ΠΟΛΥΔΙΑΣΤΑΤΕΣ ΔΟΜΕΣ ΔΕΔΟΜΕΝΩΝ

ΣΙΑΣΙΟΣ ΧΡΗΣΤΟΣ **1088618**

ΚΑΣΙΔΗΣ ΣΩΤΗΡΙΟΣ **1084498**

ΤΡΟΜΠΟΥΚΗΣ ΣΩΤΗΡΙΟΣ ΣΤΥΛΙΑΝΟΣ **1084638**

Εισαγωγή

Σκοπός της παρούσας εργασίας είναι η υλοποίηση και η πειραματική αξιολόγηση τεσσάρων βασικών πολυδιάστατων δομών δεδομένων: k-d Trees, Quad Trees, Range Trees και R-trees. Στόχος είναι η αποδοτική διαχείριση δεδομένων σε χώρο k διαστάσεων (όπου $k \leq 5$) και η εκτέλεση ερωτημάτων περιοχής και πλησιέστερου γείτονα .

Επιπλέον, υλοποιήθηκε η τεχνική Locality Sensitive Hashing (LSH) για την εκτέλεση ερωτημάτων ομοιότητας σε κειμενικά χαρακτηριστικά των δεδομένων.

Περιγραφή

Χρησιμοποιήσαμε το σύνολο δεδομένων Movies Metadata Cleaned Dataset (1900-2025). Το dataset περιλαμβάνει metadata για ταινίες, όπως τίτλο, προϋπολογισμό, έσοδα, διάρκεια, δημοτικότητα και κριτικές χρηστών.

Επιλέξαμε 5 αριθμητικά χαρακτηριστικά για τις χωρικές δομές: budget, revenue, runtime, popularity, vote_average.

Επειδή τα δεδομένα των ταινιών έχουν πολύ διαφορετικές κλίμακες (π.χ. τα «Έσοδα» μετρώνται σε εκατομμύρια, ενώ η «Βαθμολογία» είναι από 0 έως 10), εφαρμόσαμε την τεχνική Min-Max Scaling. Η διαδικασία αυτή μετασχηματίζει όλες τις αριθμητικές τιμές ώστε να βρίσκονται στο διάστημα [0, 1]. Αυτό είναι απαραίτητο διότι οι αλγόριθμοι αναζήτησης υπολογίζουν αποστάσεις. Χωρίς την κανονικοποίηση, τα χαρακτηριστικά με μεγάλες τιμές θα κυριαρχούσαν στα αποτελέσματα, αλλοιώνοντας την αναζήτηση.

K-NN

Λειτουργώντας με τη λογική του πλησιέστερου γείτονα, το σύστημα αναζητά τα K πιο όμοια αντικείμενα, στην προκειμένη τις ταινίες του CSV που είναι περισσότερο όμοιες με τα ranges που ζητάμε. Η εφαρμογή του K-NN πάνω στις δομές που υλοποιήσαμε εξασφαλίζει ότι δεν θα ελέγξουμε όλα τα "κλαδιά" του δέντρου αφού θα αποκλείσουμε μερικά που αποκλείεται να έχουν κοντινούς γείτονες. Η απόσταση των σημείων υπολογίζονται με ευκλείδεια απόσταση στον 5διάστατο χώρο.

Μεθοδολογία

Υλοποιήσαμε στη γλώσσα Python.

Αναπτύξαμε τις δομές k-d Tree, Quad Tree, Range Tree, R-tree , LSH

k-d Tree: Για τη δομή k-d Tree , επιλέξαμε την έτοιμη υλοποίηση της βιβλιοθήκης scikit-learn. Η βασική ιδέα της λειτουργίας του είναι η κυκλική εναλλαγή των διαστάσεων σε κάθε επίπεδο του δέντρου.

Συγκεκριμένα, στη ρίζα του δέντρου, τα δεδομένα διαχωρίζονται με βάση την πρώτη διάσταση (π.χ. Budget), δημιουργώντας δύο ημι-χώρους. Στο αμέσως επόμενο επίπεδο, κάθε υπο-σύνολο διαχωρίζεται με βάση τη δεύτερη διάσταση (π.χ. Revenue), και η διαδικασία συνεχίζεται εναλλάξ για όλες τις k διαστάσεις.

Quad Tree: Την υλοποιήσαμε από την αρχή και την προσαρμόσαμε ώστε να λειτουργεί σε χώρο k διαστάσεων.

Η συγκεκριμένη υλοποίηση διαιρεί κάθε κόμβο που υπερβαίνει τη χωρητικότητα σε 2^k υπο-περιοχές. Για παράδειγμα, στις 5 διαστάσεις, κάθε κόμβος αποκτά 32 παιδιά. Αυτή η διάσπαση συνεχίζεται μέχρι κάθε φύλλο του δέντρου να περιέχει έναν αποδεκτό αριθμό σημείων που μας επιτρέπει να διαχειριστούμε την πυκνότητα των δεδομένων.

Range Tree: Την υλοποιήσαμε από την αρχή. Βασίζεται σε μονοδιάστατα δέντρα. Συγκεκριμένα, κατασκευάζεται ένα ταξινομημένο Δυαδικό Δέντρο Αναζήτησης για κάθε μία από τις 5 διαστάσεις των δεδομένων.

Όταν το σύστημα δέχεται ένα ερώτημα που αφορά πολλές διαστάσεις ταυτόχρονα, εκτελεί την αναζήτηση σε κάθε δέντρο ξεχωριστά και στη συνέχεια υπολογίζει την τομή των αποτελεσμάτων. Με αυτόν τον τρόπο, εντοπίζονται γρήγορα και με ακρίβεια μόνο οι εγγραφές που ικανοποιούν τα κριτήρια σε όλες τις διαστάσεις.

R-tree: Χρησιμοποιήσαμε την βιβλιοθήκη rtree wrapper του libspatialindex. Η δομή αυτή είναι σχεδιασμένη για την αποδοτική ευρετηρίαση χωρικών δεδομένων και είναι μια ισοζυγισμένη δομή.

Το R-tree ομαδοποιεί τα γειτονικά αντικείμενα χρησιμοποιώντας Minimum Bounding Rectangles - MBRs. Κάθε κόμβος του δέντρου περιέχει το μικρότερο δυνατό ορθογώνιο που περικλείει όλα τα αντικείμενα ή τους υπο-κόμβους που ανήκουν σε αυτόν. Η μέθοδος αυτή είναι ιδιαίτερα αποδοτική για ερωτήματα περιοχής (Range Queries), καθώς επιτρέπει τον γρήγορο αποκλεισμό μεγάλων τμημάτων του χώρου που δεν τέμνονται με την περιοχή αναζήτησης.

LSH (Locality Sensitive Hashing): Την υλοποιήσαμε για την εύρεση ταινιών με παρόμοια κειμενικά χαρακτηριστικά με τη μέθοδο MinHash.

Η βασική αρχή της μεθόδου είναι η πιθανοτική ομαδοποίηση. Αντί να συγκρίνουμε κάθε ταινία με όλες τις υπόλοιπες που θα ήταν πολύ χρονοβόρο, ο αλγόριθμος δημιουργεί σύντομες "υπογραφές" για κάθε ταινία. Ταινίες που έχουν μεγάλη ομοιότητα στο περιεχόμενό τους παράγουν, με μεγάλη πιθανότητα, την ίδια υπογραφή και τοποθετούνται στον ίδιο bucket. Έτσι, η αναζήτηση περιορίζεται μόνο στις ταινίες του ίδιου bucket, επιτυγχάνοντας ταχύτατους χρόνους απόκρισης ακόμα και σε μεγάλο όγκο δεδομένων.

Πειραματική αξιολόγηση

Καταγράφηκαν οι εξής χρόνοι σε δευτερόλεπτα:

build_time: Ο χρόνος που απαιτείται για την πλήρη κατασκευή της δομής και την εισαγωγή όλων των σημείων.

knn_time: Μέσος χρόνος απόκρισης για την εύρεση των k=5 πλησιέστερων γειτόνων.

range_time: Μέσος χρόνος απόκρισης για ερωτήματα περιοχής ακτίνας r=0.05.

lsh_build_time/lsh_query_time: Χρόνοι για την κατασκευή ευρετηρίου MinHash και αναζήτησης ομοιότητας κειμένου.

name	build_time	knn_time	range_time	lsh_build_time	lsh_query_time
k-d Tree + LSH	0.0042	0.0001	0.0001	2.4986	0.0001
Quad Tree + LSH	0.0142	0.0015	0.0003	2.4986	0.0001
Range Tree + LSH	0.0655	0.0031	0.0017	2.4986	0.0001
R-tree + LSH	0.0657	0.0030	0.0003	2.4986	0.0001

Ανάλυση αποτελεσμάτων

1.Χρόνος Κατασκευής (build_time)

Το k-d Tree εμφάνισε τον μικρότερο χρόνο κατασκευής (0.0042s). Αυτό είναι αναμενόμενο, καθώς η βιβλιοθήκη scikit-learn χρησιμοποιεί βελτιστοποιημένο κώδικα χαμηλού επιπέδου (C/C++ bindings).

Το Quad Tree είχε εξαιρετική επίδοση (0.0142s), όντας περίπου 3 φορές πιο αργό από το k-d tree, αλλά σημαντικά πιο γρήγορο από τις πιο σύνθετες δομές.

Τα Range Tree και R-tree ήταν τα πιο αργά στην κατασκευή (~0.065s). Στην περίπτωση του Range Tree, αυτό οφείλεται στην ανάγκη κατασκευής πολλαπλών δέντρων (ένα για κάθε διάσταση), ενώ στο R-tree οφείλεται στον υπολογισμό των ορίων (MBRs) και στο overhead της βιβλιοθήκης Python wrapper.

Η κατασκευή του LSH (2.49s) είναι η πιο χρονοβόρα διαδικασία, καθώς απαιτεί tokenization κειμένου και δημιουργία εκατοντάδων hash signatures για κάθε ταινία.

2.Χρόνος Αναζήτησης (knn_time):

Όλες οι δομές πέτυχαν εξαιρετικούς χρόνους απόκρισης, της τάξης των χιλιοστών του δευτερολέπτου.Το k-d tree παρέμεινε το πιο γρήγορο.

3.Ποιότητα Αποτελεσμάτων LSH:

```
PS D:\sotos\ΠΟΛΥΔΙΑΣΤΑΤΕΣ ΔΟΜΕΣ ΔΕΔΟΜΕΝΩΝ\polydiastates-main> py -u -c "from lsh_implementation import filtered_lsh_top_n; result, query = filtered_lsh_top_n(text_attribute='genre_names', top_n=10, year_range
=(2000,2020), popularity_range=(3,6), vote_average_range=(3,5), runtime_range=(30,60), origin_countries=('US','GB'), original_language='en'); print('Query:', query); print(result.to_string(index=False))"
Φόρτωση δεδομένων...
Query: Ultimate G's: Zac's Flying Dream
                                      title                                      genre_names  popularity  vote_average  runtime original_language origin_country  __year__
similarity
                        Spider's Web: A Pig's Tale                    ['Animation', 'Family']      3.0996           3.8       48                en          ['US']      2006
  0.500000
                                Bending the Light                 ['Documentary', 'Family']      4.1322           3.5       60                en          ['US']      2014
  0.500000
                            The Keys of Christmas                       ['Family', 'Music']      3.0808           4.3       47                en          ['US']      2016
  0.500000
                                   Ghost Patrol           ['Animation', 'Family', 'Mystery']      3.9403           4.3       50                en    ['CA', 'US']      2016
  0.333333
                             A Monsterous Holiday ['Animation', 'Comedy', 'Family', 'Horror']      3.0484           3.0       48                en          ['US']      2013
  0.250000
     Jimmy Neutron: Operation: Rescue Jet Fusion ['Animation', 'TV Movie', 'Adventure', 'Comedy', 'Family', 'Science Fiction']      5.2546           4.0       44                en          ['US']      2003
  0.166667
                  How Proust Can Change Your Life                           ['Documentary']      3.7264           4.7       58                en    ['US', 'GB']      2000
  0.000000
Orientations: Chris Doyle - Stirred But Not Shaken                           ['Documentary']      3.5613           3.0       52                en    ['US', 'AU']      2001
  0.000000
                                Solid Geometry                     ['Drama', 'Mystery']      5.0009           4.7       30                en          ['US']      2002
  0.000000
                                 Pass The Mic!                           ['Documentary']      3.4843           3.0       52                en          ['US']      2003
  0.000000
PS D:\sotos\ΠΟΛΥΔΙΑΣΤΑΤΕΣ ΔΟΜΕΣ ΔΕΔΟΜΕΝΩΝ\polydiastates-main>
```

Όπως φαίνεται στο στιγμιότυπο εκτέλεσης, πραγματοποιήθηκε ερώτημα για την ταινία "Ultimate G's: Zac's Flying Dream". Ο αλγόριθμος LSH, ρυθμισμένος να αναζητά ομοιότητα στο πεδίο

genre_names, επέστρεψε ταινίες με υψηλό δείκτη συνάφειας Jaccard.

Συγκεκριμένα:

Η πρώτη ταινία (*"Spider's Web"*) εμφανίζει ομοιότητα 0.50, καθώς μοιράζεται κοινά είδη (Family, Animation) με την ταινία αναφοράς.

Καθώς κατεβαίνουμε στη λίστα, η ομοιότητα μειώνεται σταδιακά (0.33, 0.25, 0.16), υποδεικνύοντας λιγότερα κοινά είδη.

Οι τελευταίες ταινίες της λίστας έχουν ομοιότητα 0.0, καθώς παρόλο που ικανοποιούν τα χωρικά φίλτρα (έτος, διάρκεια, χώρα), δεν έχουν κοινό σημασιολογικό περιεχόμενο στα Genres.

Το πείραμα αυτό επιβεβαιώνει ότι η μέθοδος MinHash λειτουργεί σωστά, φιλτράροντας αποτελεσματικά τη σημασιολογική πληροφορία μέσα από χιλιάδες εγγραφές.

Συμπεράσματα

Στην παρούσα εργασία μελετήσαμε την απόδοση πολυδιάστατων δομών δεδομένων σε συνδυασμό με τεχνικές ομοιότητας κειμένου για την ανάκτηση πληροφορίας σε πραγματικά δεδομένα ταινιών. Από την πειραματική διαδικασία προέκυψαν τα εξής βασικά συμπεράσματα:

Αποδοτικότητα Χωρικών Δομών: Οι δενδρικές δομές k-d tree, Quad tree, Range tree, R-tree απέδειξαν ότι μπορούν να διαχειριστούν αποτελεσματικά πολυδιάστατα δεδομένα k=5, επιτυγχάνοντας χρόνους απόκρισης της τάξης του χιλιοστού του δευτερολέπτου.

Σημασία του LSH: Η τεχνική LSH MinHash αποδείχθηκε απαραίτητη . Παρόλο που έχει το υψηλότερο κόστος κατασκευής λόγω της επεξεργασίας κειμένου, επιτρέπει την εκτέλεση σύνθετων ερωτημάτων που δεν θα ήταν εφικτά μόνο με γεωμετρικές μεθόδους.

Συνοψίζοντας, ο συνδυασμός spatial indexing για τα αριθμητικά φίλτρα και LSH για το κείμενο αποτελεί μια βέλτιστη στρατηγική για σύγχρονα συστήματα συστάσεων recommender systems.