσαμε ότι για seeds=100 και bands=20 έχουμε περίπου 3300 υποψήφια duplicates και 100 duplicates με θ=0.7.



Τέλος ο αλγόριθμος ο οποίος βρίσκεται στο αρχείο duplicates.py ζητάει ένα threshold θ το οποίο αν είναι 0.7 επιστρέφει σε αρχείο csv τα ζευγάρια με το cosine similarity που είναι πάνω από 0.7, ειδάλλως απλά εκτυπώνει το Dataframe.

4. Υλοποίηση Κατηγοριοποίησης (Classification)

Στο συγκεκριμένο σημείο έγινε μέτρηση της απόδοσης των αλγορίθμων που αναφέρονται στην εκφώνηση χρησιμοποιώντας τα features που προτείνονται και στη συνέχεια μιας δικιά μας μέθόδου η οποία είχε αρκετά καλύτερα αποτελέσματα σε σχέση με τις υπόλοιπες. Για την μέτρηση χρησιμοποιήθηκε η τεχνική 10-fold Cross Validation και μετρήθηκαν οι παρακάτω μετρικές:

- Accuracy
- Precision
- Recall
- F-Measure

Η παραπάνω τεχνική χωρίζει σε 10 κομμάτια τον πίνακα των δεδομένων και χρησιμοποιεί τα εννέα από αυτά για να εκπαιδεύσει τους classifiers και το ένα για να προβλέψει την κατηγορία του κάθε άρθρου σε αυτό το κομμάτι. Οι μετρικές υπολογίζονται από την συγκριση των αποτελεσμάτων με των πραγματικών τιμών και στη συνέχεια από τον μέσο όρο τους.

Οι αλγόριθμοι που ζητούνται είναι οι Support Vector Machines(SVM) και Random Forest.

Ο SVM παρέχει αρκετούς classifiers και στα πλαίσια της εργασίας χρησιμοποιήθκε ο SVC, ο οποίος έχει πολυπλοκότητα περίπου τετραγωνική. Ο Random Forest χτίζει σταδιακά δέντρα απόφασης κάνοντας μια σειρά από ερωτήσεις στα χαρακτηριστικά. Η τεχνική αυτή είναι αρκετά ακριβή γιατί εξαρτάται από το πλήθος των δεδομένων, τον αριθμό τον χαρακτηριστικών και το σχήμα του δέντρου, έχοντας λογαριθμική πολυπλοκότητα.

Για την μέτρηση της απόδοσης των αλγορίθμων χρησιμοποιήθηκαν τα παρακάτων features:

- Bag of Words
- SVD διατηρώντας το 90% του variance.
- Average word vector για κάθε κείμενο.

Με το Bag of Words ουσιαστικά γίνεται μέτρηση της κάθε λέξης που υπάρχει στο document συγκριτικά με το λεξικό μας που στην συγκεκριμένη περίπτωση είναι οι λέξεις που υπάρχουν σε όλα τα άρθρα. Με την βοήθεια της τεχνικής LDA παράγεται ένας vector με όλα τα documents σε μορφή hashing.

Με το SVD γίνεται μείωση της διάστασης ενός vector που έχει παραχθεί από κάποιον vectorizer. Στη συγκεκριμένη περίπτωση καθώς θέλαμε να έχουμε 90% variance ήταν απαραίτητο να έχουμε τουλάχιστον 600 διαστάσεις στο παραγώμενο vector, ο οποίος είχε προέλθει από την χρήση του CountVectorizer ο οποίος προτιμήθηκε σε σχέση με τον TfidfVectorizer, γιατί ο δεύτερος ήθελε αρκετά μεγαλύτερο αριθμό διαστάσεων σε σχεση με τον πρώτο για να διατηρηθεί το 90% variance.

Τέλος ο W2D είναι αρκετά χρήσιμο σε περιπτώσεις που η πρόβλεψη γίνεται συγκρίνοντας λέξεις μεταξύ τους και όχι documents, όπως στην δικιά μας περίπτωση. Για την υλοποίηση του

συγκεκριμένου feature υλοποιήθηκε αλγόριθμός ο οποίος επιστρέφει τον μέσο όρο των word vectors που παράγει ο η Word2Vec για την χρησιμοποίηση του στον κάθε classifier.

Παρατηρώντας τα αποτελέσματα των μετρήσεων των παραπάνω περιπτώσεων πειραματιστήκαμε και καταλήξαμε στην χρησιμοποίηση του SVD χωρίς όμως να μας ενδιαφέρει το variance και σε συνδυασμό με τον TfidfVectorizer με μόλις 50 διαστάσεις καταφέραμε να έχουμε αρκετά καλύτερες μετρήσεις σε σχέση με τα υπόλοιπα. Από τους δύο αλγορίθμους προτιμήθηκε ο SVM.SVC καθώς είχε αρκετά καλύτερα αποτελέσματα σε σχέση με τον Random Forest.

Οι μετρήσεις φαίνονται στον παρακάτω πίνακα:

Statistic Measure	SVM (BoW)	Random Forest (BoW)	SVM (SVD)	Random Forest (SVD)	SVM (W2V)	Random Forest (W2V)	Our Method
Accuracy	0.85	0.79	0.94	0.55	0.40	0.36	0.96
Precision	0.86	0.72	0.94	0.59	0.42	0.33	0.96
Recall	0.85	0.79	0.94	0.55	0.40	0.36	0.96
F-Measure	0.85	0.76	0.94	0.57	0.41	0.35	0.96

Τέλος με την χρήση της μεθόδου μας έγινε εκτίμηση της κατηγορίας του κάθε άρθρου που βρίσκεται στο αρχείο test.csv και τα αποτελέσματα βρίσκονται στο testSet_categories.csv.