ΑΝΑΚΤΗΣΗ ΠΛΗΡΟΦΟΡΙΑΣ

Εργασία: Μηχανή αναζήτησης ταινιών

Φάση 1: Αρχικός σχεδιασμός και συλλογή δεδομένων

Μέλη Ομάδας

Ονοματεπώνυμο: Ευάγγελος Ηλιάδης ΑΜ: 3117

Ονοματεπώνυμο: Βασίλης Νίκας ΑΜ: 3143

GitHub link: https://github.com/Vassilisn/Information-Retrieval

(1)

(α) Για τις ανάγκες της εργασίας του μαθήματος «Ανάκτηση Πληροφορίας» κληθήκαμε να συλλέξουμε έναν όγκο εγγράφων. Σκοπός είναι η δημιουργία ευρετηρίων για τον μεγάλο αυτό όγκο δεδομένων. Έτσι ώστε να μπορεί να επιτευχθεί αναζήτηση(ανάκτηση) πληροφορίας, με λέξεις κλειδιά από αυτή τη συλλογή. Μας δόθηκε η επιλογή μεταξύ εγγράφων σχετικά με ταινίες/ σειρές ή δημοσιεύσεων από κοινωνικά δίκτυα. Για τους σκοπούς της εργασίας θα επιλέξουμε έτοιμες συλλογές εγγράφων σχετικά με ταινίες/ σειρές. Η συλλογή αυτή βρίσκεται υπό την μορφή .csv αρχείων. Γνωρίζουμε ότι το εσωτερικό format αυτών των αρχείων χρησιμοποιεί το κόμμα (“,”) για τον διαχωρισμό των επιμέρους κελιών/ στηλών (columns) της εκάστοτε γραμμής (row). Οι δε γραμμές χωρίζονται με αλλαγή γραμμής “\n”. Στόχος μας είναι μέχρι το πέρας της εργασίας η υποστήριξη επιπλέον format αρχείων όπως .json, .xlsx etc. Τα αρχεία θα περιέχουν πληροφορίες σχετικά με τον τίτλο, τη χρονιά, την περιγραφή του περιεχομένου, την κατηγορία , τη διάρκεια, τη βαθμολογία, τους ηθοποιούς κ.α. Το πρόγραμμά μας θα παρέχει πληροφορίες για τις σημαντικότερες εξ’ αυτών. Η πηγή από την οποία αντλήσαμε τα έγγραφα είναι η ιστοσελίδα https://www.kaggle.com/datasets .

(β) Όπως αναφέραμε σκοπός του προγράμματός μας είναι η «Ανάκτηση Πληροφορίας» ταινιών. Αρχικά το πρώτο πράγμα με το οποίο έρχεται σε επαφή ο χρήστης του προγράμματός μας, είναι η γραφική διεπαφή(gui). Η ανάπτυξη του οποίου έγινε με τη βοήθεια της javax swing και πιο συγκεκριμένα της βιβλιοθήκης “JFrame”. Μία σύντομη περιγραφή κάποιων από τις λειτουργίες του gui:

- **Search bar:** Εδώ ο χρήστης πληκτρολογεί τις λέξεις κλειδιά που ενδιαφέρεται να ψάξει.
- **Field buttons:** Μέσω αυτών ο χρήστης θα καθορίζει εάν θέλει η λέξη κλειδί που πληκτρολόγησε να αναζητηθεί μέσα σε κάποιο συγκεκριμένο πεδίο (title, genre etc.).

- **History:** Πρόκειται για το ιστορικό αναζητήσεων του χρήστη.
- **File open:** Έτσι ο χρήστης μπορεί να επιλέγει κάθε τη συλλογή δεδομένων στην οποία επιθυμεί να πραγματοποιήσει την αναζήτηση.
- **Search result panel:** Εδώ παρουσιάζονται τα αποτελέσματα αποτελεσμάτων που ταιριάζουν με την αναζήτηση του χρήστη.

Όσον αφορά την δημιουργία ευρετηρίων θα χρησιμοποιήσουμε τη βιβλιοθήκη lucene της apache. Διατρέχοντας τα αρχεία, κατασκευάζουμε Document αντικείμενα (org.apache.lucene.document.Document) τα οποία περιέχουν ως πεδία τους τις σημαντικές πληροφορίες για την εκάστοτε ταινία. Έπειτα με τη βοήθεια της lucene θα κάνουμε parsing του κειμένου πληροφοριών για κάθε ταινία.Με άλλα λόγια θα «σπάσουμε» το κείμενο σε κομμάτια (tokens). Πρόκειται για τις λέξεις που το αποτελούν. Σε πρώτη φάση για την κατασκευή των ευρετηρίων χρησιμοποιήσαμε αντικείμενα τύπων “IndexWriter”, analyzer, “IndexWriterConfig”. Με λίγα λόγια ο “IndexWriter” είναι υπεύθυνος για τη δημιουργία και διατήρηση του index, ο analyzer είναι υπεύθυνος για το σπάσιμο του text σε κομμάτια και το “IndexWriterConfig” παραμετροποιεί τον “IndexWriter” να χρησιμοποιήσει τον analyzer κατά τη δημιουργία του ευρετηρίου. Εδώ αξίζει να αναφερθεί ότι υπάρχουν πολλά είδη analyzer ανάλογα με τον τρόπο που επιθυμούμε να κάνουμε το “tokenization” του κειμένου. Παράδειγμα σημαντικής προδιαγραφής, που πρέπει να ληφθεί υπόψιν κατά την επιλογή analyzer, είναι η απαλοιφή των “stop words”, με σκοπό την ελαχιστοποιήση του μεγέθους ευρετηρίου.

Επίσης, προκειμένου να καθίσταται εφικτή η ανάκτηση πληροφοριών δημιουργήσαμε έναν QueryParser. Με το όρο “query” αναφερόμαστε στα “ερωτήματα”, τις λέξεις κλειδιά που δίνει ο χρήστης προς αναζήτηση. Κατά αντιστοιχία με τη δημιουργία των ευρετηρίων, πρέπει και εδώ να λάβει χώρα ένα parsing του κειμένου του query. Είναι πολύ σημαντικό να χρησιμοποιηθεί ο ίδιος analyzer με αυτόν κατά την δημιουργία του index. Αυτό γιατί το κείμενο του query πρέπει να “αντιμετοπιστεί”/ ”μεταχειριστεί” με τον ίδιο τρόπο, με αυτόν του κειμένου που βρίσκεται στο ευρετήριο. Πιο συγκεκριμένα, πρέπει να είναι ίδιο το tokenization (π.χ. απαλοιφή “stop words”), το stemming (απομόνωση και διατήρηση μόνο της ρίζας της λέξης) κ.α. Προκειμένου να γίνεται πάντα σωστή αντιστοίχηση “ερωτημάτων” – αποτελεσμάτων. Επίσης σημαντική είναι και η πρόβλεψη πιθανών ορθογραφικών λαθών του χρήστη, καθώς και η ανακατεύθυνση αυτών στα “κοντινότερα” σωστά αποτελέσματα.

Επιπλέον, για την αναζήτηση των ταινιών θα χρησιμοποιήσουμε αντικείμενα τύπου TopDocs (org.apache.lucene.search.TopDocs) και ScoreDoc (org.apache.lucene.search.ScoreDoc)
Το αντικείμενο θα δείχνει στα τοπ N αποτελέσματα που ταιριάζουν καλύτερα με την αναζήτηση που έκανε ο χρήστης. Συγκεκρειμένα θα είναι pointers που δείχνουν σε αντικείμενα Document.

(2) Μετά την ανάγνωση του .csv αρχείου με τη συλλογή ταινιών, με τη βοήθεια της lucene δημιουργούμε Document αντικείμενα που περιέχουν τα σημαντικά στοιχεία κάθε ταινίας. Έπειτα η δομή που επιλέξαμε, για την αποθήκευση των αντικειμένων αυτών, είναι ένα ArrayList<Document>(). Όπως αναφέρθηκε και παραπάνω, το εσωτερικό format αυτών των csv αρχείων χρησιμοποιεί το κόμμα (“,”) για τον διαχωρισμό των επιμέρους κελιών/ στηλών (columns) της εκάστοτε γραμμής (row). Οι δε γραμμές χωρίζονται με αλλαγή γραμμής “\n”.

Το dataset που επιλέξαμε σε πρώτη φάση για την ανάπτυξη του προγράμματός μας είναι το ακόλουθο:

https://www.kaggle.com/datasets/harshitshankhdhar/imdb-dataset-of-top-1000-movies-and-tv-shows

Το συγκεκριμένο csv αρχείο περιέχει μία συλλογή 1000 ταινιών. Μέγεθος αρκετά ικανοποιητικό για τις ανάγκες ανάπτυξης του προγράμματος. Εννοείται πως αυτή η συλλογή μπορεί να εμπλουτιστεί μελλοντικά.