ΕΞΟΡΥΞΗ ΓΝΩΣΗΣ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΕΑΡΙΝΟ ΕΞΑΜΗΝΟ 2022

Kaggle challenge: Citation Prediction

Ονοματεπώνυμο: Βασίλειος Ηλίας Δρούζας
Επωνυμία Ομάδας (Kaggle): DataFreak
Αριθμός Μητρώου: 3180051
Διδάσκων: Κωνσταντίνος Σκιάνης

Μάιος 2022

Εργαλεία: Jupyter Notebook,Python,Latex

Περιεχόμενα

1 Εισαγωγή και περιγραφή της μελέτης περίπτωσης: Citation prediction

Το link prediction είναι το πρόβλημα της πρόβλεψης της ύπαρξης μιας σύνδεσης μεταξύ δύο οντοτήτων σε ένα δίκτυο. Στο πρόβλημά μας, θα ασχοληθούμε με το πρόβλημα της πρόβλεψης ενός paper να αναφέρει κάποιο άλλο.

Μας δίνεται ένα δίκτυο αναφορών που αποτελείται από πολλές χιλιάδες ερευνητικές εργασίες, μαζί με τις περιλήψεις τους και τη λίστα των συγγραφέων τους. Ο αγωγός που ακολουθείται συνήθως για την αντιμετώπιση του προβλήματος είναι παρόμοιος με αυτόν που εφαρμόζεται σε οποιοδήποτε πρόβλημα ταξινόμησης. Ο στόχος είναι να χρησιμοποιήσουμε πληροφορίες ακμών για να μάθουμε τις παραμέτρους ενός ταξινομητή και στη συνέχεια να χρησιμοποιήσουμε τον ταξινομητή για να προβλέψουμε εάν δύο κόμβοι συνδέονται με μια ακμή ή όχι.

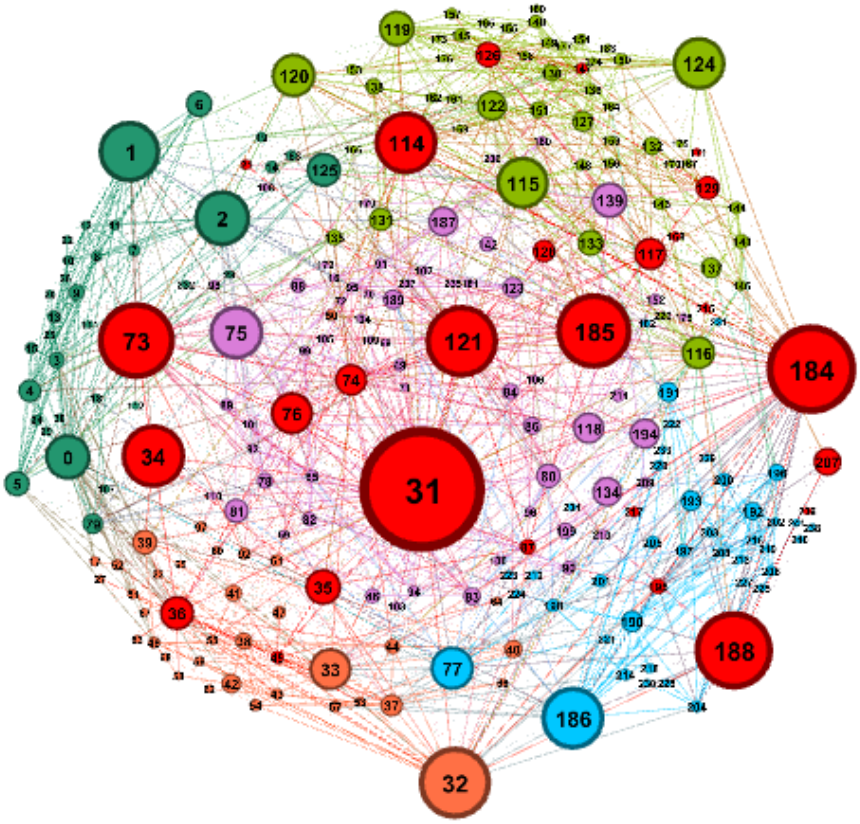

Figure 1: Υπόδειγμα ενός γράφου κοινωνικού δικτύου

2 Ανάλυση γράφου

Εφόσον διαγραμματικά είναι δύσκολο να σχεδιάσουμε τον κανονικό γράφο (παίρνει απαγορευτικό χρόνο) θα κάνουμε subgraphs, δοκιμάζοντας με τους πρώτους 10000 και τους τελευταίους 10000 κόμβους.

Ο πρώτος γράφος έχει τα 10000 τελευταία nodes και 2168 edges και είναι μη κατευθυνόμενος. Αν τον σχεδιάσουμε παίρνουμε το ακόλουθο αποτέλεσμα:

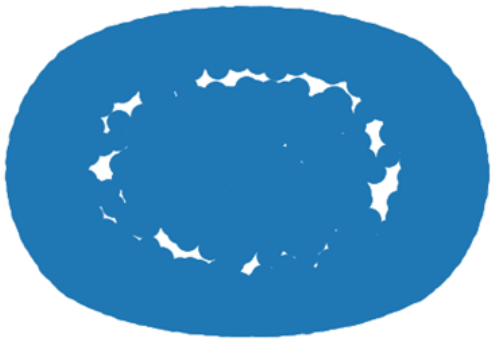

Figure 2: Γράφος με τα τελευταία 10000 nodes.

Στον αρχικό γράφο:

2.1 Density

Μπορούμε να βρούμε το density. Η πυκνότητα ενός γραφήματος είναι ένα μέτρο του πόσους δεσμούς μεταξύ κόμβων υπάρχουν σε σύγκριση με το πόσοι δεσμοί είναι δυνατο να υπάρχουν μεταξύ κόμβων. Με άλλα λόγια, είναι ένα μέτρο του πόσο κοντά είναι το γράφημα σε ένα πλήρες γράφημα με τον ίδιο αριθμό κόμβων. Βρίσκουμε ότι το density του G είναι 0.0011246324632463247, άρα μιλάμε για πολύ αραιό δίκτυο.

2.2 Μονοπάτια - διάμετρος

Μπορούμε να βρούμε τα κοντινότερα μονοπάτια για κάθε κόμβο που θέλουμε, π.χ. από τον κόμβο 154 στον 6551 είναι το μονοπάτι [154, 66, 245, 4860, 6551]

Υπάρχουν και άλλα metrics που προκύπτουν από τα Shortest paths, π.χ. η διάμετρος. Στο γράφο μας όμως, ο οποίος είναι μη κατευθυνόμενος, είναι αδύνατον να υπολογιστεί η διάμετρος, εφόσον έχουμε παραπάνω από ένα component. (Υπάρχουν κάποιοι κόμβοι που δεν έχουν μονοπάτι για όλους τους άλλους)

2.3 Triadic closure

Έπειτα μπορούμε να ελέγξουμε για το triadic closure. Το triadic closure μας λέει ότι αν δύο άτομα γνωρίζουν το ίδιο άτομο, πιθανόν να γνωρίζονται και μεταξύ τους. Ένας τρόπος να το υπολογίσουμε λοιπόν είναι μέσω του transitivity(ratio των τριγώνων εναντίον όλων των πιθανών τριγώνων). Βρίσκουμε τιμή 0.0269696828739865 4 για το transitivity.(λίγο παραπάνω από την τιμή που πήραμε για το density). Επειδή ο γράφος δεν είναι πυκνός,υπάρχουν και λιγότερα πιθανά τρίγωνα . Κόμβοι που ήδη έχουν πολλές συνδέσεις πιθανόν να αποτελούν κομμάτι των τριγώνων.

2.4 Centrality

Τα centrality measures αποτελούν measures για την σημασία τωνκόμβων, π.χ. αν θέλουμε να βρούμε τους πιο κρίσιμους κόμβους στο δίκτυο. Χρησιμοποιούμε την έννοια του degree (βαθμού) ενός κόμβου. Το degree ενός κόμβου είναι πολύ απλά το άθροισμα των ακμών του. Μπορούμε να βρούμε το degree όποιου κόμβου θέλουμε, π.χ. για τον κόμβο 2112 το degree είναι 21. Βρίσκουμε και τους πρώτους 20 κόμβους με τα μεγαλύτερα degrees:

```
Top 20 nodes by degree:
(369, 2994)
(4859, 1295)
(4860, 1154)
(8492, 767)
(798, 662)
(2442, 553)
(8921, 492)
(4169, 416)
(1292, 362)
(407, 347)
(7421, 319)
(5413, 306)
(1882, 301)
(796, 290)
(4284, 260)
(1796, 252)
(9517, 248)
(66, 239)
(4344, 232)
(8912, 209)
```

Figure 3: Οι πρώτοι 20 κόμβοι με τα μεγαλύτερα degrees.

Υπάρχουν και άλλα metrics για τον υπολογισμό του centrality, π.χ. το eigenvector centrality, betweenness centrality κλπ. Η closeness centrality προσπαθεί να βρει κόμβους που διαδίδουν πληροφορίες στο γράφημα. Οι κόμβοι με υψηλή βαθμολογία εγγύτητας έχουν τη μικρότερη απόσταση από όλους τους άλλους κόμβους. Η κεντρική θέση μεταξύ είναι ένας τρόπος ανίχνευσης της ποσότητας επιρροής που έχει ένας κόμβος πάνω από τη ροή των πληροφοριών στο γράφημα. Μπορούμε να το χρησιμοποιήσουμε για να βρούμε κόμβους που χρησιμεύουν ως γέφυρα από το ένα μέρος του γραφήματος στο άλλο. Δουλεύοντας αντίστοιχα μπορούμε να βρούμε τους πρώτους 20 κόμβους με βάση το betweeness centrality.

2.5 Communities

Ένα άλλο ερώτημα είναι ποια είναι τα μικρότερα groups και communities μέσα σε ένα μεγάλο δίκτυο. Κλίκες, clusters και communities μπορούν να υπολογιστούν με το modularity. Το modularity είναι ένα measure της πυκνότητας του δικτύου, π.χ. μια community έχει μεγάλη πυκνότητα σε σχέση με τους κόμβους που βρίσκονται στην 'περιοχή' της, αλλά μικρή σε σχέση με εξωτερικούς. Μιας και το δίκτυο μας είναι αραιό, θα είναι ίσως πιο έυκολο να χωρίσουμε σε communities. Με τη χρήση κατάλληλων μεθόδων βρίσκουμε τον κατάλληλο αριθμό communities για τον γράφο.Μπορούμε να βρούμε τις τάξεις που προκύπτουν από το modularity και τα μέλη τους, με ένα loop στη λίστα των communities.

Το συγκεκριμένο ερώτημα, επειδή τρέχει σε απαγορευτικό χρόνο στο κανονικό γράφο, το έτρεξα στο subgraph με τα πρώτα 10000 nodes.Ο αλγόριθμος βρίσκει 69 communities. Η ομαδοποίηση που κάνει είναι η εξής:

Class 0	3360 nodes
Class 1	2835 nodes
Class 2	1550 nodes
Class 3	1112 nodes
Class 4	675 nodes
Class 5	95 nodes
Class 6	93 nodes
Class 7	48 nodes
Class 8	44 nodes
Class 9	41 nodes
Class 10	40 nodes
Class 11	20 nodes
Classes 12-13	9 nodes
Class 14	7 nodes
Class 15	3 nodes
Classes 16-21	2 nodes
Classes 22-68	1 node

Figure 4: Χωρισμός του υπογράφου σε communities.

2.6 Bridges - Cliques

Μια γέφυρα σε ένα γράφημα είναι μια ακμή της οποίας η αφαίρεση προκαλεί την αύξηση του αριθμού των συνδεδεμένων στοιχείων του γραφήματος. Ισοδύναμα, μια γέφυρα είναι μια άκρη που δεν ανήκει σε κανέναν κύκλο. Ελέγχουμε τον υπογράφο και για γέφυρες. Ένα μέρος των αποτελεσμάτων αποτελούν οι εξής:

```
(1, 3)
(1, 7)
(1, 15)
(1, 17)
(1, 19)
(2, 27)
(2, 29)
(2, 46)
(2, 56)
(7, 8)
(43, 412)
(43, 428)
(4, 5202)
(798, 4014)
(798, 4137)
```

Figure 5: Εύρεση γεφυρών στον υπογράφο.

Ανάλογα μπορούμε να ελέγξουμε και για την ύπαρξη τοπικών γεφυρών (local bridges). Όσο αφορά τις κλίκες του γράφου, βρίσκουμε ότι τόσο ο αρχικός γράφος όσο και ο υπογράφος που μελετάμε αποτελούνται από 37302 κλίκες.

Έπειτα βρίσκουμε το assortavity coefficient,που χρησιμοποιείται για τον προσδιορισμό του βαθμού στον οποίο ορισμένοι συνδεδεμένοι κόμβοι έχουν παρόμοια χαρακτηριστικά. Για τον αρχικό γράφο και τον υπογράφο το assortavity coefficient είναι -0.0812, άρα δεν υπάρχει ισχυρή συσχέτιση των values μεταξύ των συνδεδεμένων κόμβων. Το αποτέλεσμα είναι κοντά στο 0, κάτι που υποδεικνύει και το μεγάλο μέγεθος του γράφου.

Αν θέλουμε να ελέγξουμε και τον υπογράφο με τα πρώτα 10000 nodes, παίρνουμε διαγραμματικά το εξής αποτέλεσμα:

Figure 6: Γράφος με τα πρώτα 10000 nodes.

Παρατηρούμε ότι διαφέρει σημαντικά σε σχέση με τον πρώτο γράφο.

3 Ανάλυση κειμένου

3.1 Word2Vec-Gensim

Θα χρησιμοποιήσουμε τον αλγόριθμο Word2Vec , ο οποίος φτιάχνει word embeddings τέτοια ώστε τα embeddings με παρόμοιο meaning να δείχνουν προς την ίδια κατεύθυνση. Επομένως μπορεί να βρει τα συνώνυμα για κάθε λέξη.

Φτιάχνουμε ένα Word2vec μοντέλο με gensim και μετά μπορούμε να βρούμε:

- Την ομοιότητα (similarity) μεταξύ δύο λέξεων που βρίσκονται στο κείμενο (χρησιμοποιεί το συνημίτονο) στη κλίμακα 0-1.
- Δοθέντων κάποιων λέξεων, να βρει ποια ή ποιες δε ταιριάζουν με τις υπόλοιπες που δόθηκαν.
- Να βρει μεταξύ κάποιων λέξεων ποια λέξη ταιριάζει περισσότερο με μια άλλη (πάλι δείκτης 0-1).

Στο Figure 7 βλέπουμε τα αποτελέσματα: 1) Του ελέγχου cosine similarity για τις λέξεις 'development' και 'placed', 2) ανάμεσα στις λέξεις 'development', "recommendations' και 'placed' ποια είναι αυτή που ταιριάζει λιγότερο με τις υπόλοιπες και 3) τις τρεις λέξεις που ταιριάζουν περισσότερο με τις "development", "recommendations", "placed" και ταυτόχρονα που ταιριάζουν όσο το δυνατόν λιγότερο με τη λέξη "inside".

```
Cosine similarity between 'development' and 'placed' - CBOW :  0.06146417
recommendations
[('each', 0.1920749545097351), ('that', 0.16941361129283905), ('data', 0.1678740680217743)]
```

Figure 7: Τα outputs για τις παραπάνω εντολές.

3.2 Sentiment Intensity Analyzer

Στη συνέχεια μπορούμε να βρούμε αν το κείμενο είναι ουδέτερο ή θετικό/αρνητικό με τη χρήση του SentimentIntensityAnalyzer. Τρέχοντας για όλο το αρχείο abstracts θα ήταν η ιδανική επιλογή, όμως παίρνει απαγορευτικό χρόνο λόγω του μεγάλου μεγέθους. Όμως επειδή όλα τα nodes γενικά ακολουθούν παρόμοιο στυλ , το οποίο θεωρούμε εκ των προτέρων ότι είναι ουδέτερο, μπορούμε να παρουσιάσουμε τα αποτελέσματα του πρώτου κόμβου:

'neg': 0.0, 'neu': 0.935, 'pos': 0.065, 'compound': 0.8481

Κατά 93.5% το κείμενο βρέθηκε ουδέτερο και κατά 6.5% θετικό, κάτι που επιβεβαιώνει τις αρχικές προβλέψεις μας. Η κλίμακα αυτή χοντρικά μπορεί να γενικευτεί για όλους τους κόμβους, άρα το κείμενό μας είναι ουδέτερο.

3.3 FreqDist

Η συνάρτηση FreqDist δίνει στο χρήστη την κατανομή συχνότητας όλων των λέξεων στο κείμενο. Τρέχοντας για όλο το αρχείο abstracts οι 5 πιο συχνές περιλαμβάνονται 68.654,107.423,230,44.318 και 55.571 φορές (το βλέπουμε με την most_common).

Αντίστοιχη ανάλυση μπορεί να γίνει και στο αρχείο των συγγραφέων, για να βρούμε τους συγγραφείς που αναγράφονται τις περισσότερες φορές. Στα αποτελέσματα των 5 πιο συχνών περιλαμβάνονται οι:

```
[(20300, 'Žiga Emeršič,Vitomir Štruc,Peter Peer\n'),
 (97681,
  'Želmira Tóthová,Jaroslav Polec,Tatiana Orgoniková,Lenka Krulikovská\n'),
 (110528, 'Željko Agić,Barbara Plank,Anders Søgaard\n'),
 (114532, 'Željko Agić,Nikola Ljubešić,Danijela Merkler\n'),
 (51933, 'Željko Agić,Dirk Hovy,Anders Sogaard\n')]
```

Figure 8: Οι 5 πιο συχνές ομάδες συγγραφέων.

4 Στάδια

4.1 Τι είδους αναπαράσταση δεδομένων χρησιμοποίησα και ποια χαρακτηριστικά;

Εδώ ακολουθήθηκε η μέθοδος του baseline, δηλαδή μαζεύουμε τα abstracts σε ένα λεξικό και μετά το αντιστοιχίζουμε σε tuple(αντί set) για καλύτερη απόδοση. Τα χαρακτηρηστικά που χρησιμοποιήθηκαν ήταν τα τρία του baseline, δηλ. 1. Το άθροισμα των μοναδικών όρων των abstracts των δύο κόμβων , 2. Η απόλυτη τιμή της διαφοράς τους, 3. Ο αριθμός των κοινών όρων μεταξύ των κόμβων των abstracts . Επίσης πρόσθεσα άλλο ένα, το πηλίκο μεταξύ των abstracts των δύο κόμβων. Απέφυγα να προσθέσω περισσότερα, διότι ναι μεν όσο αυξάνονται τα features αυξάνεται και η ακρίβεια της μεθόδου πρόβλεψης αλλά παρατηρείται το φαινόμενο της υπερπροσαρμογής (overfitting), όπου τα δεδομένα αποδίδουν εξαιρετικά στα δεδομένα εκπαίδευσης και όχι καλά στο δεδομένα ελέγχου.

4.2 Εφάρμοσα τεχνικές μείωσης διαστάσεων;

Θεωρητικά, ο πιο σωστός τρόπος είναι πρώτα να κάνουμε την PCA και μετά να χωρίσουμε τα δεδομένα. Προσπάθησα λοιπόν αρχικά να κάνω scale τα δεδομένα (ΌΛΑ, δηλ. τη μεταβλητή abstracts) αλλά με πέταξε εκτός (τέλειωσε η μνήμη...). Οπότε τελικά δοκίμασα να το κάνω αφότου είχα χωρίσει σε train και test, το οποίο και πέτυχε. Εφάρμοσα PCA με 99% variance στις X_train,y_train και X_test. Η λογιστική παλινδρόμηση δίνει αποτέλεσμα 0.6769715784991139, λίγο χειρότερο από το αποτέλεσμα που πήραμε χωρίς αυτήν.

Δοκιμάζω και την LDA, η οποία δεν είναι μόνο μία τεχνική για dimensionality reduction, αλλά είναι και από μόνη της ένας classifier. Πετυχαίνει σχεδόν την ίδια απόδοση με την λογιστική παλινδρόμηση.

Όσο αφορά το scaling των δεδομένων, δοκίμασα και να βρω το μεγαλύτερο value από τα χαρακτηριστικά και μετά με ένα loop στα features να διαιρέσω το value κάθε feature με το max στοιχείο (.π.χ. αν έχω τα χαρακτηριστικά με τιμές 3,6,9 θα μετρατραπούν σε 0.33(3/9), 0.67(6/9) και 1(9/9). Με αυτό το τρόπο μπορούμε να πετύχουμε scaling αφού τα στοιχεία θα είναι στο [0,1].Παρ'όλα αυτά, η τακτική αυτή χειροτέρεψε το score στο Kaggle.

4.3 Ποιους αλγόριθμους δοκίμασα και γιατί; Ποια η απόδοση των μεθόδων και ο χρόνος εκπαίδευσης;

Δοκίμασα πολλούς αλγόριθμους και πήρα τα αποτελέσματα που φαίνονται στον παρακάτω πίνακα:

Αλγόριθμος	Ακρίβεια	Χρόνος εκτέλεσης (sec)
BernoulliNB	0.5286357954311304	1.2795186042785645
GaussianNB	0.637980502859550	1.1321206092834473
MultinomialNB	0.584332687702332	1.6254520416259766
Logistic Regression	0.7177905682926494	8.780513048171997
KNN	0.6386998548474982	18.467426300048828
Random Forest	0.7564354758208901	966.0988993644714
MLP	0.7188547147089395	202.66692900657654
Adaboost	0.7186486622617232	258.907026052475
LDA (1 component)	0.7185030518656904	2.481757402420044

Figure 9: Η απόδοση των αλγορίθμων μηχανικής μάθησης.

Το GridSearchCV είναι μια λειτουργία βιβλιοθήκης που είναι μέλος του πακέτου model_selection του sklearn. Βοηθά ένα βρόχο μέσα από προκαθορισμένες υπερπαραραμέτρους για να ταιριάζει ο εκτιμητής μας (μοντέλο) στο σετ εκπαίδευσής μας. Έτσι, στο τέλος, μπορούμε να επιλέξουμε τις καλύτερες παραμέτρους από τις υπερπαραμέτρους που αναφέρονται. Γενικά το GridsearchCV , ανάλογα και με το πόσες παραμέτρους του δώσουμε να εξετάσει, παίρνει αρκετό χρόνο για να αποφανθεί. Επομένως, η ιδέα ήταν να διαλέξω τις καλύτερες 2 μεθόδους από πλευράς

απόδοσης και να εφαρμόσω έλεγχο Gridsearch σε αυτές για να βελτιστοποιήσω το τελικό αποτέλεσμα.

4.4 Αναφορά σε ό,τι δε λειτούργησε και γενικότερα σε ό,τι είναι ενδιαφέρον

Οι τεχνικές μείωσης διαστάσεων δε δούλεψαν. Η LDA (που από μόνη της είναι τεχνική μείωσης διαστάσεων) φέρνει παρόμοια αποτελέσματα με την λογιστική παλινδρόμηση. Η PCA αποφέρει λίγο χειρότερα αποτελέσματα αφότου εφαρμοστεί.

Γενικά ο γράφος ήταν τεράστιος και πολλά ερωτήματα, κυρίως σχετικά με την ανάλυση του, ήταν αδύνατον να εφαρμοστούν πάνω στο μέγεθός του. Επομένως η ιδέα ήταν η δημιουργία ενός (αρκετά μικρότερου) γράφου που να αντιπροσωπεύει τον αρχικό γράφο. Ο καλύτερος τρόπος είναι να βρούμε αν υπάρχει giant component , το οποίο και θα κάνει εξαιρετικά αυτή τη δουλειά, αλλά και αυτό θέλει πάρα πολύ χρόνο για να εκτελεστεί. Δοκιμάσαμε να πάρουμε τον υπογράφο με τους πρώτους 10000 και αυτόν με τους τελευταίους 10000 κόμβους. Μερικές από τις εντολές (οι πολύ χρονοβόρες) εκτελέστηκαν πάνω στον πρώτο γράφο και οι υπόλοιπες στον αρχικό.

5 Τελικά συμπεράσματα

Στο challenge αυτό έχουμε την ευκαιρία να δοκιμάσουμε πολλούς αλγορίθμους για το fitting του μοντέλου και με στόχο την ελαχιστοποίηση του log loss. Καθοριστικό ρόλο παίζει και ο τρόπος που χωρίζουμε τα δεδομένα, τα features που επιλέγουμε κλπ.

Το project αυτό ήταν μια ευκαιρία να εισχωρήσω στα challenges του Kaggle και να πάρω μια πρώτη γεύση από τον κόσμο του Data Science, στον οποίο στοχεύω να εμβαθύνω μετά το πτυχίο.