Threshold top(k) algoritmus

Trinh Dinh Quang

Popis projektu

Projekt je implementace vyhledávání v databázi pomocí Threshold top(k) algoritmu a naivním sekvenčním průchodem. Vyhledávání je možné parametrizovat n-tice atributů, parametrem (k) či agregační funkcí. Výsledkem bude (k) objektů seřazené sestupně podle hodnoty vypočítané z vybraných atributů pomocí agregační funkce. Jednotlivé objekty je poté zobrazena v tabulce na webové aplikaci.

Způsob řešení

Způsob uložení dat
Data jsem uložil do 2D pole s řádky označený indexem a sloupce označeným atributem. Dále jsem vytvořil jednotlivé pole pro každý atribut, ve kterém je uložena dvojice (index v objekt v databázi, objekt). Pole je poté setříděna sestupně. Nakonec jsem použil haldu pro udržování k objektů.

Vyhledávání pomocí Threshold top(k) algoritmu je implementována následujícím způsobem:

- Procházím setříděné pole atributy sekvenčně, pro každý element lze už snadno vypočítat threshold z dalších vybraných atributů
- Najdu objekt podle indexu v databázi a vypočítám hodnotu agregační funkce z řádky databáze
- Přidávám objekt do haldy, udržuju si haldu tak, aby vždy obsahovala jen k-nejlepších objektů.
- Porovnám threshold s hodnotou vypočítané z objektu, pokud nejmenší hodnota v haldě je větší nebo rovno threshold, vrátím výsledek.

Vyhledávání sekvenčně:

- Procházím celou databázi a vypočítám pro každý objekt agregaci
- Udržuju haldu, pro k-nejlepších objektů a poté výsledek vrátím.

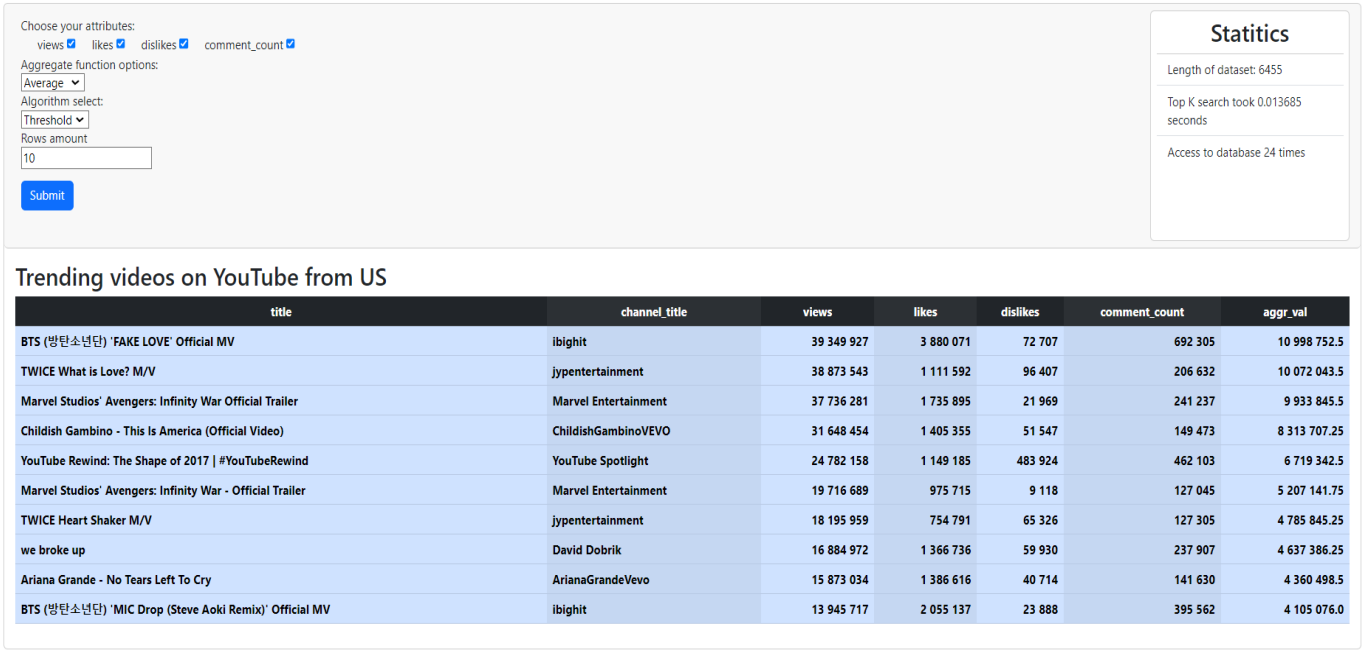
Implementace

Implementoval jsem aplikace v Pythonu, protože obsahuje velké množství knihoven a frameworků. Konkrétně jsem využíval webový framework Flask pro vytvoření webové aplikace.

Jako databáze jsem používal knihovnu “pandas”, který načte data z csv soubor a následně uloží do tzv. Dataframe (tabulka s řádky označený indexem a sloupce označeným atributem). “Pandas” navíc obsahuje řada užitečných metod pro zpracování dat. Pro webový interface jsem použil HTML + Bootstrap framework.

Příklad výstupu

Na obrázku můžeme vidět výstup aplikace. Ukázka nám vrátil populární videa v USA z Youtube. Byla použita average jako agregační funkce, která vypočítá průměr ze všech atributů a k tomu byla aplikována algoritmus Threshold. Aplikace poté vrátila 10 výsledků podle zadaných k parametr. Vpravo v menu je vidět i počet přístupu do databáze i doba zpracování dotazu.



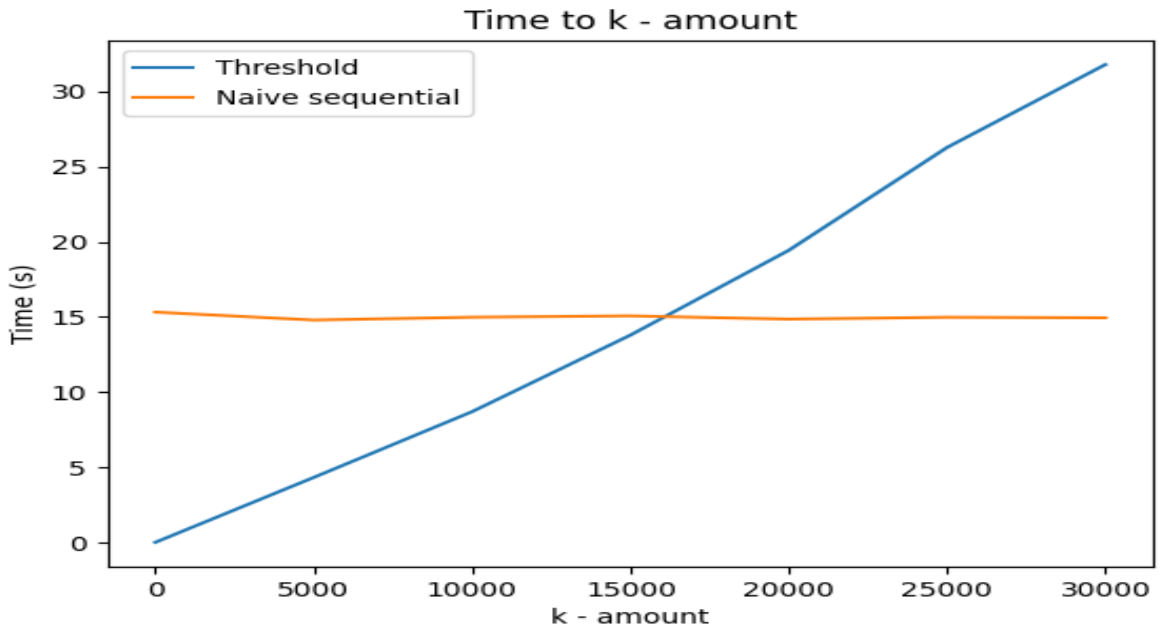
Trending videos on YouTube from US

title	channel_title	views	likes	dislikes	comment_count	aggr_val
BTS (방탄소년단) 'FAKE LOVE' Official MV	ibighit	39 349 927	3 880 071	72 707	692 305	10 998 752.5
TWICE What is Love? M/V	jypentertainment	38 873 543	1 111 592	96 407	206 632	10 072 043.5
Marvel Studios' Avengers: Infinity War Official Trailer	Marvel Entertainment	37 736 281	1 735 895	21 969	241 237	9 933 845.5
Childish Gambino - This Is America (Official Video)	ChildishGambinoVEVO	31 648 454	1 405 355	51 547	149 473	8 313 707.25
YouTube Rewind: The Shape of 2017 | #YouTubeRewind	YouTube Spotlight	24 782 158	1 149 185	483 924	462 103	6 719 342.5
Marvel Studios' Avengers: Infinity War - Official Trailer	Marvel Entertainment	19 716 689	975 715	9 118	127 045	5 207 141.75
TWICE Heart Shaker M/V	jypentertainment	18 195 959	754 791	65 326	127 305	4 785 845.25
we broke up	David Dobrik	16 884 972	1 366 736	59 930	237 907	4 637 386.25
Ariana Grande - No Tears Left To Cry	ArianaGrandeVevo	15 873 034	1 386 616	40 714	141 630	4 360 498.5
BTS (방탄소년단) 'MIC Drop (Steve Aoki Remix)' Official MV	ibighit	13 945 717	2 055 137	23 888	395 562	4 105 076.0

Experimentální sekce

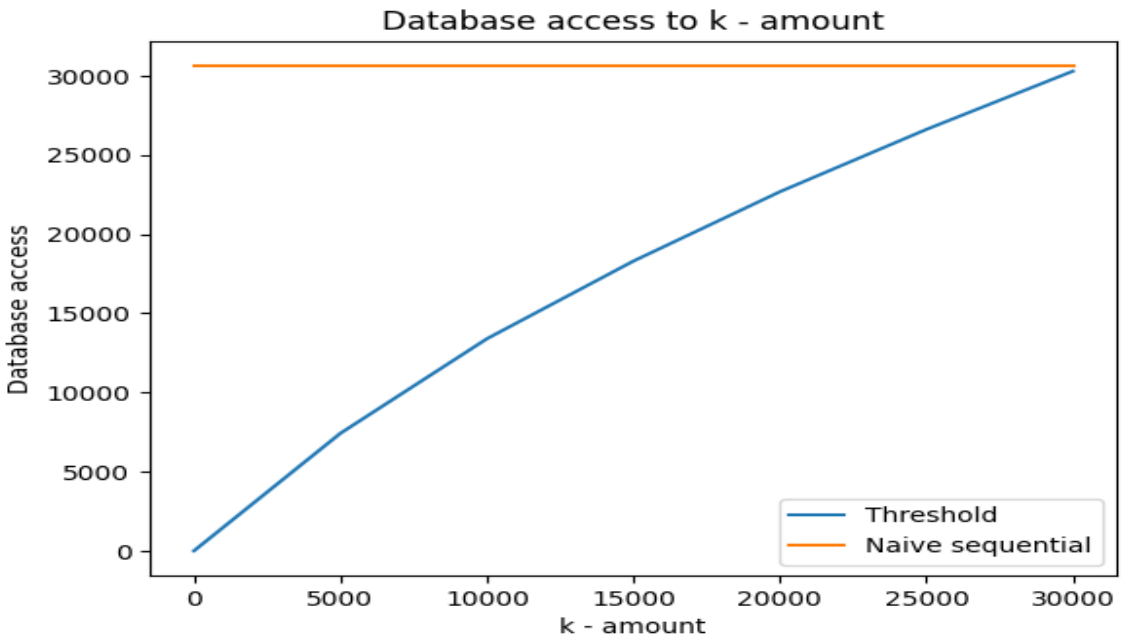
V tomto sekci jsem zvětšil dataset na 30 000 objektů. Porovnal jsem sekvenční průchod a Threshold algoritmus, jak se liší v počtu přístupu do databází a doba běhu s rostoucím “k” parametrem.

Porovnání doba běhu vzhledem k rostoucím “k” parametrem



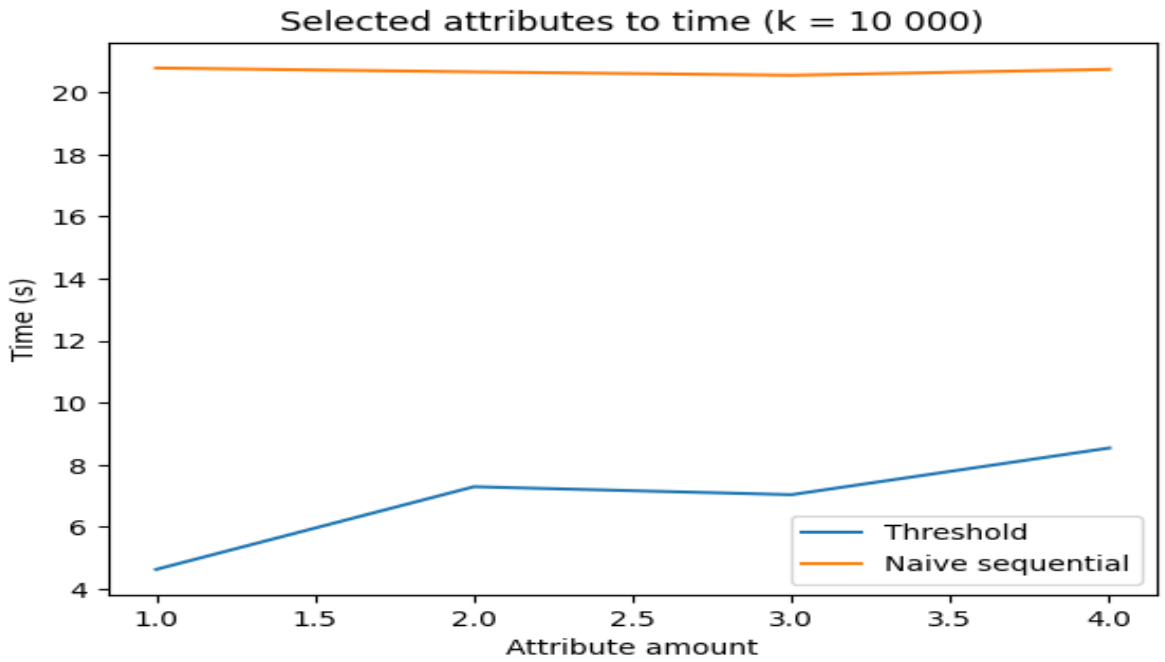
Z obrázku je vidět, že doba běhu algoritmus Threshold roste lineárně vzhledem k “k” parametru, naproti tomu sekvenční průchod je konstantní k “k” parametru. Vyplatí se použít Threshold algoritmus jen k malému “k” parametru.

Porovnání přístupu do databáze vzhledem k rostoucím “k” parametrem



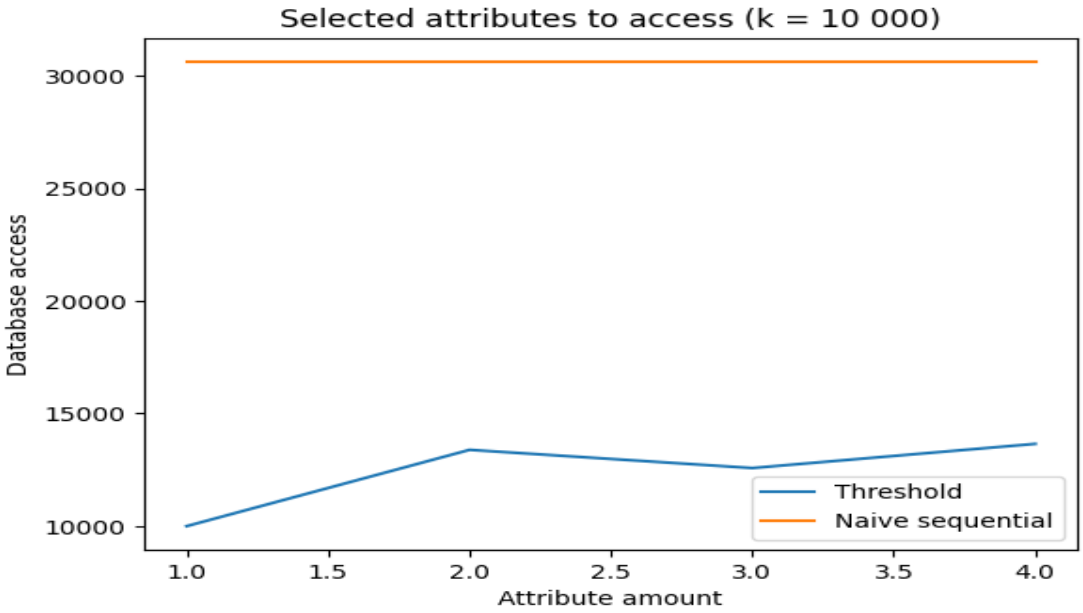
Z analýzy je jasné, že algoritmus Threshold využívá méně přístupů do databáze než sekvenční průchod, jelikož naivní průchod prochází vždy celou databázi.

Porovnání doba běhu vzhledem k počtu atributů



S rostoucím počtu atributů má jen malý vliv k efektivitě Threshold algoritmu, K sekvenčnímu průchodu nemá vůbec vliv.

Porovnání přístupu do databáze vzhledem k počtu atributů



S rostoucím počtu atributů má jen malý vliv k počtu přístupu do databáze Threshold algoritmu, K sekvenčnímu průchodu nemá vůbec vliv, prochází totiž vždy celou databázi.

Diskuse

Projekt jsem zpracoval podle znalost z přednášky, abych zjistil zda algoritmus je skutečně efektivní. Neměl jsem prostor pro zkoumání všechny části programu, jestli jsou už optimální. Dalo by se optimalizovat rychlost programu v algoritmech a datových strukturách. Daším nedostatkem aplikace je načítání data do paměti místo používání databáze pro dotazování. Bylo by lepší zahrnovat databáze do programu pro větší dataset. Mohl jsem také vybrat či vygenerovat dataset s vícemi atributy pro porovnání.

Závěr

Projekt byl pro mě docela přínosný, naučil jsem vytvořit web přes framework Flask. Měl jsem možnost pracovat s datami pomocí knihovny pandas. Naučil jsem také analyzovat nakonec jednotlivé funkce a následně vykreslit do grafu pomocí knihovny matplotlib. Jednotlivé algoritmus v programu fungoval podle mého očekávání. Nakonec podle výsledku porovnání jsem zjistil výhody a nevýhody Threshold top(k) algoritmu.