Proiect Data Mining

Descrierea codului:

Functia create_index() este folosita pentru a crea index-ul, in cadrul acestei functii se apleaza functia extract_title_and_content(file_path) care primeste ca argument locatia fisierului cu paginile de wikipedia.Aceasta functie sparge fisierul pe documente, documentele fiind formate din titlu si continut .Continutul este tokenizat urmand ca asupra fiecarui token sa se apeleze functia de “stemming” si eliminarea acestuia daca apartine listei de “stop-words” sau daca acesta nu contine exclusiv caractere de tip alfanumeric.In urma acestora, fiecare document va fi introdus in fisierul index.

Functia search_index(withCategory) care priemste ca argument o variabila booleana care indica daca se va folosi sau nu cateogoria intrebarii, implicit fiind setata ca “false” deschide fisierul index de la locatia specificata.Apoi urmeaza sa preluam intrebarile din fisierul “questions.txt”, intrabrile sunt supuse aceluiasi proces ca si continutul documentelor.Query-ul de cautare este construit pe baza intrebarilor prelucrate in functie de valoarea flag-ului withCategory, fiind alipita si categoria.Indexul ne returneaza ca raspuns la intrebare, titlul primului document semnificativ si comparam cu rezultatul asteptat.

Probleme intampinate din cauze specific Wikipedia:

- Spargerea documentelor dupa titlul care avea urmatorul format :

  “ [[Titlu]] ...content "

- Structura specifica Wikipedia care separa paragrafele prin secvente de forma “==Other== “ care trebuie eliminate din document.

Masurarea performantei:

Pentru a masura performanta sistemului ne folosim de precizie, adica verificam de cate ori primul rezultat obtinut corespunde celui asteptat intrebarii.Fiind intr-un scenariu de joc, precizia este mai relevanta deoarece se asteapta un rezultat rapid si corect.NDCG este folosit pentru evaularea calitatii de ranking si considera pozitia in rezultat al raspunsului corect.Aceasta masuratoare ar fi mai potrivita intr-un scenariu in care conteaza pozitionarea si rank-urile determinate.
Exista 4 cazuri :
1.Cand ne folosim de categorie si operatorul “AND” intre tokenuri. Precizie 7%.
2.Cand ne folosim de categorie si operatorul “OR” intre tokenuri. Precizie 20%.

3.Cand nu ne folosim de categorie si operatorul “AND” intre tokenuri. Precizie 9%.
4.Cand nu ne folosim de categorie si operatorul “OR” intre tokenuri.Precizie 14%.

Analiza erorilor:
In cel mai bun caz, sistemul a raspuns corect la 20 de intrebari din cele 100.Numarul de raspunsuri corect se poate datora faptului ca atat documentele cat si intrebarile si categoriile au fost supuse tokenizarii si a fost aplicat “stemming-ul” si eliminarea stop words, iar prin operatorul “OR” nu obliga sistemul pentru a face un match de 100% la continutul documentului cu cuvintele din intrebare ajunge daca o parte din aceste cuvinte apar in continutul documentului. Raspunsurile gresite se pot datora faptului ca sistemul nu poate interpreta sensul cuvintelor, ci doar le cauta aparitia.Un alt caz ar putea fi cautarea intrebarilor in anumite categorii, altele decat categoriile pe care sistemul le-a prelucrat cu o precizie ridicata.

Imbunatatirea cautarii:
Pentru imbunatatirea cautarii am ales sa ne folosim de ChatGPT, facand un call prin care ii cerem inteligentei artificiale sa aleaga un titlu relevant din cele 10 returnate de cautarea in index.Pentru a putea face aceasta alegere, ii oferim si intrebarea (indiciul) pentru a obtine o performantat cat mai ridicata.Cerem sa primim doar titlul fara alte detalii pentru a putea avea o comparatie cat mai simplificata.

Schimbarea preciziei la cele 4 cazuri:
1.Cand ne folosim de categorie si operatorul “AND” intre tokenuri. Precizie 10%.
2.Cand ne folosim de categorie si operatorul “OR” intre tokenuri. Precizie 24%.
3.Cand nu ne folosim de categorie si operatorul “AND” intre tokenuri. Precizie 8%.
4.Cand nu ne folosim de categorie si operatorul “OR” intre tokenuri.Precizie 17%.

In 3 dintre cazuri, precizia a crescut prin folosirea serviciului oferit de OpenAI, cel mai semnificativ fiind cazul in care ne folosim de categorie si operatorul “OR” intre tokenuri crescand precizia pana la 24%.Se poate observa si un caz in care precizia a fost diminuata cu 1%, in cazul in care nu ne folosim de categorie si folosim operatorul “AND” intre tokenuri.