Γιακουμάκη Ελένη 4651
Δημητροπούλου Μαρία 4664

NLP 053 - CSE UOI 2025 Kaggle Challenge

Ομάδα Kaggle: ****ChatGPTeam****

0. Εισαγωγή

Η παρούσα εργασία ασχολείται με το πρόβλημα του *Citation Prediction*, το οποίο εντάσσεται στην κατηγορία του link prediction σε γράφους επιστημονικών παραπομπών (citation networks). Ο στόχος είναι, δεδομένου ενός ζεύγους επιστημονικών άρθρων, να προβλέψουμε αν το ένα κάνει αναφορά στο άλλο, αξιοποιώντας τα διαθέσιμα δεδομένα:

- το περιεχόμενου κειμένου (***abstracts***) των άρθρων
- τους συγγραφείς κάθε άρθρου (***authors***)
- τη δομή του citation graph, όπως ορίζεται από την λίστα ακμών (***edgelist***), όπου κάθε ακμή αντιπροσωπεύει μια παραπομπή από ένα άρθρο προς ένα άλλο

Η προσέγγιση που ακολουθήσαμε βασίστηκε στην εξαγωγή και συνδυασμό κατάλληλων χαρακτηριστικών από τα παραπάνω δεδομένα με σκοπό την εκπαίδευση μοντέλων, τα οποία προβλέπουν την πιθανότητα ύπαρξης citation μεταξύ δύο άρθρων.

Η αξιολόγηση της μεθόδου πραγματοποιήθηκε στο πλαίσιο διαγωνισμού μέσω Kaggle.

Ακολουθούν οι περιγραφές των βασικών βημάτων που ακολουθήθηκαν για την υλοποίηση της εργασίας: preprocessing, κατασκευή χαρακτηριστικών, επιλογή και σύγκριση μοντέλων, καθώς και αξιολόγηση αποτελεσμάτων.

1. Προεπεξεργασία (Preprocessing)

Η προεπεξεργασία περιλαμβάνει τον καθαρισμό, την τυποποίηση και τη μετατροπή των αρχικών δεδομένων σε μορφή κατάλληλη για ανάλυση και εξαγωγή χαρακτηριστικών.

Αυτά τα βήματα δεν εφαρμόζονται μαζικά στην αρχή. Κάθε χαρακτηριστικό που κατασκευάζεται περιλαμβάνει μόνο την προεπεξεργασία που χρειάζεται. Έτσι, κάθε τμήμα κώδικα δεν εξαρτάται από κοινή αρχικοποίηση και είναι πιο αυτόνομο.

- ***Για τα abstracts*:** Φορτώνουμε το αρχείο *abstracts.txt* και αντιστοιχίζουμε κάθε κείμενο με το *paper_id* του, αφού πρώτα μετατρέψουμε τα IDs σε string (*str*) για ασφαλές merging. Επίσης, αφαιρούμε κενές εγγραφές (*NaN*) και εφαρμόζουμε lowercasing και απλό tokenization (με *.split()*), ώστε να κατασκευάσουμε word sets για lexical-overlap-based χαρακτηριστικά.
- ***Για τους auuthors***: Διαβάζουμε το αρχείο *authors.txt* και για κάθε άρθρο αποθηκεύουμε το αντίστοιχο σύνολο συγγραφέων. Αυτή η πληροφορία χρησιμοποιείται αργότερα υπολογισμό similarity μετρικών (όπως Jaccard).
- ***Για τα edgelist*:** Φορτώνουμε το αρχείο *edgelist.txt* και από τις σχέσεις ανάμεσα στα άρθρα κατασκευάζουμε έναν κατευθυνόμενο γράφο (μέσω της βιβλιοθήκης network) με κόμβους τα άρθρα και ακμές τις παραπομπές.

2. Feature Engineering

Η δημιουργία των χαρακτηριστικών (*features*) είναι ίσως το πιο σημαντικό κομμάτι της εργασίας, αφού μέσω αυτών προσπαθούμε να αποτυπώσουμε πιθανές σχέσεις ανάμεσα σε δύο άρθρα, με βάση τα διαθέσιμα δεδομένα.

Για να το πετύχουμε, πρώτα χρειάζεται να φτιάξουμε ένα σύνολο από ζεύγη άρθρων για τα οποία γνωρίζουμε αν υπάρχει παραπομπή ή όχι. Χρησιμοποιώντας το edgelist.txt δημιουργούμε θετικά παραδείγματα (όπου υπάρχει παραπομπή) και αρνητικά παραδείγματα (όπου δεν υπάρχει), ώστε το μελλοντικό μοντέλο να ‘’μάθει’’ να ξεχωρίζει τις δύο περιπτώσεις. Από τα ζεύγη φτιάχτηκε το ***train_pairs.csv***, το οποίο (μαζί με το ***test.txt***) χρησιμοποιείται στην εξαγωγή χαρακτηριστικών.

Κάθε feature υπολογίζεται ξεχωριστά και αποθηκεύεται σε δικό του *.csv* αρχείο (για test και train ζεύγη), ώστε να μπορούμε να τα συνδυάζουμε χωρίς να χρειαστεί να επανεκτελέσουμε τον ίδιο χρονοβόρο κώδικα. Επιπλέον, κάθε κομμάτι κώδικα που παράγει κάποιο feature είναι αυτόνομο (αρκεί να έχουν προηγηθεί τα imports και τα paths), κάτι που μας επέτρεψε να δουλεύουμε πιο ευέλικτα.

2.1. Κειμενικά Χαρακτηριστικά (Textual Features)

Ξεκινάμε την εξαγωγή χαρακτηριστικών από τα abstracts, με την υπόθεση ότι αν δύο άρθρα είναι θεματικά κοντινά, είναι πιθανότερο να υπάρχει παραπομπή από το ένα στο άλλο.

- ***TF-IDF & Cosine Similarity:***

Για να μετρήσουμε πόσο παρόμοια είναι δύο abstracts ως προς τις λέξεις που χρησιμοποιούν, εφαρμόσαμε την τεχνική ***TF-IDF***, χρησιμοποιώντας την κλάση *TfidfVectorizer* της βιβλιοθήκης *scikit-learn*. Η λογική πίσω από το *TF-IDF* είναι να δίνει μεγαλύτερη βαρύτητα σε λέξεις που είναι σημαντικές και σπάνιες και μικρότερη σε γενικές λέξεις που επαναλαμβάνονται συχνά (stop words). Με αυτόν τον τρόπο, κάθε abstract μετατράπηκε σε διάνυσμα πολλών διαστάσεων, και για κάθε ζεύγος papers υπολογίσαμε την *cosine similarity* μεταξύ των αντίστοιχων διανυσμάτων τους. Η cosine similarity μετράει την ‘’γωνιακή απόσταση’’ των δύο διανυσμάτων, και επομένως μας δίνει μια ένδειξη για το πόσο κοντά είναι θεματικά τα δύο abstracts (για υψηλή ομοιότητα τιμή κοντά στο 1, για χαμηλή τιμή κοντά στο 0).

Η διαδικασία αυτή εφαρμόζεται και για τα ζεύγη του train_pairs.csv και για τα test.txt, και τα αποτελέσματα αποθηκεύονται στα αρχεία ***text_similarity_train.csv*** και ***text_similarity_test.csv***. Αυτά τα αρχεία χρησιμοποιούνται στη συνέχεια για τη δημιουργία των τελικών input datasets.

- ***Sentence-BERT & Cosine Similarity:***

Για να αξιοποιήσουμε πιο εξελιγμένα χαρακτηριστικά κειμένου που λαμβάνουν υπόψιν και την σημασιολογική πληροφορία (το γενικό νόημα), χρησιμοποιείται το προεκπαιδευμένο μοντέλο ***Sentence-Bert***, συγκεκριμένα την εκδοχή *all-MiniLM-L6-v2*, μέσω της βιβλιοθήκης *sentence-transformers*. Το μοντέλο αυτό μετατρέπει κάθε abstract σε ένα διάνυσμα (embedding) 768 διαστάσεων, το οποίο συμπυκνώνει το συνολικό του νόημα.

Σε αντίθεση με το *TF-IDF* που συγκρίνει κυρίως λέξεις, το *Sentence-Bert* επιτρέπει σύγκριση νοήματος μεταξύ δύο κειμένων, ακόμα και όταν αυτό εκφράζεται από διαφορετικό λεξιλόγιο.

Τα embeddings υπολογίζονται για όλα τα abstracts και αποθηκεύονται για χρήση στα επόμενα βήματα. Στη συνέχεια, για κάθε ζεύγος από τα *train_pairs.csv και test.txt,* υπολογίζεται η cosine similarity ανάμεσα στα embeddings των δύο abstracts, ως δείκτη νοηματικής ομοιότητας. Τα αποτελέσματα αποθηκεύονται στα αρχεία ***bert_similarity_train.csv*** και ***bert_similarity_test.csv*** για την δημιουργία των τελικών input datasets.

- ***Lexical Overlap & Text Ratio:***

Παράλληλα με τα πιο σύνθετα χαρακτηριστικά που βασίζονται σε διανυσματικές αναπαραστάσεις, υπολογίζουμε και μερικά πιο απλά metrics που βασίζονται στην άμεση λεξιλογική επικάλυψη δύο abstracts. Από τα tokens κάθε abstract δημιουργούμε σύνολα λέξεων, και για κάθε ζεύγος υπολογίζουμε:

- ***shaired_word_count:*** τον αριθμό κοινών λέξεων

- ***shaired_word_ratio:*** το ποσοστό κοινών λέξεων ως προς το συνολικό πλήθος
- ***text_ratio:*** πόσες φορές μεγαλύτερο (ή μικρότερο) είναι ένα κείμενο σε σχέση με ένα άλλο

Αυτά τα χαρακτηριστικά αποτυπώνουν γρήγορες ‘’επιφανειακές’’ σχέσεις και παρότι έχουμε ήδη *TF-IDF* και *Sentence-Bert,* αυτά τα πιο απλά metrics προσφέρουν μια συμπληρωματική οπτική, ειδικά σε περιπτώσεις που τα σύνθετα μοντέλα δεν αποδίδουν καλά ή το κείμενο είναι μικρό.

Οι υπολογισμοί γίνονται και τα *train_pairs.csv και test.txt,* και τα αποτελέσματα αποθηκεύονται στα αντίστοιχα csv αρχεία (***shared_features, shared_word_count, shared_word_ratio, text_ratio***) για ενσωμάτωση στο τελικό dataset.

2.2. Χαρακτηριστικά Γράφου (Graph-based Features)

Εκτός από τα κείμενα, σημαντική πληροφορία προέρχεται και από την θέση των άρθρων μέσα στο ***citation graph***, δηλαδή το πώς συνδέονται μεταξύ τους μέσω παραπομπών.

Φορτώνουμε το αρχείο *edgelist.txt* και κατασκευάζουμε έναν γράφο με τη βιβλιοθήκη *networkx*, όπου κάθε κόμβος είναι ένα άρθρο και κάθε ακμή αντιστοιχεί σε παραπομπή.

Η βασική μας υπόθεση είναι ότι η θέση δύο άρθρων μέσα στον γράφο και ο τρόπος που συνδέονται με άλλα άρθρα μπορεί να σχετίζεται με το αν το ένα παραπέμπει στο άλλο. Για τον υπολογισμό των χαρακτηριστικών χρησιμοποιούμε τόσο τον κατευθυνόμενο γράφο (*DiGraph*), όσο και μια μη κατευθυνόμενη εκδοχή (*G_undirected*), ανάλογα με τη μετρική.

Τα χαρακτηριστικά που εξάγονται για κάθε ζεύγος άρθρων είναι τα εξής:

- ***Common Neighbors:*** Υπολογίζουμε τον αριθμό κοινών γειτόνων των δύο κόμβων στον μη κατευθυνόμενο γράφο. Η ύπαρξη κοινών γειτόνων μπορεί να υποδηλώνει θεματολογική συνάφεια ή επιστημονική "συγγένεια" μεταξύ των άρθρων.
- ***Jaccard Coefficient:*** Μετράει την αναλογία των κοινών γειτόνων των δύο άρθρων σε σχέση με το συνολικό πλήθος των διαφορετικών γειτόνων τους. Είναι μια πιο ισορροπημένη εκδοχή του χαρακτηριστικού Common Neighbors, καθώς προσαρμόζεται στον αριθμό των συνδέσεων που έχει κάθε άρθρο και επιτρέπει δίκαιη σύγκριση ακόμη και όταν τα άρθρα έχουν πολύ διαφορετικό “μέγεθος” στο γράφο.
- ***Adamic-Adar Index:*** Δίνει περισσότερο βάρος σε κοινούς γείτονες που είναι πιο "μοναδικοί", δηλαδή που έχουν μικρό βαθμό στον γράφο. Θεωρείται πιο ευαίσθητη μετρική για προβλήματα link prediction.

- ***Shortest Path Length:*** Υπολογίζουμε τη μικρότερη απόσταση (σε πλήθος ακμών) μεταξύ των δύο άρθρων στον κατευθυνόμενο γράφο. Όσο μικρότερη η απόσταση, τόσο πιθανότερο είναι τα άρθρα να σχετίζονται, έστω και έμμεσα.

Τα παραπάνω χαρακτηριστικά υπολογίζονται για τα ζεύγη του *train_pairs.csv* και *test.txt*, και αποθηκεύονται στα αρχεία ***graph_features_train.csv*** και ***graph_features_test.csv***. Στη συνέχεια, το χαρακτηριστικό *adamic_adar* κανονικοποιείται με λογαριθμική κλίμακα (log1p) για μείωση των ακραίων τιμών, και όλα τα χαρακτηριστικά ενσωματώνονται στα τελικά datasets για την εκπαίδευση των μοντέλων.

2.3. Χαρακτηριστικά Συγγραφέων (Author-based Features)

Εκτός από το περιεχόμενο των άρθρων και τη θέση τους στον citation graph, εξετάζουμε και πληροφορία σχετική με τους συγγραφείς κάθε άρθρου. Η βασική ιδέα είναι ότι η σύνδεση δύο άρθρων μπορεί να οφείλεται και σε σχέσεις μεταξύ των συντακτών τους — για παράδειγμα, αν έχουν συνεργαστεί ξανά, αν ανήκουν σε κοινές ερευνητικές ομάδες ή συμμετέχουν σε παρόμοια επιστημονικά δίκτυα.

Χρησιμοποιούμε το αρχείο *authors.txt*, το οποίο περιέχει τους συγγραφείς κάθε paper, και υπολογίζουμε την ***Author Jaccard Similarity*** για κάθε ζεύγος άρθρων. Η μετρική αυτή εκφράζει την αναλογία κοινών συγγραφέων σε σχέση με το συνολικό πλήθος διαφορετικών συγγραφέων των δύο άρθρων. Είναι μια κανονικοποιημένη προσέγγιση, που επιτρέπει δίκαιη σύγκριση ακόμα και όταν τα άρθρα έχουν πολύ διαφορετικό αριθμό συντακτών.

Οι τιμές υπολογίζονται τόσο για το *train_pairs.csv* όσο και για το *test.txt*, και αποθηκεύονται στα αρχεία ***author_similarity_train.csv*** και ***author_similarity_test.csv*** αντίστοιχα, ώστε να ενσωματωθούν στα τελικά input datasets.

2.4. Ενοποίηση & Τελική Προετοιμασία Δεδομένων

Αφού υπολογιστούν όλα τα χαρακτηριστικά ξεχωριστά και αποθηκευτούν σε αρχεία *.csv*, το επόμενο βήμα είναι να τα ενοποιήσουμε σε ένα ενιαίο σύνολο εισόδου για τα μοντέλα. Για κάθε ζεύγος άρθρων στο *train_pairs.csv* και *test.txt*, συνδυάζουμε τις στήλες των αντίστοιχων χαρακτηριστικών και δημιουργούμε τα αρχεία ***X_train.csv*** και ***X_test.csv***. Παράλληλα, οι ετικέτες (0 ή 1) από το *train_pairs.csv* αποθηκεύονται στο ***y_train.csv***.

Η συγχώνευση των χαρακτηριστικών γίνεται με βάση το *pair_id*, ώστε κάθε σειρά στα τελικά datasets να περιλαμβάνει όλα τα διαθέσιμα χαρακτηριστικά για το αντίστοιχο ζεύγος άρθρων. Το κάθε .csv αρχείο που προκύπτει από προηγούμενα blocks (π.χ. *text_similarity_train.csv*, *graph_features_test.csv*, *author_similarity_train.csv* κ.λπ.) έχει ως στόχο να είναι πλήρως

αυτόνομο και να μην εξαρτάται από προηγούμενη εκτέλεση άλλου κώδικα. Αυτή η προσέγγιση μάς επιτρέπει να ανανεώνουμε ή να αφαιρούμε συγκεκριμένα χαρακτηριστικά χωρίς να ξανατρέχουμε όλο το pipeline.

Με αυτό τον τρόπο, φτιάχνονται τα τελικά input datasets για τα μοντέλα ταξινόμησης που θα παρουσιαστούν στην επόμενη ενότητα.

3. Models, tuning, and comparison

Μετά τη δημιουργία του τελικού dataset, επικεντρωνόμαστε στην εκπαίδευση μοντέλων ταξινόμησης και στην αξιολόγηση των προβλέψεών τους με βάση το score του leaderboard. Οι οργανωμένες δοκιμές που πραγματοποιήθηκαν βρίσκονται στο παραδοτέο αρχείο ***submission_experiments.ipynb***, μέσα στον φάκελο *other_notebooks*, και περιλαμβάνουν διαφορετικά μοντέλα και συνδυασμούς χαρακτηριστικών. Τα αρχεία που δημιουργήθηκαν από αυτές τις δοκιμές και υποβλήθηκαν στο *Kaggle* βρίσκονται στον φάκελο *submission_trials*.

3.1. Προσέγγιση & Πειραματική Λογική

Από την αρχή, επιλέγουμε να προσεγγίσουμε το πρόβλημα με σταδιακή εξέλιξη: ξεκινώντας από απλά μοντέλα και περιορισμένα χαρακτηριστικά, και σταδιακά εμπλουτίζοντας τα δεδομένα εισόδου και τις μεθόδους, ώστε να παρακολουθούμε καθαρά την επίδραση κάθε επιλογής στην τελική απόδοση.

Για κάθε υποβολή, χρησιμοποιούμε τα τελικά datasets *X_train*, *y_train* και *X_test*, που δημιουργήσαμε από τον συνδυασμό όλων των επιλεγμένων χαρακτηριστικών. Εκπαιδεύουμε το μοντέλο με βάση τα δεδομένα του *X_train* και τις αντίστοιχες ετικέτες του y_train, ενώ στη συνέχεια εφαρμόζουμε το μοντέλο στα δεδομένα του *X_test*, ώστε να προβλέψουμε την πιθανότητα παραπομπής για κάθε άγνωστο ζεύγος άρθρων.

Αρχικά δημιουργούμε ένα baseline χρησιμοποιώντας μόνο ένα ή δύο χαρακτηριστικά (π.χ. *TF-IDF similarity*), ώστε να έχουμε ένα σημείο αναφοράς. Στη συνέχεια, προσθέτουμε εναλλάξ νέα χαρακτηριστικά (π.χ. *Sentence-BERT*, μετρικές γράφου, συγγραφείς) ώστε να αξιολογούμε ποια από αυτά συνεισφέρουν ουσιαστικά στο τελικό αποτέλεσμα.

Κατά τη διάρκεια των πειραμάτων, εκπαιδεύουμε και αξιολογούμε κάθε μοντέλο με τα ίδια ζεύγη και συγκρίνουμε τις αποδόσεις μέσω του δημόσιου score στο *Kaggle leaderboard*. Η διαδικασία αυτή μας επιτρέπει να κατανοήσουμε τόσο τη συμπεριφορά των μοντέλων, όσο και τη σημασία κάθε ομάδας χαρακτηριστικών, οδηγώντας μας σε ένα τελικό setup που ισορροπεί απόδοση, απλότητα και αξιοπιστία.

3.2. Επιλεγμένα Μοντέλα & Παραμετροποίηση

Κατά τη διάρκεια των δοκιμών, χρησιμοποιήσαμε τρία διαφορετικά μοντέλα για την εκπαίδευση και πρόβλεψη: ένα απλό, ένα πιο "έξυπνο" αλλά γενικού τύπου, και ένα πιο σύγχρονο με καλύτερη απόδοση σε δομημένα δεδομένα.

Ξεκινήσαμε με ***Logistic Regression*** (της *LogisticRegression*), το οποίο είναι απλό γραμμικό μοντέλο για δυαδική ταξινόμηση. Είναι εύκολο στην κατανόηση και μας επέτρεψε να δούμε αν κάποια βασικά χαρακτηριστικά (όπως οι similarity μετρήσεις) μπορούν να δώσουν νόημα από νωρίς. Δεν χρειάζεται κάποια ιδιαίτερη ρύθμιση, οπότε το χρησιμοποιήσαμε κυρίως για να έχουμε ένα σημείο αναφοράς στις πρώτες υποβολές.

Στη συνέχεια χρησιμοποιήσαμε ***Random Forest*** (από την *RandomForestClassifier*), ένα μοντέλο που βασίζεται σε πολλά decision trees. Αντί να εκπαιδευτεί ένα μόνο δέντρο, το Random Forest δημιουργεί πολλά διαφορετικά δέντρα και συνδυάζει τη ''γνώμη'' όλων πριν καταλήξει στην τελική πρόβλεψη. Αυτό το κάνει πιο σταθερό και ικανό να μάθει πιο σύνθετες σχέσεις στα δεδομένα, χωρίς να κινδυνεύει εύκολα από υπερπροσαρμογή (*overfitting*). Χρησιμοποιήσαμε 100 δέντρα (*n_estimators*=100) και σταθερό seed (*random_state*=42) για να έχουμε σταθερά αποτελέσματα, χωρίς άλλες ειδικές ρυθμίσεις.

Τέλος, επικεντρωθήκαμε στο **XGBoost** (από την βιβλιοθήκη *xgboost*), ένα σύγχρονο και πολύ αποδοτικό μοντέλο που βασίζεται και αυτό σε δέντρα, αλλά τα εκπαιδεύει σταδιακά, βελτιώνοντας κάθε φορά το λάθος του προηγούμενου. Γενικά είναι από τα μοντέλα που πετυχαίνουν υψηλά σκορ σε τέτοιου τύπου δεδομένα, και σε εμάς όντως έδωσε τις καλύτερες επιδόσεις. Σε κάποιες περιπτώσεις το χρησιμοποιήσαμε όπως είναι, με τις προκαθορισμένες ρυθμίσεις, ενώ αλλού πειραματιστήκαμε με βασικές παραμέτρους όπως το βάθος των δέντρων, τον αριθμό των γύρων εκπαίδευσης και την ταχύτητα μάθησης.

Το τελικό μας σκορ προήλθε από δύο υποβολές με ***XGBoost***: μία με τις *default* ρυθμίσεις του μοντέλου και μία με ήπιες αλλαγές στις βασικές παραμέτρους. Επιπλέον, χρησιμοποιώντας σταθερό *training set* και αποφεύγοντας υπερβολική βελτιστοποίηση πάνω στο *leaderboard*, περιορίσαμε τον κίνδυνο υπερπροσαρμογής (*overfitting*).

3.3. Ανάλυση & Συνδυασμός Χαρακτηριστικών

Μετά την ολοκλήρωση και αποθήκευση των χαρακτηριστικών, δοκιμάσαμε διαφορετικούς συνδυασμούς για να αξιολογήσουμε την πραγματική συνεισφορά του καθενός. Ακολουθήσαμε μια επαναληπτική διαδικασία: εκπαιδεύαμε μοντέλο, υποβάλλαμε το αποτέλεσμα και ερμηνεύαμε το score για να αποφασίσουμε το επόμενο βήμα.

Ξεκινήσαμε με απλά setups, όπως μόνο *TF-IDF similarity* με ***Logistic Regression***, που έδωσαν χαμηλά σκορ αλλά λειτούργησαν ως baseline. Η προσθήκη του *Sentence-BERT* βελτίωσε άμεσα

την απόδοση, επιβεβαιώνοντας τη σημασία της σημασιολογικής πληροφορίας. Αργότερα ενσωματώσαμε χαρακτηριστικά από γράφο και συγγραφείς, δοκιμάζοντας ταυτόχρονα μοντέλα όπως ***Random Forest*** και ***XGBoost***.

Το *XGBoost* αποδείχθηκε πιο σταθερό και αποδοτικό, ακόμα και χωρίς παραμετροποίηση. Με απλές αλλαγές σε βασικές παραμέτρους και διαφορετικά subsets χαρακτηριστικών, φτάσαμε στις δύο κορυφαίες υποβολές. Η καλύτερη επιτεύχθηκε με τις default ρυθμίσεις, ενώ η δεύτερη με ελαφρύ tuning.

Όλη η πειραματική διαδικασία καταγράφεται στο αρχείο ***submission_experiments.ipynb***, το οποίο περιλαμβάνεται στον φάκελο other_notebooks και προσφέρει ένα καθαρό δείγμα της μεθοδολογίας που ακολουθήθηκε.

3.4. Τελικές Υποβολές και Αξιολόγηση

Οι δύο καλύτερες υποβολές που πετύχαμε βασίστηκαν και οι δύο στο μοντέλο ***XGBoost***, αλλά με διαφορετική επιλογή χαρακτηριστικών. Η υποβολή με το υψηλότερο score: *0.13758* (submission 13) χρησιμοποιεί όλα τα διαθέσιμα χαρακτηριστικά, με τις προκαθορισμένες ρυθμίσεις του μοντέλου, ενώ η δεύτερη καλύτερη με score: *0.14199* (submission 12) βασίζεται σε επιλεγμένα χαρακτηριστικά και ήπια παραμετροποίηση.

Η μικρή διαφορά στις αποδόσεις τους δείχνει ότι η επιτυχία δεν εξαρτάται αποκλειστικά από την πολυπλοκότητα ή το πλήθος των features, αλλά από τη σωστή χρήση βασικών πληροφοριών και τη σταθερότητα του μοντέλου.

Οι δύο αυτές υποβολές έχουν συμπεριληφθεί και στα δύο notebooks (***citation_prediction.ipynb*** και ***submission_experiments.ipynb***), ενώ τα τελικά αρχεία υποβολής βρίσκονται συγκεντρωμένα στους φακέλους *submission_trials* και *submissions.*

4. Περιβάλλον Υλοποίησης & Τεχνικές Πληροφορίες

Η υλοποίηση της εργασίας πραγματοποιήθηκε σε γλώσσα *Python* (έκδοση 3.11), μέσα από το περιβάλλον του *Visual Studio Code*, με χρήση του extension *Jupyter* για τη δημιουργία και εκτέλεση των notebooks (.ipynb).

Για τη διαχείριση των εξαρτήσεων και των βιβλιοθηκών χρησιμοποιήθηκε ξεχωριστό kernel, ενώ οι απαιτούμενες βιβλιοθήκες έχουν καταγραφεί στο αρχείο *requirements.txt*, το οποίο περιλαμβάνεται στα παραδοτέα.

Η τελική αναφορά συντάχθηκε στο Microsoft Word, ενώ η παρουσίαση του project δημιουργήθηκε στην πλατφόρμα Canva.

5. Βιβλιογραφία

Σημειώσεις διαλέξεων Μαθήματος 053 «Επεξεργασία Φυσικής Γλώσσας», Κωνσταντίνος Σκιάνης, CSE Πανεπιστήμιο Ιωαννίνων

https://arxiv.org/abs/1809.05679

https://dl.acm.org/doi/10.1145/1217299.1217301

https://dl.acm.org/doi/10.1145/2939672.2939785

https://xgboost.readthedocs.io/en/stable/get_started.html

https://mayurdhvajsinhjadeja.medium.com/jaccard-similarity-34e2c15fb524

https://www.researchgate.net/publication/378828038_Graph-based_substructure_pattern_mining_with_edge-weight